INTERACTIVE SYSTEMS AND THEIR APPLICATIONS

FEBRUARY 1979

KYOTA AOKI

INTERACTIVE SYSTEMS AND THEIR APPLICATIONS

by

KYOTA AOKI

Submitted in partial fulfillment of

the requirement for the degree of

DOCTOR OF ENGINEERING

(Information & Computer Sciences)

at

OSAKA UNIVERSITY

TOYONAKA, OSAKA, JAPAN

February 1979

ACKNOWLEDGEMENTS

The author would like to express his gratitude to Professor K. Tanaka, his thesis supervisor. Prof. Tanaka has been a constant source of invaluable advice and encouragement throughout the course of the research for this thesis. Special thanks are also due to Prof. M. Kizawa, Prof. T. Kasami, Prof. T. Fujisawa and Prof. N. Tokura of the Department of Information and Computer Sciences of Osaka University and Prof. S. Tsuji of the Department of Control Engineering of Osaka University.

He wishes to thank Associate Prof. J. Toyoda of the Department of Information and Computer Sciences of Osaka University and Associate Prof. M. Mizumoto of the Department of Management Engineering of Osaka Electro-Communication University, for their invaluable technical suggestions and encouragements through the course of his graduate studies.

He is also grateful to Associate Prof. S. Tamura, Dr. R. Ross and the colleagues of Prof. Tanaka's laboratory, especially, Mr. H. Miyoshi and Mr. N. Yokoya, for their relevant suggestions and useful discussions.

ABSTRACT

Kyota Aoki, Graduate School of Osaka University, February 1979.

Doctoral Thesis: Interactive Systems and their Applications.

The research here described is concerned with a formal system and its applications, in particular, a formal system which models systems interacting each other.

In this thesis, a new formal system: interactive system, is proposed, which is constructed with two web grammar systems and interaction functions which represent interactions between two web grammar systems. Many kinds of interactive systems are proposed and their abilities are studied.

For representing a problem, production systems has been used. A new production system is proposed, which is developed by extending descriptive power of the interactive system by introducing variables. Three examples of interactive graph production systems are shown. They are three coin problem, monkey and banana problem and block world manipulations problem. In the three coin problem, changing of situations in a data base is described. In the monkey and banana problem, try and error process is represented. And in block world manipulations problem, complex situations are represented.

Using the IGPS an image interpreting system is constructed. The image interpreting system is represented simply using the IGPS, and can be easily modified. The image interpreting system contains about 120 productions, but it is experimental one. The image interpreting system treats a simple visual world which is constructed with 5 kinds of objects.

TABLE OF CONTENTS

CHAPTER 1

Introduction

Mankind can not live without any information. And 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 result obtained by manipulating symbols can also successfuly be applied to real world.

For modelling languages many grammar systems have been proposed, and using those abstractions our understanding about languages have been developed. And grammar systems were used as production systems for expressing complex problems.

When production systems were first proposed by Post as a general computational mechanism [34], production system is very simple construct of a set of rules, a data base, and an interpreter. And the production system expresses symbol string manipulations.

Becoming problems treated more complex, production systems also have become more complex. For instance, in DENDRAL system [18, 39] the data base contains complex graph structures which represent molecules and molecular fragments.

Grammar systems also have been become more complex. For instance, grammar systems were used for representing patterns, and the grammar systems were used in pattern recognitions [20, 25].

For expressing complex structures, in 1969 Pfaltz and Rosenfeld introduced the notion of a web grammar [33], whose productions replace subwebs by subwebs. This notion provides a general formalism for modelling a wide variety of data structures, in paticular, 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 aspect of web grammars for various classes of graphs [26, 32, 36], 'Chomsky hierarchies' for such grammars [1-7, 31, 17], web acceptors [15, 16, 23, 28, 37, 38], pattern analysis [42] and data structure manipulation by web grammars [41].

In the field of artificial intelligence, we have trend to treat more and more complex probelms. So we find many cases that have interactions in the situations. And there exist a lot of problems that are easily solved through interactions. So we need a formal system which models situations interacting each other.

In this thesis, standing the preceding view points, formal models for complex situations interacting each other are proposed. And the formal models are developed into production systems and using that production system image interpretation is discussed.

In chapter 2 interactive systems are proposed, that are new formal systems which represent situations interacting each other. And an interactive system is defined as a formal system which is constructed with two web grammar systems interacting each other through interaction functions, and their abilities are studied. It is shown that the well-known quotation from Homer's Iliad:

"Two heads are better than one."

is true for formal systems, too.

In chapter 3, an interactive graph production system will be proposed. IGPS (Interactive Graph Production System) is developed by extending descriptive power of the interactive system by introducing variables. And IGPS is simplified by constricting interactive systems' formalisms. Three examples will be shown. Three examples are the three coin problem, monkey and banana problem and block world manipulations problem. In the three coin problem, changing of situations in a data base is described. In the monkey and banana problem, try and error process is represented. And in the block world manipulations problem, complex situations are represented.

In chapter 4, using the IGPS an image interpreting system will be constructed. The image interpreting system is represented simply using IGPS, and can be easily modified. The image interpreting system contains about 120 productions, but it is experimental one. So the image interpreting system treats a simple world which is contructed with 5 kinds of objects.

CHAPTER 2

Interactive Systems and their Abilities

2.1 Introduction

In recent years many robot systems and problem solving systems have been constructed. Most of them, for instance, STRIPS [20] is constructed with a goal oriented method and, as a result, it could handle little interaction. In the real world robot never have all knowledges of environments where they may take actions previously. Therefore, robots must take actions according to their assumptions, and confirm their assumptions by observing reactions. For instance, when a robot is going to recognize its environment with a TV-camera, it can correct its recognition by viewing the reaction to its action. Therefore in a robot planning, there must be used a dynamic method which has its base on interactions between a robot and its environment, not as usual statistic goal oriented method.

A few robot planning systems have considered interaction. But it has not been investigated theoretically and practically that what changes of the ability of a system are caused by interaction or what relations exist between systems with and without interactions. In this chapter, for handling the system which has interaction theoretically, we will define an interactive system which models the system which has interactions. To solve those problems, we study the abilities of interactive systems. First, we will discuss the condition for modelling systems with interactions. Second, we will propose formal definitions of interactive systems. Third, we will study the behaviors of the interactive systems of mode A which have transcendental correspondence between sub-systems which interact each other. Fourth, we will study the behaviors of the interactive systems of mode B which have no transcendental correspondence between sub-systems. Fifth, relations between interactive systems of mode A and mode B will be studied. And last, interactive systems which are constructed with two kinds of sub-systems will be studied.

2.2 Interactive systems

In this chapter, we will provide theoretical models for the systems such as in Fig. 2.1. For instance, system-1 is a robot and system-2 is an environment, or system-1 is a man and system-2 is a computer system.

Generally system-1 and system-2 are complex systems. In the sense of interaction, system-1 and system-2 are tokens which interact each other. In this chapter we call system-1 and system-2 *token systems*, and states of these *tokens*. Next we shall describe how handle token systems and tokens. A token system can be an automaton. But for this chapter's purpose it is desired that a token has structures in itself. Because token systems interact through tokens. Therefore graphs, which have structures in themselves, are fit for a token. So for token systems we use *web grammar systems* which are originated from

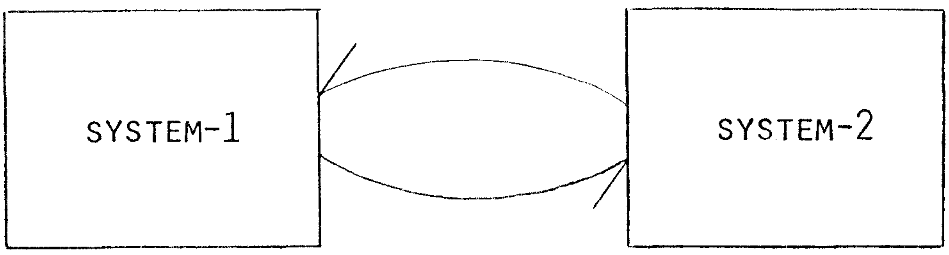


Fig. 2.1 Structure of a system which is handled in this chapter.

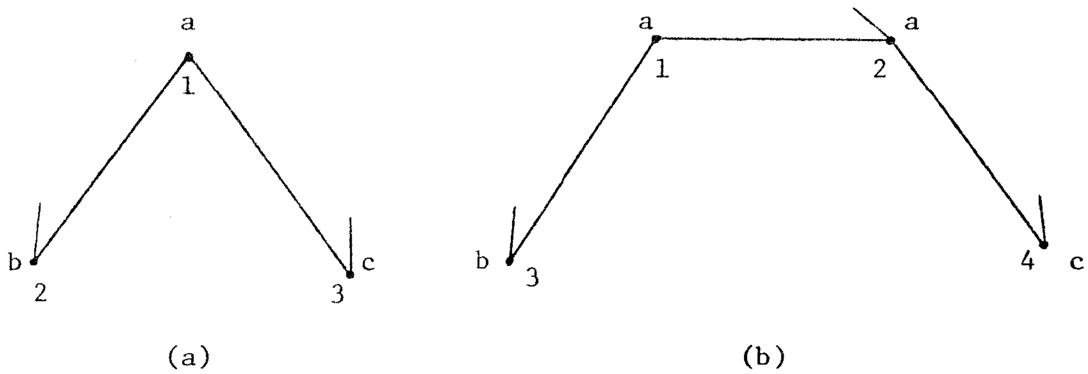


Fig. 2.2 Pictorial representations of tokens.

the work of Pfaltz and Rosenfeld [33] and studied by many researchers [1, 14]. A web grammar system handles graphs for its language. We use web grammar systems which handle labelled directed graphs. Labelled directed graphs are handled in this chapter by the forms in Fig. 2.2. A class of token systems corresponds to an ability or a complexity of it. An interactive system is constructed with two token systems and two *interaction functions* which model interactions between token systems. In Fig. 2.1 an arrow from system-1 to system-2 corresponds to an interaction function 1, and an arrow from system-2 to system-1 does to an interaction function 0. An interaction function 1 provides the set of productions of a token system 1 for a vertex of a token of a token system 1 in the context of a token of a token system 0.

The ability of an interactive system which is constructed with two token systems and two interaction functions is defined with the set of tokens which can be generated. More kinds of tokens an interactive system can generate, higher ability it has. Therefore we study relations among the families of the sets of tokens which can be generated by interactive systems.

2.3 Definitions of interactive systems

We shall describe the definitions of interactive systems and token systems which are sub-systems of interactive systems.

Definition 2.1. An interactive system is specified as a 4-tuple,

$$S = (T_0 , T_1 , f_0 , f_1),$$

where T_0 and T_1 are token systems, and f_0 and f_1 are interaction functions.

Definition 2.2. A token system T_i (i=0 or 1) is a 4-tuple,

$$T_i = (s_i , V_{ni} , V_{ti} , P_i),$$

where s_i is an initial token, V_{ni} is a finite set of nonterminal symbols, V_{ti} is a finite set of terminal symbols, and P_i is a finite set of productions of web grammar.

Definition 2.3. A token is represented by a 3-tuple,

$$t = (N , \nu , \delta),$$

where N is a nonempty set of vertices which is represented by natural numbers, ν: $N \to V_n \cup V_t$ is a label function, and δ: $N \times N \to \{0, 1\}$ is an edge function.

We show examples of a token and of a token system.

[Example 2.1] Let

$$N = \{ 1, 2, 3 \},$$

$$\nu(1) = a ,$$

$$\nu(2) = b ,$$

$$\nu(3) = c ,$$

and $$\delta(m, n) = \begin{cases} 1 & m=1 \text{ and } n=2,\ 3 \\ 0 & \text{otherwise} . \end{cases}$$

Then $t = (N, \nu, \delta)$ is a token and is shown in Fig. 2.2 (a).

[Example 2.2] Let

$$s = \mathrm{S} ,$$

$$V_n = \{\mathrm{A, B, C, S}\} ,$$

$$V_t = \{\mathrm{a, b, c}\} ,$$

and $P = \{\mathrm{S} \equiv> \mathrm{a\ B}\ ,\ \mathrm{B} \equiv> \mathrm{b\ C}\ ,\ \mathrm{C} \equiv> \mathrm{c\ A}\ ,\ \mathrm{A} \equiv> \mathrm{a\ B}\ ,\ \mathrm{C} \equiv> \mathrm{c}\}$.

Then $T = (s, V_n, V_t, P)$ is a token system.

An initial token s is usually S which is one vertex token, and we express a token system T by only the set of productions P usually. For instance the token system of Example 2.2 is represented by the form in Table 2.1. Hereafter an initial token s, a set of nonterminal symbols V_n and a set of terminal symbols V_t are not expressed explicitly.

Next we shall define an interaction function.

Definition 2.4. An interaction function is specified as

$$f_i : \tau_{1-i} \rightarrow 2^{P_i}$$

where P_i is a set of productions of token system T_i, and τ_{1-i} is a set:

$$\cup\, N_i \times \{t_{1-i}\},$$

Table 2.1 An example of table representation of a token system.

S	≡>	a B
B	≡>	b C
C	≡>	c A
A	≡>	a B
C	≡>	c

Table 2.2 Table representation of the interactive system of Examp. 2.3.

Interaction function 0	Productions of token system 0	Interaction function 1	Productions of token system 1
	S ≡> A b B		S ≡> A
(A)	A ≡> a A		A ≡> B
(B)	B ≡> a B		B ≡> A
(d)	A ≡> a		B ≡> d
(d)	B ≡> a		

where N_i is any set of verticies of any token t_i which can be a part of token system T_i and t_{1-i} is any token which can be a part of token system T_{1-i}.

An interaction function determines the set of productions which are permitted to apply, by a vertex and a token.

We shall show an example of an interactive system.

[Example 2.3] Let

$$T_0 = (S, \{S, A, B\}, \{a, b\}, \{S \equiv> A\ b\ B\ ,\ A \equiv> a\ A\ ,$$
$$A \equiv> a\ ,\ B \equiv> a\}),$$

$$T_1 = (S, \{S, A, B\}, \{d\}, \{S \equiv> A\ ,\ A \equiv> B\ ,\ B \equiv> A\ ,\ B \equiv> d\}),$$

$$f_0(n,\ S) = \{\ S \equiv> A\ b\ B\ \},$$

$$f_0(n,\ A) = \{\ A \equiv> a\ A\ \},$$

$$f_0(n,\ B) = \{\ B \equiv> a\ B\ \},$$

$$f_0(n,\ d) = \{\ A \equiv> a\ ,\ B \equiv> a\ \},$$

and

$$f_1(n,\ t) = \{\ S \equiv> A\ ,\ A \equiv> B\ ,\ B \equiv> A\ ,\ B \equiv> d\ \},$$

where n is any natural number and t is any token.

$$S = (\ T_0,\ T_1,\ f_0,\ f_1\ )$$

is an interactive system which is also expressed by the form of

Table 2.2.

We shall next define a move of an interactive system.

Definition 2.5. One move of an interactive system S corresponds to both one move of token system T_0 and next one move of token system T_1.

Definition 2.6. One move of a token system T_i (i=0 or 1) is defined as follows.

A token t_i of token system T_i and a token t_{1-i} of token system T_{1-i} are given. Let n be some vertex of t_i. We have $f_i(n, t_{1-i})$. Then one applicable rule in $f_i(n, t_{1-i})$ is applied to t_i with the center of the rule corresponding to the node n of t_i. Then the next token of token system T_i is obtained.

Let n be any node of t_i, if there is no rule in $f_i(n, t_{1-i})$ which can be applied to the token t_i with the center of the rule corresponding to the node n of the token t_i, then the token t_i is preserved. So the next token of token system T_i is same as the previous one.

We shall explain the concept of "a rule is applied to the token t with the center of the rule corresponding to the node n of t".

The token of Fig. 2.3 (a) is given. Then the rule of Fig. 2.3 (b) is applicable to it with the center of the rule corresponding to the node 1 of the token. But the rule of Fig. 2.3 (b) is not applicable to the token with one corresponding to the node 2 or the node 3 of the token. And the rule of Fig. 2.3 (c) is applicable to the

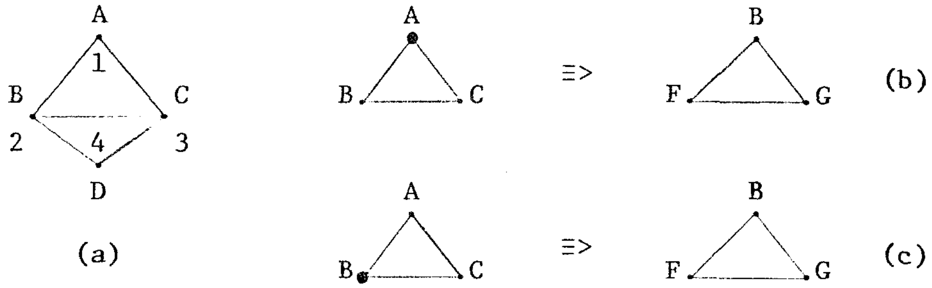


Fig. 2.3 Examples of application of productions with centers.

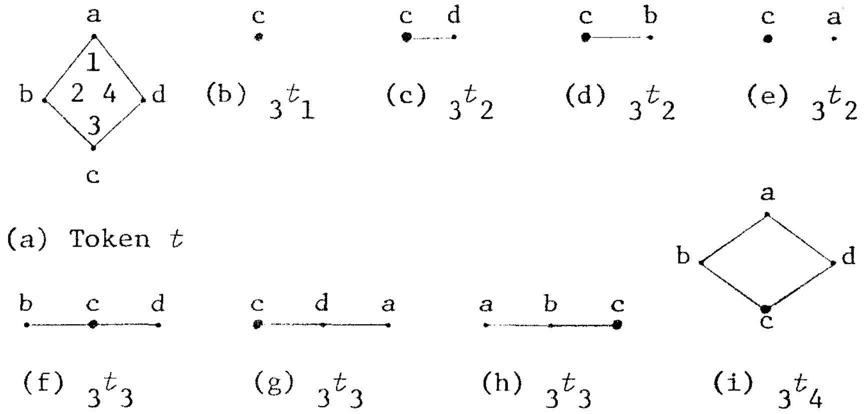


Fig. 2.4 Examples of sub-tokens.

token with the center of it corresponding to the node 2 of the token, but it is not applicable with one corresponding to the node 1 or 3 of the token. The result of the application of the rule of Fig. 2.3 (b) is equal to that of Fig. 2.3 (c).

Next we will explain moves of the interactive system of Table 2.2.

[Example 2.4] In Table 2.3 there is an example of moves of the interactive system of Table 2.2. First there are $\underset{\bullet}{S}$ of T_0 (token system 0) and $\underset{\bullet}{S}$ of T_1 (token system 1). The first move of the interactive system is explained as follows. In this state, we have

$$f_0(n,\ S) = \{\ \underset{\bullet}{S} \equiv> \underline{A\ b\ B}\ \}.$$

Therefore $\underset{\bullet}{S} \equiv> \underline{A\ b\ B}$ is applied to (1) in Table 2.3 and this is the move of token system T_0. Then in token system T_0 (2) in Table 2.3 is obtained. Thus the present token in token system T_0 is (2) in Table 2.3 and the present token in token system T_1 is (1') in Table 2.3. Next the first move of token system T_1 is as follows. Here we have

$$f_1(n,\ \underline{A\ b\ B}) = \{\ \underset{\bullet}{S} \equiv> \underset{\bullet}{A}\ ,\ \ \underset{\bullet}{A} \equiv> \underset{\bullet}{B}\ ,\ \ \underset{\bullet}{B} \equiv> \underset{\bullet}{A}\ ,\ \ \underset{\bullet}{B} \equiv> \underset{\bullet}{d}\ \}.$$

Thus $\underset{\bullet}{S} \equiv> \underset{\bullet}{A}$ is applied to (1') in Table 2.3. So (2') is obtained. Then the present token in token system T_0 is (2) and the present token in token system T_1 is (2').

The next move of the interactive system is as follows. Here we have

Table 2.3 Moves of the interactive system of Table 2.2.

token of token system 0		token of token system 1	
S	(1)	S	(1')
A b B	(2)	A	(2')
A a b B	(3)	B	(3')
A a b a B	(4)	A	(4')
A a a b a B	(5)	B	(5')
A a a b a a B	(6)	d	(6')
a a a b a a B	(7)	d	(7')
a a a b a a a	(8)		

$$f_0(n,\ \underset{\bullet}{\mathrm{A}}) = \{\ \underset{\bullet}{\mathrm{A}} \Rrightarrow \underline{\mathrm{a\ A}}\ \}.$$

Therefore $\underset{\bullet}{\mathrm{A}} \Rrightarrow \underline{\mathrm{a\ A}}$ is applied to (2). Thus (3) is obtained. Then the present token in token system T_0 is (3) and the present token in token system T_1 is (2'). Here $\underset{\bullet}{\mathrm{A}} \Rrightarrow \underset{\bullet}{\mathrm{B}}$ is applied to (2'). Thus the present token in token system T_0 is (3) and the present token in token system T_1 is (3'). Then we have

$$f_0(n,\ \underset{\bullet}{\mathrm{B}}) = \{\ \underset{\bullet}{\mathrm{B}} \Rrightarrow \underline{\mathrm{a\ B}}\ \}.$$

Thus $\underset{\bullet}{\mathrm{B}} \Rrightarrow \underline{\mathrm{a\ B}}$ is applied to (3). Then (4) is obtained. Next $\underset{\bullet}{\mathrm{B}} \Rrightarrow \underset{\bullet}{\mathrm{A}}$ is applied to (3'). Then (4') is obtained. Next $\underset{\bullet}{\mathrm{A}} \Rrightarrow \underline{\mathrm{a\ A}}$ is applied to (4). Then (5) is obtained. Next $\underset{\bullet}{\mathrm{A}} \Rrightarrow \underset{\bullet}{\mathrm{B}}$ is applied to (4'). Then (5') is obtained. Thus the present token in token system T_0 is (5) and the present token in token system T_1 is (5'). Then $\underset{\bullet}{\mathrm{B}} \Rrightarrow \underline{\mathrm{a\ B}}$ is applied to (5). Thus (6) is obtained. Next $\underset{\bullet}{\mathrm{B}} \Rrightarrow \underset{\bullet}{\mathrm{d}}$ is applied to (5'). Then (6') is obtained. Thus the present token in token system T_0 is (6) and the present token in token system T_1 is (6'). Here we have

$$f_0(n,\ \underset{\bullet}{\mathrm{d}}) = \{\ \underset{\bullet}{\mathrm{A}} \Rrightarrow \underset{\bullet}{\mathrm{a}}\ ,\ \ \underset{\bullet}{\mathrm{B}} \Rrightarrow \underset{\bullet}{\mathrm{a}}\ \}.$$

Thus $\underset{\bullet}{\mathrm{A}} \Rrightarrow \underset{\bullet}{\mathrm{a}}$ is applied to (6). Then (7) is obtained. Thus the present token in Token system T_0 is (7) and the present token in token system T_1 is (6'). Here in token system T_1 no rules can be applied to (6'). Therefore (6') is preserved by Definition 2.6. Thus (7') is obtained. Here we have

$$f_0(n, \underset{\cdot}{\mathrm{d}}) = \{\ \underset{\cdot}{\mathrm{A}} \equiv\!> \underset{\cdot}{\mathrm{a}}\ ,\ \ \underset{\cdot}{\mathrm{B}} \equiv\!> \underset{\cdot}{\mathrm{a}}\ \}.$$

Thus $\underset{\cdot}{\mathrm{B}} \equiv\!> \underset{\cdot}{\mathrm{a}}$ is applied to (7). Then (8) is obtained in token system T_0.

Here the token (8) in token system T_0 contains no nonterminal symbols, therefore (8) is an output of the interactive system of Table 2.2 by Definition 2.7.

Definition 2.7. An output of an interactive system is a token of token system T_0 if no nonterminal symbols are contained in it.

Next we will define a context-free token system and a context-sensitive token system.

Definition 2.8. Context-free token system is defined as a token system whose productions satisfy the condition below.

[Condition 2.1] Let a production be

$$t_1 \equiv\!> t_2\ , \tag{2.1}$$

where t_1 is (N_1, ν_1, δ_1) and t_2 is (N_2, ν_2, δ_2). Then $|N_1|$ is one, and N_1 is included by N_2. And if $\nu_1(n) \in V_t$, then $\nu_2(n)=\nu_1(n)$. And if $\delta_1(n)=1$, then $\delta_2(n)=1$.

Definition 2.9. Context-sensitive token system is defined as a token system whose productions satisfy the condition below.

[Condition 2.2] Let a production be (2.1). Then N_1 is included by N_2. And if $\nu_1(n) \in V_t$, then $\nu_2(n)=\nu_1(n)$. And if $\delta_1(n)=1$, then

$\delta_2(n)=1$.

We shall define the idea of *mode A* and *mode B* in 2.4 and in 2.5 respectively.

2.4 Interactive Systems of mode A

We shall define an interactive system of mode A which has correspondence between vertices of a token of token system T_0 and those of token system T_1. We express the width of the pass between token system T_0 and token system T_1 by a suffix i such as mode A_i. In the definition, the width of the pass between token system T_0 and token system T_1 is represented by the number of vertices of *sub-tokens* which are refered by interaction functions. For the definition, we will explain the concept of sub-token ${}_nt_j$ which has n vertices and whose center is j.

When we are given a token of Fig. 2.4 (a), the sub-token ${}_3t_1$ is Fig. 2.4 (b) and the sub-token ${}_3t_2$ is a collection of Figs. 2.4 (c), (d) and (e). And we define that an empty sub-token ${}_nt_0$ is contained in any token.

Definition 2.10. We say that an interactive system which has correspondence between vertices of a token of token system T_0 and those of token system T_1 is of mode A_j. More formally, of mode A_j is an interactive system whose interaction functions: f_0 and f_1, are both have a form of (2.2).

$$f(n,\ t) = \bigcup_{k \leq j} f'({}_nt_k) \qquad (2.2)$$

In (2.2), f' is a function: $\{{}_n t_k\} \rightarrow 2^P$.

In this section we will show

$$CFL \subset ICFL(A_1) \subseteq ICFL(A_2) \subseteq \ldots \subseteq ICFL(A_n)$$

and
$$CSL \subset ICSL(A_1) = ICSL(A_2) = \ldots = ICSL(A_n),$$

where CFL is the family of the languages which are generated by CF (context-free web grammar systems), $ICFL(A_i)$ is the family of the sets of outputs of $ICF(A_i)$ (interactive systems of mode A_i whose token systems are context-free), $ICSL(A_i)$ is the family of the sets of outputs of $ICS(A_i)$ (interactive systems of mode A_i whose token systems are context-sensitive).

We will show one example of $ICF(A_1)$ (interactive systems of mode A_1 whose token systems are context-free).

[Example 2.5] Let

$$T_0 = (\underset{\cdot}{S}\ ,\ \{S, A\}, \{a, b\}, \{\ \underset{\cdot}{S} \equiv> \underset{2\ 1\ 3}{\overline{A\ b\ A}}\ ,\ \ \underset{\cdot}{A} \equiv> \underset{1\ 2}{\overline{a\ A}}\ ,\ \ \underset{\cdot}{A} \equiv> \underset{\cdot}{a}\ \}),$$

$$T_1 = (\underset{\cdot}{S}\ ,\ \{S, B, C, D\}, \{a, d\}, \{\ \underset{\cdot}{S} \equiv> \underset{1\ 2}{\overline{a\ B}}\ ,\ \ \underset{\cdot}{B} \equiv> \underset{1\ 2}{\overline{a\ C}}\ ,$$

$$\underset{\cdot}{C} \equiv> \underset{1\ 2}{\overline{a\ B}}\ ,\ \ \underset{\cdot}{C} \equiv> \underset{1\ 2}{\overline{a\ D}}\ ,\ \ \underset{\cdot}{D} \equiv> \underset{1\ 2}{\overline{a\ d}}\ \}),$$

$$f_0(n,\ t) = \bigcup_{k \leq 1} f'_0({}_n t_k),$$

where
$$f'_0({}_n t_0) = \{\ \underset{\cdot}{S} \equiv> \underset{2\ 1\ 3}{\overline{A\ b\ A}}\ \},$$

$$f'_0(\underset{\cdot}{B}) = \{ \underset{\cdot}{A} \Rrightarrow \underset{1\ 2}{\underline{a\ A}} \},$$

$$f'_0(\underset{\cdot}{C}) = \{ \underset{\cdot}{A} \Rrightarrow \underset{1\ 2}{\underline{a\ A}} \},$$

$$f'_0(\underset{\cdot}{D}) = \{ \underset{\cdot}{A} \Rrightarrow \underset{\cdot}{a} \},$$

and $$f'_0(\underset{\cdot}{d}) = \{ \underset{\cdot}{A} \Rrightarrow \underset{\cdot}{a} \},$$

$$f_1(n,\ t) = \bigcup_{k \le 1} f'_1({}_n t_k),$$

where $$f'_1({}_n t_0) = \{ \underset{\cdot}{S} \Rrightarrow \underset{1\ 2}{\underline{a\ B}},\ \ \underset{\cdot}{B} \Rrightarrow \underset{1\ 2}{\underline{a\ C}},\ \ \underset{\cdot}{C} \Rrightarrow \underset{1\ 2}{\underline{a\ B}},$$
$$\underset{\cdot}{C} \Rrightarrow \underset{1\ 2}{\underline{a\ D}},\ \ \underset{\cdot}{D} \Rrightarrow \underset{1\ 2}{\underline{a\ d}} \}.$$

Then $S = (T_0,\ T_1,\ f_0,\ f_1)$ is $ICF(A_1)$ and which is the interactive system of Table 2.4.

Next we show an example of moves of the interactive system of Table 2.4. In Table 2.5 there are moves of the interactive system.

First there are $\underset{\cdot}{S}$ in token system T_0 and $\underset{\cdot}{S}$ in token system T_1. In this state, we have

$$f_1(1,\ \underset{\cdot}{S}) = \{ \underset{\cdot}{S} \Rrightarrow \underset{2\ 1\ 3}{\underline{A\ b\ A}} \},$$

therefore this production is applied to (1) in Table 2.5 and this is the first move of token system T_0. Then in token system T_0, (2) in Table 2.5 is obtained. Thus the present token in token system T_0 is

Table 2.4 Table representation of the interactive system of Examp. 2.5.

Interaction function 0	Productions of token system 0	Interaction function 1	Productions of token system 1
	S ≡> A b A 2 1 3		S ≡> a B 1 2
(B)	A ≡> a A 1 2	(a)	B ≡> a C 1 2
(C)	A ≡> a A 1 2	(a)	C ≡> a B 1 2
(D)	A ≡> a	(a)	C ≡> a D 1 2
(d)	A ≡> a	(a)	D ≡> a d 1 2

Table 2.5 Moves of the interactive system of Table 2.4.

Tokens of token system 0		Tokens of token system 1	
S 1	(1)	S 1	(1')
A b A 2 1 3	(2)	a B 1 2	(2')
A a b A 4 2 1 3	(3)	a a C 1 2 3	(3')
A a b a A 4 2 1 3 5	(4)	a a a D 1 2 3 4	(4')
a a b a A 4 2 1 3 5	(5)	a a a a d 1 2 3 4 5	(5')
a a b a a 4 2 1 3 5	(6)		

(2) in Table 2.5 and the present token in token system T_1 is (1') in Table 2.5.

Next the first move of token system T_1 is as follows. Here we have

$$f_1(1, \underset{2\ 1\ 3}{\underline{A\ b\ A}}) = \{ \underset{\cdot}{S} \equiv> \underset{1\ 2}{\underline{a\ B}} ,\quad \underset{\cdot}{B} \equiv> \underset{1\ 2}{\underline{a\ C}} ,\quad \underset{\cdot}{C} \equiv> \underset{1\ 2}{\underline{a\ B}} ,$$

$$\underset{\cdot}{C} \equiv> \underset{1\ 2}{\underline{a\ D}} ,\quad \underset{\cdot}{D} \equiv> \underset{1\ 2}{\underline{a\ d}} \}.$$

Thus $\underset{\cdot}{S} \equiv> \underset{1\ 2}{\underline{a\ B}}$ is applied to (1') in Table 2.5. So (2') is obtained.

Then the present token in token system T_0 is (2) in Table 2.5 and the present token in token system T_1 is (2') in Table 2.5.

The next move of the interactive system is as follows. Here we have

$$f_0(1, \underset{1\ 2}{\underline{a\ B}}) = \{ \underset{\cdot}{S} \equiv> \underset{2\ 1\ 3}{\underline{A\ b\ A}} \},$$

$$f_0(2, \underset{1\ 2}{\underline{a\ B}}) = \{ \underset{\cdot}{S} \equiv> \underset{2\ 1\ 3}{\underline{A\ b\ A}} ,\quad \underset{\cdot}{A} \equiv> \underset{1\ 2}{\underline{a\ A}} \},$$

$$f_0(3, \underset{1\ 2}{\underline{a\ B}}) = \{ \underset{\cdot}{S} \equiv> \underset{2\ 1\ 3}{\underline{A\ b\ A}} \},$$

therefore $\underset{\cdot}{A} \equiv> \underline{a\ A}$ is applied to (2) in Table 2.5. Thus in token system T_0 (3) is obtained. Then the present token in token system T_0 is (3) in Table 2.5 and the present token in token system T_1 is (2') in Table 2.5. In this state, we have

$$f_1(1, \underset{4\;2\;1\;3}{\underline{A\ a\ b\ A}}) = \{\ \underset{\cdot}{S} \Rrightarrow \underset{1\;2}{\underline{a\ B}}\ ,\ \ \underset{\cdot}{B} \Rrightarrow \underset{1\;2}{\underline{a\ C}}\ ,$$

$$\underset{\cdot}{C} \Rrightarrow \underset{1\;2}{\underline{a\ B}}\ ,\ \ \underset{\cdot}{C} \Rrightarrow \underset{1\;2}{\underline{a\ D}}\ ,\ \ \underset{\cdot}{D} \Rrightarrow \underset{1\;2}{\underline{a\ d}}\ \},$$

and $$f_1(2, \underset{4\;2\;1\;3}{\underline{A\ a\ b\ A}}) = f_1(3, \underset{4\;2\;1\;3}{\underline{A\ a\ b\ A}}) = f_1(4, \underset{4\;2\;1\;3}{\underline{A\ a\ b\ A}}).$$

Thus $\underset{\cdot}{B} \Rrightarrow \underline{a\ C}$ is applied to (2') in Table 2.5. Then (3') is obtained in token system T_1. Thus the present token in token system T_0 is (3) and the present token in token system T_1 is (3'). Here we have

$$f_0(1, \underset{1\;2\;3}{\underline{a\ a\ C}}) = \{\ \underset{\cdot}{S} \Rrightarrow \underset{2\;1\;3}{\underline{A\ b\ A}}\ \},$$

$$f_0(3, \underset{1\;2\;3}{\underline{a\ a\ C}}) = \{\ \underset{\cdot}{S} \Rrightarrow \underset{2\;1\;3}{\underline{A\ b\ A}}\ ,\ \ \underset{\cdot}{A} \Rrightarrow \underset{1\;2}{\underline{a\ A}}\ \},$$

and $$f_0(2, \underset{1\;2\;3}{\underline{a\ a\ C}}) = f_0(4, \underset{1\;2\;3}{\underline{a\ a\ C}}) = f_0(1, \underset{1\;2\;3}{\underline{a\ a\ C}}).$$

Therefore $\underset{\cdot}{A} \Rrightarrow \underline{a\ A}$ is applied to (3). Then in token system T_0, (4) is obtained. Thus the present token in token system T_0 is (4) and the present token in token system T_1 is (3'). Here we have

$$f_1(1, \underset{4\;2\;1\;3\;5}{\underline{A\ a\ b\ a\ A}}) = f_1(2, \underset{4\;2\;1\;3\;5}{\underline{A\ a\ b\ a\ A}}) = f_1(3, \underset{4\;2\;1\;3\;5}{\underline{A\ a\ b\ a\ A}})$$

$$= \{\ \underset{\cdot}{S} \Rrightarrow \underset{1\;2}{\underline{a\ B}}\ ,\ \ \underset{\cdot}{B} \Rrightarrow \underset{1\;2}{\underline{a\ C}}\ ,\ \ \underset{\cdot}{C} \Rrightarrow \underset{1\;2}{\underline{a\ B}}\ ,\ \ \underset{\cdot}{C} \Rrightarrow \underset{1\;2}{\underline{a\ D}}\ ,\ \ \underset{\cdot}{D} \Rrightarrow \underset{1\;2}{\underline{a\ d}}\ \}.$$

Therefore $\underset{\cdot}{C} \Rrightarrow \underline{a\ D}$ is applied to (3'). Then in token system T_1, (4') is obtained. Thus the present token in token system T_0 is (4) and the present token in token system T_1 is (4'). Here we have

$$f_0(1, \underset{1\ 2\ 3\ 4}{\underline{a\ a\ a\ D}}) = f_0(2, \underset{1\ 2\ 3\ 4}{\underline{a\ a\ a\ D}}) = f_0(3, \underset{1\ 2\ 3\ 4}{\underline{a\ a\ a\ D}})$$

$$= \{ \underset{\bullet}{S} \Rrightarrow \underset{2\ 1\ 3}{\underline{A\ b\ A}} \},$$

and

$$f_0(4, \underset{1\ 2\ 3\ 4}{\underline{a\ a\ a\ D}}) = \{ \underset{\bullet}{S} \Rrightarrow \underset{2\ 1\ 3}{\underline{A\ b\ A}} ,\ \ \underset{\bullet}{A} \Rrightarrow \underset{\bullet}{a} \}.$$

Thus $\underset{\bullet}{A} \Rrightarrow \underset{\bullet}{a}$ is applied to (4). Then in token system T_0, (5) is obtained. Thus the present token in token system T_0 is (5) and the present token in token system T_1 is (4'). Next $\underset{\bullet}{D} \Rrightarrow \underset{\bullet}{d}$ is applied to (4'). Then in token system T_1, (5') is obtained. Thus the present token in token system T_0 is (5) and the present token in token system T_1 is (5'). Here we have

$$f_0(1, \underset{1\ 2\ 3\ 4\ 5}{\underline{a\ a\ a\ a\ d}}) = f_0(2, \underset{1\ 2\ 3\ 4\ 5}{\underline{a\ a\ a\ a\ d}}) = f_0(3, \underset{1\ 2\ 3\ 4\ 5}{\underline{a\ a\ a\ a\ d}})$$

$$= f_0(4, \underset{1\ 2\ 3\ 4\ 5}{\underline{a\ a\ a\ a\ d}}) = \{ \underset{\bullet}{S} \Rrightarrow \underset{2\ 1\ 3}{\underline{A\ b\ A}} \},$$

and

$$f_0(5, \underset{1\ 2\ 3\ 4\ 5}{\underline{a\ a\ a\ a\ d}}) = \{ \underset{\bullet}{A} \Rrightarrow \underset{\bullet}{a} \}.$$

Thus $\underset{\bullet}{A} \Rrightarrow \underset{\bullet}{a}$ is applied to (5). Then in token system T_0, (6) is obtained.

Here a token (6) in token system T_0 contains no nonterminal symbols, therefore (6) is an output of the interactive system of Table 2.4 by Definition 2.7.

Theorem 2.1.

$$CFL \subset ICFL(A_1) \subseteq ICFL(A_2) \subseteq \dots \subseteq ICFL(A_n)$$

Proof. We shall show that $CFL \subset ICFL(A_1)$. For any context-free web grammar system:

$$CF = (s\ ,\ V_n\ ,\ V_t\ ,\ P),$$

let $$T_0 = (s\ ,\ V_n\ ,\ V_t\ ,\ P)$$

and $$f_0(n, t) = P,$$

where n is any natural number and t is any token, and T_1 is any context-free token system and f_1 is any interaction function of mode A_1. Then

$$S = (\ T_0, T_1, f_0, f_1\ )$$

is $ICF(A_1)$ (an interactive system of mode A_1 whose token systems are context-free) and simulates the CF from Definitions 2.1, 2.2, 2.8 and 2.10. Therefore $CFL \subseteq ICFL(A_1)$ is proved.

From Definition 2.10 $ICF(A_i)$ is also $ICF(A_{i+1})$, therefore $ICFL(A_i) \subseteq ICFL(A_{i+1})$ is apparent.

Next we shall show

$$CFL \neq ICFL(A_1).$$

In [1], it was shown that only the graphs as in Fig. 2.5 can not be generated by any CF (context-free web grammar system), but the interactive system of Table 2.4 generates only the graphs as in Fig. 2.5,

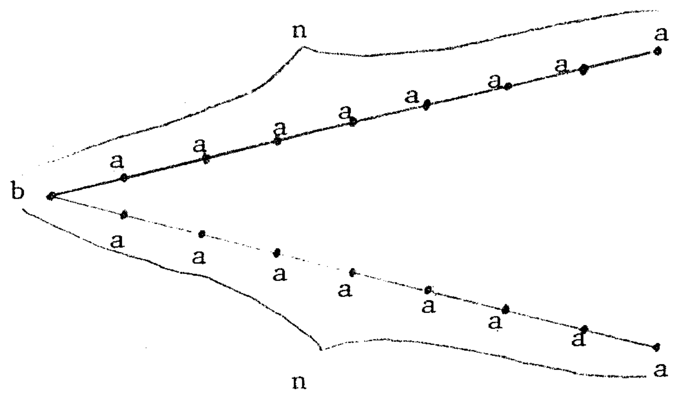


Fig. 2.5 An example of the graph in the set which can not be included in *CFL*.

and which is of mode A_1 and whose token systems are context-free. Therefore $CFL \neq ICFL(A_1)$ is apparent. Thus $CFL \subset ICFL(A_1)$ is proved.

Theorem 2.2

$$CSL \subset ICSL(A_1) = ICSL(A_2) = \dots = ICSL(A_n).$$

Proof. We shall show that $CSL \subset ICSL(A_1)$. From Definitions 2.1, 2.2, 2.9 and 2.10, $CSL \subseteq ICSL(A_1)$ is apparent. Next we show that $CSL \neq ICSL(A_1)$. The set of all separable graphs is not included by CSL from [1]. The interactive system of Table 2.6 generates all separable graphs, and it is $ICS(A_1)$ (an interactive system of mode A_1 whose token systems are context-sensitive). Therefore

$$CSL \subset ICSL(A_1)$$

is proved. We show an example of moves of the interactive system of Table 2.6 in Table 2.7.

Next we shall show $ICSL(A_i) = ICSL(A_{i+1})$. From Definition 2.10 $ICSL(A_i) \subseteq ICSL(A_{i+1})$ is apparent. Then we show

$$ICSL(A_i) \supset ICSL(A_{i+1}).$$

In order to prove this formula we show that $ICS(A_1)$ (an interactive system of mode A_1 whose token systems are context-sensitive) can simulate $ICS(A_i)$ (an interactive system of mode A_i whose token systems are context-sensitive). $ICS(A_i)$ is expressed by two sets of the rules of the form as (2.3) as shown in Table 2.6.

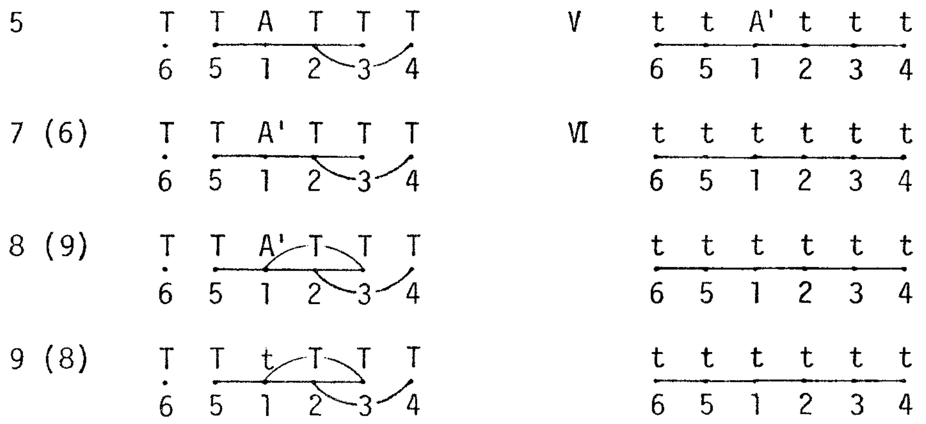
Table 2.6 The interactive system which generates all separable graphs.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
	S(1) ≡> A"(1) — B'(2)	(1)		S(1) ≡> A(1) — B'(2)	(I)
(B')	B'(1) ≡> B(1) B'(2)	(2)	(B)	B'(1) ≡> B(1) — B'(2)	(II)
(B')	B'(1) B(2) ≡> B'(1) — B(2)	(3)	(T)	B(2) — B'(1) ≡> t(2) — t(1)	(III)
(B')	B'(1) ≡> T(1)	(4)	(T)	t(1) — B(2) ≡> t(1) — t(2)	(IV)
(t)	B(1) ≡> T(1)	(5)	(T)	t(1) — A(2) ≡> t(1) — A'(2)	(V)
(A')	A(1) ≡> A(1) B'(2)	(6)	(A')	A'(1) ≡> t(1)	(VI)
(A')	A(1) ≡> A'(1)	(7)	(A)	A'(1) ≡> A(1) — B'(2)	(VII)
(t)	A(1) T(2) ≡> A(1) — T(2)	(8)	(T)	B'(1) — A(2) ≡> t(1) — A'(2)	(VIII)
(t)	A'(1) ≡> t(1)	(9)			
(A')	A"(1) ≡> A(1) — B'(2)	(10)			
	t(1) T(2) ≡> t(1) t(2)	(11)			

Table 2.7 Moves of the interactive system of Table 2.6.

Applied production	Tokens of token system 0	Applied production	Tokens of token system 1
	S 1		S 1
1	A" B' 1 2	I	A B' 1 2
2 (4)	A" B B' 1 2 3	II	A B B' 1 2 3
3 (2, 4)	A" B B' 1 2 3		A B B' 1 2 3
2 (3, 4)	A" B B B' 1 2 3 4	II	A B B B' 1 2 3 4
3 (2, 4)	A" B B B' 1 2 3 4		A B B B' 1 2 3 4
4 (2, 3)	A" B B T 1 2 3 4	III	A B t t 1 2 3 4
5	A" B T T 1 2 3 4	IV	A t t t 1 2 3 4
5	A" T T T 1 2 3 4	V	A' t t t 1 2 3 4
10	B' A T T T 5 1 2 3 4	VII	B' A t t t 5 1 2 3 4
2 (4)	B' B A T T T 6 5 1 2 3 4	II	B' B A t t t 6 5 1 2 3 4
4 (2, 3)	T B A T T T 6 5 1 2 3 4	III	t t A t t t 6 5 1 2 3 4

continue


5 T T A T T T
6 5 1 2 3 4
V t t A' t t t
6 5 1 2 3 4
7 (6) T T A' T T T
6 5 1 2 3 4
VI t t t t t t
6 5 1 2 3 4
8 (9) T T A' T T T
6 5 1 2 3 4
t t t t t t
6 5 1 2 3 4
9 (8) T T t T T T
6 5 1 2 3 4
t t t t t t
6 5 1 2 3 4

$$(t_1) \qquad t_2 \Rrightarrow t_3 \tag{2.3}$$

$$(y^+) \qquad t_2^\times \Rrightarrow t_3^\circ \tag{2.4}$$

$$(y^+) \qquad t_2^\times \Rrightarrow t_3 \tag{2.5}$$

$$(x^\circ) \qquad y^+ \Rrightarrow y^\times \tag{2.6}$$

$$(x^\times) \qquad t_1^\circ \Rrightarrow t_1^+ \tag{2.7}$$

Basically the rule of (2.3) of $ICS(A_i)$ is simulated by the rules of (2.4)-(2.7) of $ICS(A_1)$. If the rule of (2.3) belongs to token system T_0 of the $ICS(A_i)$, then the rules of (2.4) and (2.5) belong to token system T_0 of the $ICS(A_1)$, and the rules of (2.6) and (2.7) belong to token system T_1, and vice versa. The rules of (2.7) fill the role of checking whether sub-token t_1 in (2.3) is contained in the token of the token system which contains the rule of (2.7). In the rule of (2.7), t_1° is constructed with only t_1 of (2.3) and x° (any one vertex token whose label is superfixed with '$\circ$'). t_1 may contain x°. For any t_1° there is the rule of (2.7) which contains the t_1°. And t_1^+ in (2.7) is the sub-token which is obtained by changing x° in t_1° in (2.7) to x and attaching the superfix: '$+$' to the center's label of t_1 in t_1°. If the rule of (2.7) is applied, then the rules of (2.4) and (2.5) are tried to apply when they know that they are allowed to apply by seeing the vertex whose label is changed to a label which is superfixed with '$+$' by the rule of (2.7). In the rules of (2.4) $t_2^\times$ is any sub-token which is constructed with only t_2 in (2.3) and $x^\times$ (any one vertex token whose label is superfixed with '×'). And t_2 may cintain $\overset{\times}{x}$. For any $\overset{\times}{t_2}$ there is the rule of (2.4) which contains the $\overset{\times}{t_2}$. And $\overset{\circ}{t_3}$ in the rule of (2.4) is the sub-token which is obtained by changing $\overset{\times}{x}$ in $\overset{\times}{t_2}$ to x and t_2 in $\overset{\times}{t_2}$ to $\overset{\circ}{t_3}$ which is obtained by attaching the superfix '∘' to the center's label of t_3 in the rule of (2.3). The rule of (2.6) fills the role of handing over a move to the other token system, when it is informed of applying of the rule of (2.4) by the vertex whose label is superfixed by '∘'. The rule of (2.5) is same as the rule of (2.4) except $\overset{\circ}{t_3}$. $\overset{01}{t_3}$ in the rule of (2.5) is obtained by changing the superfix: '∘' of t_3 in the rule of (2.4) to '01'. The rule of (2.5) has the role of requiring to the other token system to perform an empty move (preservation of the token) which is defined in Definition 2.6. We shall later show an example of changing the rule of (2.2) of $ICS(A_3)$ to the rules of (2.4) and (2.7).

It is necessary to confirm that no productions can be applied when an empty move is required by the rule of (2.5). To confirm that no productions can be applied, the rules in Table 2.8 or Table 2.9 are necessary. If it is confirmed that no productions can be applied to the token which is in token system T_0 by the rules in Table 2.8 or Table 2.9, then a move is handed over to token system T_1, and vice versa. We shall later explain Table 2.8 and Table 2.9.

The interactive system which is constructed with the rules of (2.4)-(2.7) and in Tables 2.8 and 2.9 is of mode A_1. And it simulates the move of the original interactive system. Therefore $ICS(A_i)$ can be simulated by $ICS(A_1)$. Then we have

$$ICSL(A_1) \supseteq ICSL(A_i).$$

Table 2.8 Rules for confirming that no productions can be applied.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
(y^{01})	$x^{01}\ t_1 \equiv> x\ t_1^{11}$	(A)	(x^{01})	$y^2 \equiv> y^{01}$	(a)
	$x^{11}\ x \equiv> x^{12}\ x^{21}$	(B)	(x^{11})	$t_2 \equiv> t_2^3$	(b)
(x^{21})	$x^{11}\ t_1 \equiv> x^4\ t_1^{11}$	(C)	(x^{21})	$x\ y^{01} \equiv> x^{21}\ y$	(c)
(x^{01})	$x^4\ t_1^{11} \equiv> x\ t_1^{11}$	(D)	(x^4)	$x^{21} \equiv> x^{01}$	(d)
(x^{11})	$x^{01} \equiv> x^{51}$	(E)	(x^{21})	$x^{21} \equiv> x^{11}$	(e)
(x^{11})	$x^{21} \equiv> x^{51}$	(F)	(x^{01})	$x^{01} \equiv> x^{11}$	(f)
(x^3)	$x^{51} \equiv> x^3$	(G)	(x^{51})	$x^{11}\ x^3 \equiv> x^3\ x^3$	(g)
(x^{11})	$x^{51} \equiv> x^{71}$	(H)	(x^{71})	$x^{11} \equiv> x^{81}$	(h)
(x^{81})	$x^{71}\ x^{12} \equiv> x^{62}\ x$	(I)	(x^{72})	$x^{81} \equiv> x^{12}$	(i)
(x^{22})	$x^{71} \equiv> x^{02}$	(J)	(x^{71})	$x^{81} \equiv> x^{22}$	(j)
(x^{12})	$x^{52}\ x^{12} \equiv> x^{53}\ x$	(K)	(x^{53})	$x^{12} \equiv> x^{13}$	(k)
(x^{13})	$x^{53}\ x^{12} \equiv> x^{52}\ x$	(L)	(x^{52})	$x^{12} \equiv> x^{22}$	(l)
(x^{22})	$x^{53} \equiv> x^{02}$	(M)	(x^{53})	$x^{13} \equiv> x^{22}$	(m)
(x^{22})	$x^{52} \equiv> x^{02}$	(N)	(x^{52})	$x^{13} \equiv> x^{12}$	(n)
			(x^{02})	$x^{22} \equiv> x^{02}$	(o)

Table 2.9 Rule for confirming that no productions can be applied.

Interaction function 0	Productions of token system 0	Interaction function 1	Productions of token system 1
(y^{3})	$x^{0\,1} \equiv> x^{3}$	$(x^{0\,1})$	$y^{2}\ t_{2} \equiv> y^{3}\ t_{2}$
(y^{2})	$x^{0\,1} \equiv> x^{0\,2}$		

Therefore $$ICSL(A_1) = ICSL(A_i)$$

is apparent. So

$$ICSL(A_i) = ICSL(A_{i+1})$$

is proved.

We show an example of changing the rule of (2.3) of $ICS(A_3)$ to the rules of (2.4) and (2.7) of $ICS(A_1)$.

[Example 2.6] If the rule of (2.3) of $ICS(A_3)$ is Fig. 2.6 (1), then the rules of (2.4) of $ICS(A_1)$ are Figs. 2.6 (2)-(12) and the rules of (2.7) are Figs. 2.6 (13)-(23).

Then we shall explain the rules in Tables 2.8 and 2.9. In Table 2.8, x^α and y^α expresses one vertex token whose label is superfixed with 'α', and $x^\alpha \ t_i^\beta$ does the sub-token which is constructed with only one vertex token x^α and t_i^β which is obtained by superfixing 'β' to the label of the center of the sub-token t_i in the rule of (2.3) and superfixing any superfix to labels of other vertices. t_i^β may contain x^α. And their center is the vertex of x^α. $x^\alpha \ x^\beta$ expresses the sub-tokens $\underset{\bullet}{x^\alpha} \ \underset{\bullet}{x^\beta}$ and $\underrightarrow{x^\alpha \quad x^\beta}$.

Next we explain moves of the rules in Table 2.8. First, (a) in Table 2.8 is notified by the vertex whose label is superfixed with '01' that an empty move is required. Next, (A) in Table 2.8 is tried to apply. If the token of token system T_0 contains the sub-token t_1 in the rule of (2.3), then (A) in Table 2.8 is applied. So (b) in Table 2.8 is notified that t_1 is contained in the token. Then whether

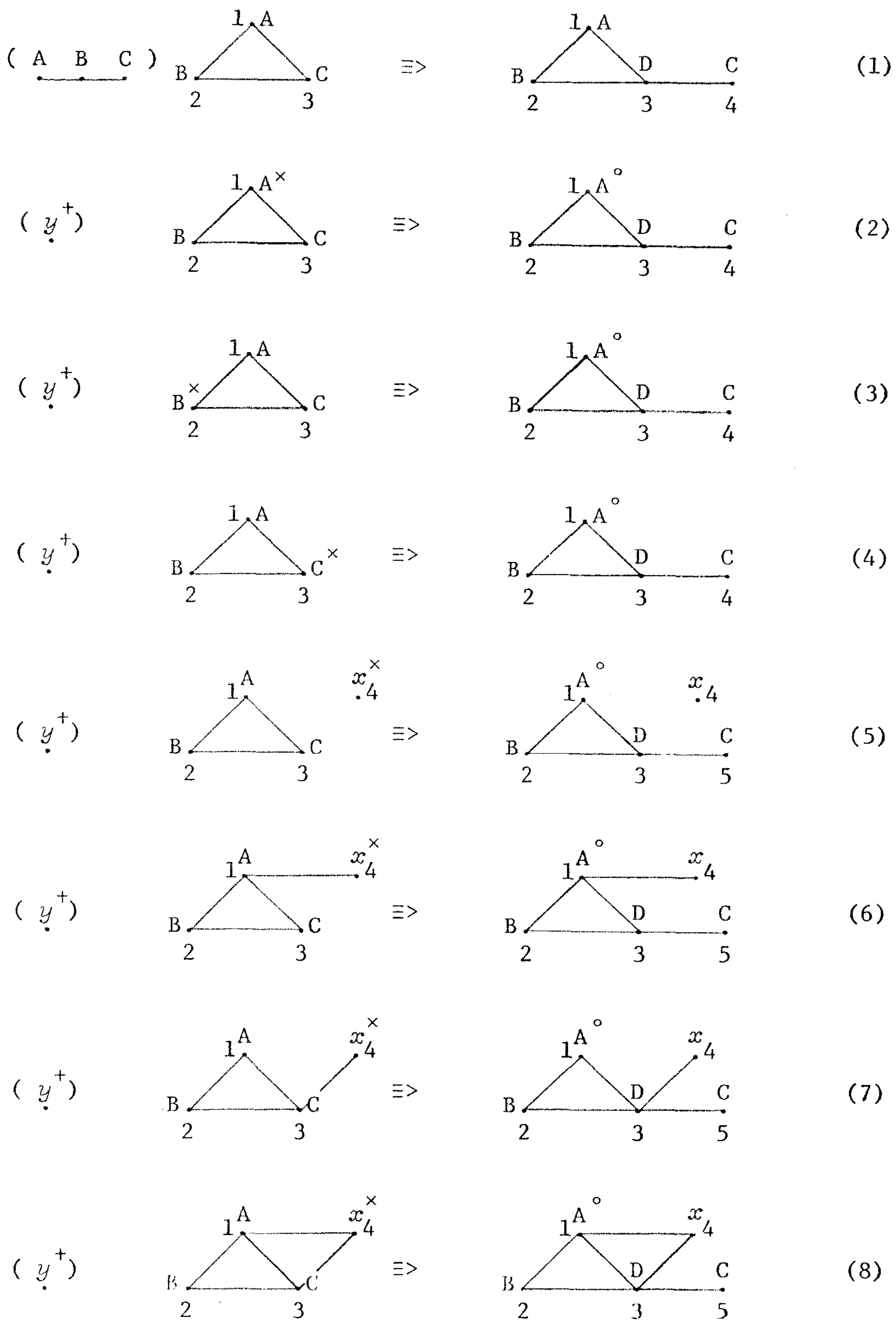


Fig. 2.6 Examples of productions transformed from $ICS(A_3)$.(partial)

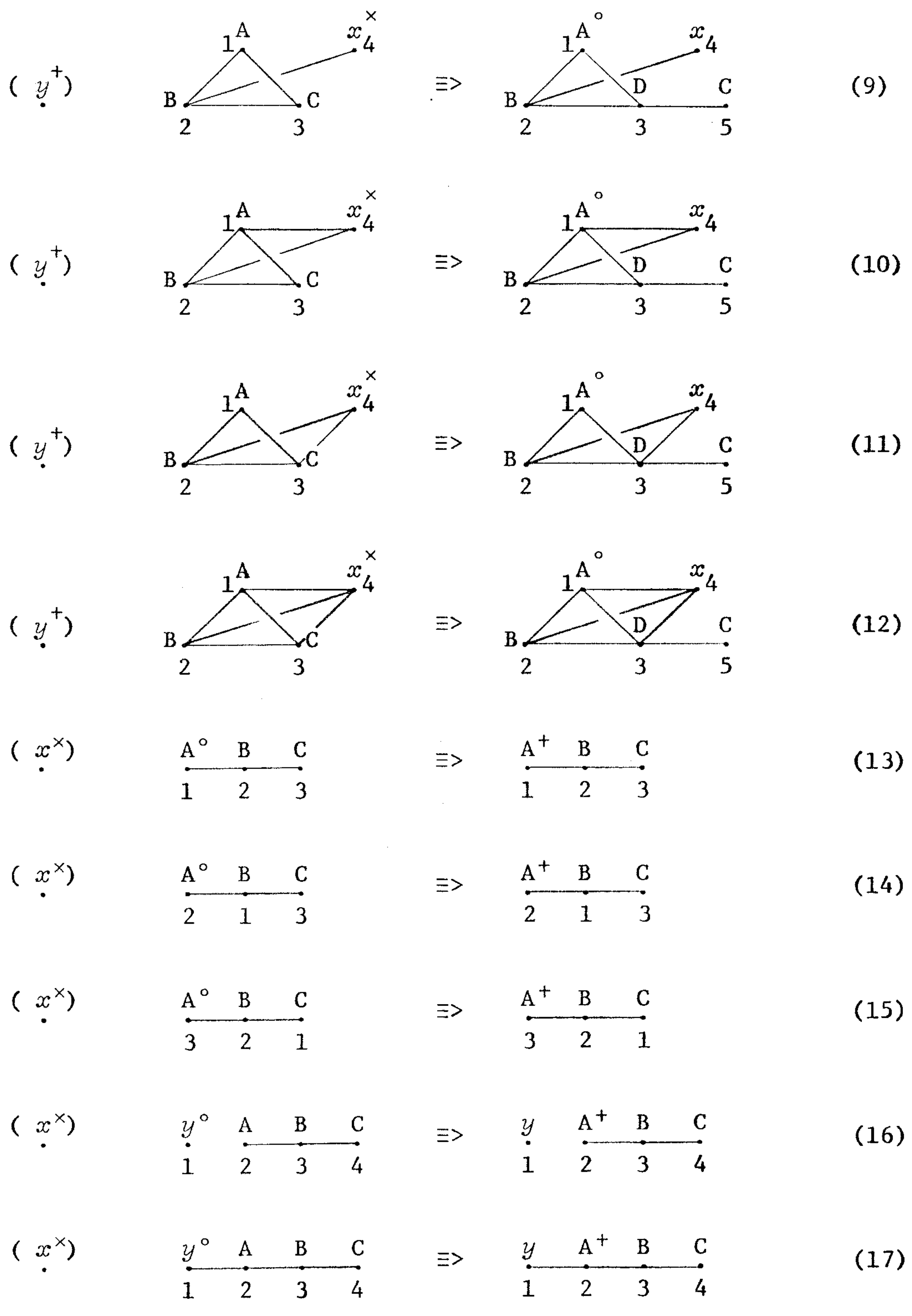


Fig. 2.6 Continued.

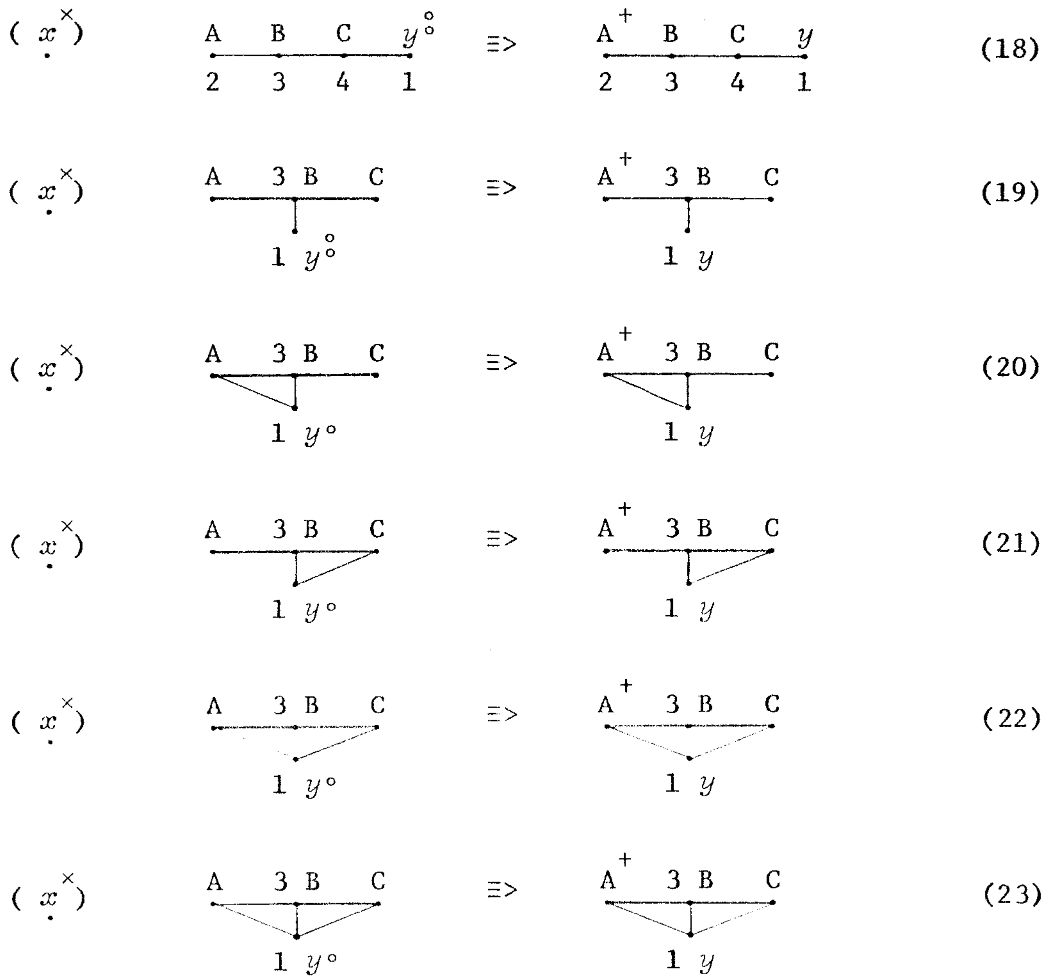


Fig. 2.6 Continued.

the sub-token t_2 in the rule of (2.3) whose center corresponds to x^{11} exists or not is checked. If it exists, then its center's label is changed to the label which is nonterminal symbol which can not be rewritten to termanal symbols. So later by (G) in Table 2.8 the same vertex is generated in token system T_0. Therefore the interactive system never generate an output. If the sub-token t_2 in the rule of (2.3) whose center corresponds to x^{11} does not exist, then there is no rules which can be applied. Therefore in token system T_1 the token is preserved. So a move is handed over token system T_0 and (B) in Table 2.8 is applied. Then (c) in Table 2.8 is applied. Next (C) in Table 2.8 is tried to apply. If t_1 in the rule of (2.3) exists in the token of token system T_0, then (C) in Table 2.8 is applied. So (d) in Table 2.8 is applied. Then (D) is applied. The state when (D) has been applied is the same as the state at the time when (A) was applied, except x^{11}. If (C) can not be applied, then (e) is applied. Then (F) is applied. If x^3 which notifies that an empty move is not allowed, exists, then (g) is applied. So (G) is applied. If x^3 does not exist, then (H) is applied. Then (I)-(N) and (h)-(n) do preparations of a next check.

For each rule of (2.3) of $ICS(A_i)$ the rules in Table 2.8 are made. In table 2.8, (N) notifys the next check that this check ends, by x^{02}. (o) corresponds to (a) of a next check.

If in the rule of (2.3) t_1 is an empty sub-token, then the rules in Table 2.9 are used to confirm that no productions can be applied. Then we will explain Table 2.9.

If t_2 in the rule of (2.3) of $ICS(A_i)$ exists in the token of token system T_1, then (a) in Table 2.9 is applied. So (A) in Table 2.9 is applied. Therefore if t_2 in the rule of (2.3) is contained in the token of token system T_1, then the nonterminal symbol which can not be rewritten is generated. So the interactive system can not generate an output. If t_2 in the rule of (2.3) is not contained in the token of token system T_1, then (B) in Table 2.9 is applied. So a next check starts.

We will show an example of generating $ICS(\mathrm{A}_1)$ which simulates $ICS(\mathrm{A}_{24})$.

[Example 2.7] We show $ICS(\mathrm{A}_{24})$ in Table 2.10. It models a crossing where signals and cars interact one another. In the $ICS(\mathrm{A}_{24})$, token system T_0 models a signal system, and token system T_1 models cars. By the number of vertices whose label is 'A', the number of cars are expressed. In token system T_0 of the $ICS(\mathrm{A}_{24})$ we express by the label 'G', the signal is green, and by the label 'Y', the signal is yellow, and so on. In Table 2.11 we express parts of $ICS(\mathrm{A}_1)$ which simulates the $ICS(\mathrm{A}_{24})$ of Table 2.10. In Table 2.12 moves of the $ICS(\mathrm{A}_{24})$ are shown, and in Table 2.13 moves of the $ICS(\mathrm{A}_1)$ of Table 2.11 are shown.

Proposition 2.1

$$ICFL(\mathrm{A}_i) \not\supset CSL$$

Proof. Ezawa [14] shows that CSL includes the set of all complete graphs. In any $ICF(\mathrm{A}_i)$ (an interactive system of mode A_i whose token systems are context-free) token system T_0 is a context-free web grammar system, therefore it can not generates all complete graphs.

Table 2.10 An example of $ICS(A_{24})$.

Interaction function 0	Productions of token system 0	Interaction function 1	Productions of token system 1

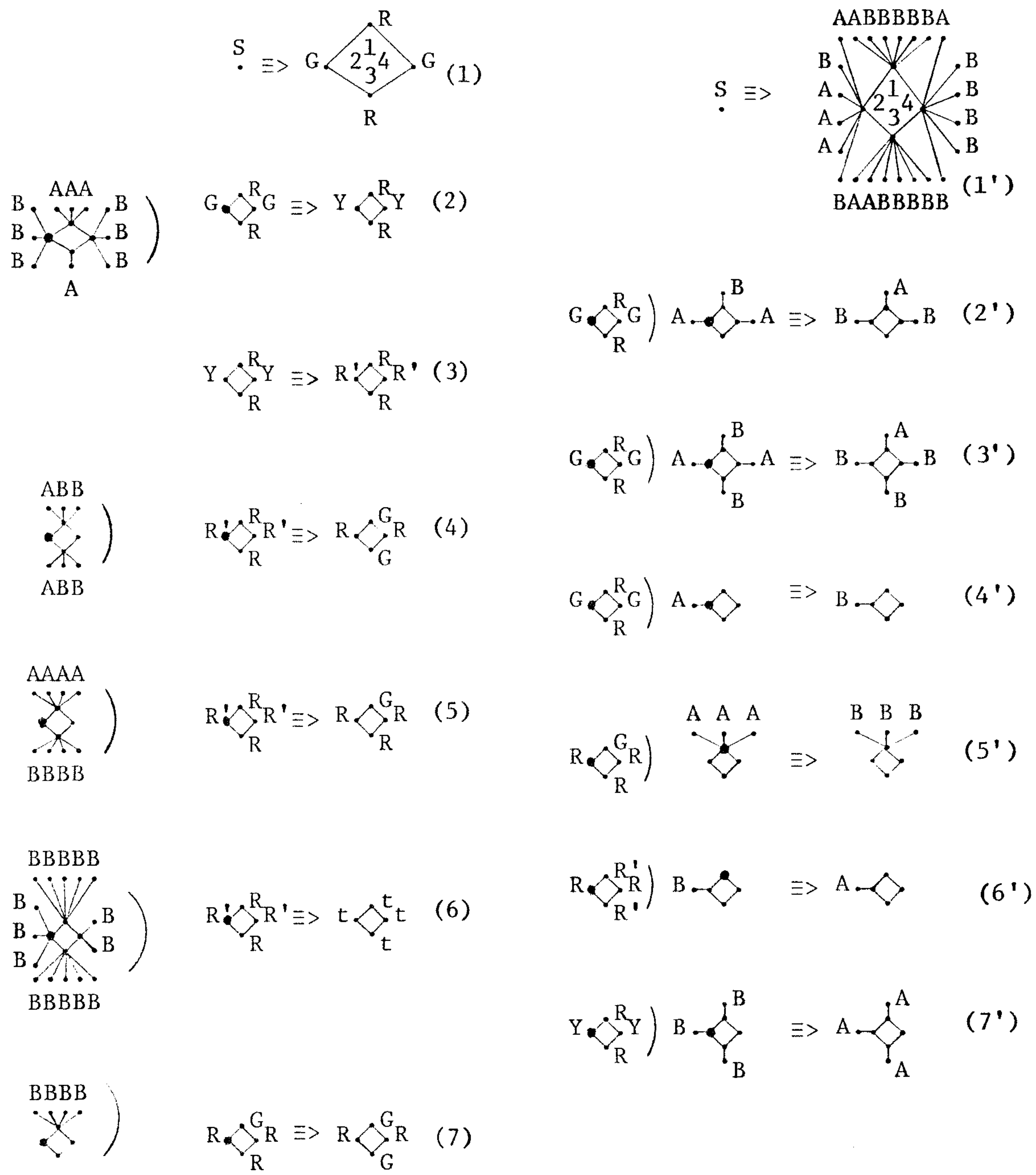


Table 2.11 The *ICS*(A_1) which simulates the *ICS*(A_{24}) of Table 2.10.

Interaction function 0	Productions of token system 0	Interaction function 1	Productions of token system 1

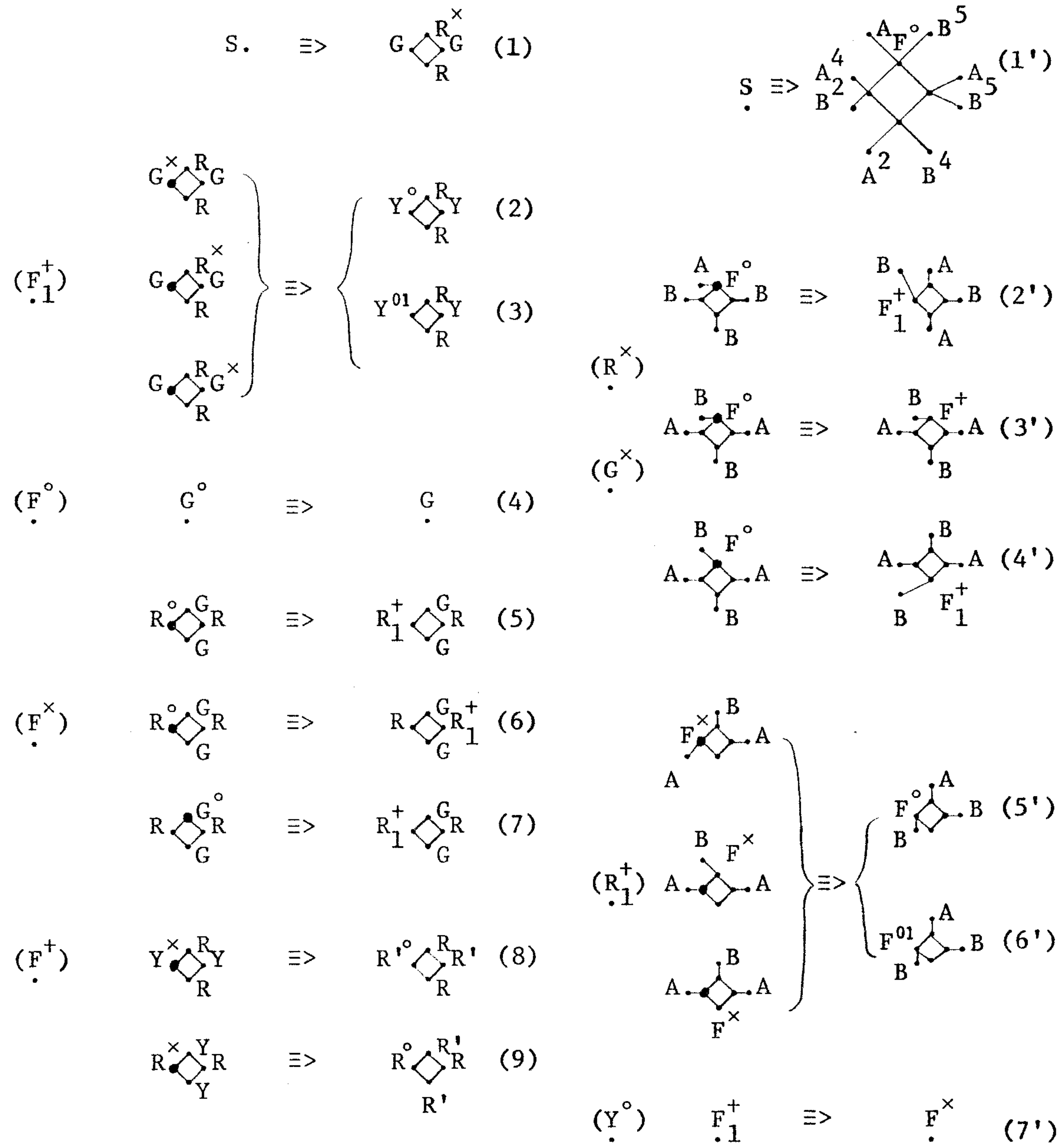
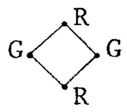
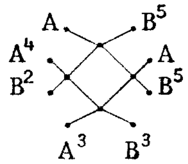
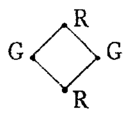
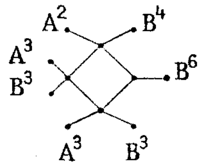
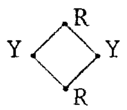
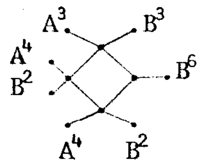
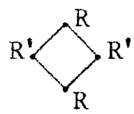
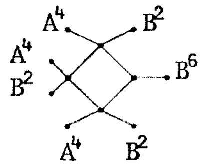
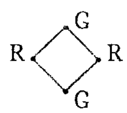
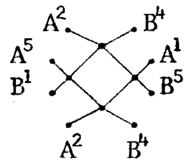
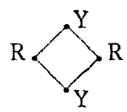
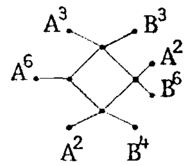


Table 2.12 Moves of the interactive system of Table 2.10.

Applied production	Token of token system 0	Applied production	Token of token system 1
	S		S
1		1'	
		3'	
2		7'	
3		2'	
4		2'	
2		7'	

continue

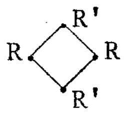
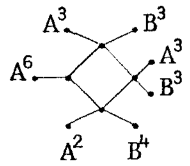
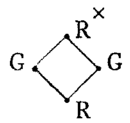
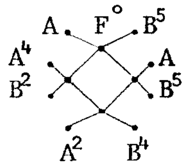
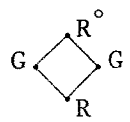
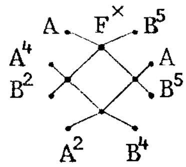
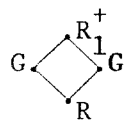
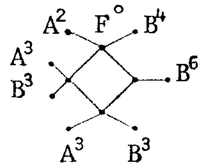
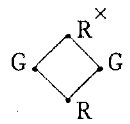
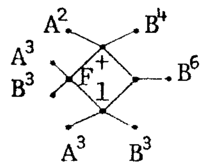
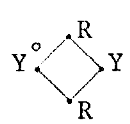
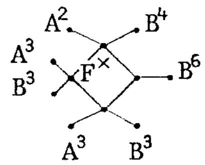
3		6'	

Table 2.13 Moves of the interactive system of Table 2.11.

Applied production	Token of token system 0	Applied production	Token of token system 1
1			
			
5		5'	
		4	
2		7'	

So $ICFL(A_i) \neq CSL$

is apparent.

We show a relation between $ICFL(A_i)$ and $ICSL(A_i)$.

Proposition 2.2.

$$ICSL(A_1) \supset ICFL(A_i)$$

Proof. From Definitions 2.8, 2.9 and 2.10 $ICSL(A_i) \supseteq ICFL(A_i)$ is apparent. From Proposition 2.1 and Theorem 2.2 we have

$$ICSL(A_i) \neq ICFL(A_i) .$$

From Definition 2.2 we have $ICSL(A_i)=ICSL(A_1)$, therefore this proposition is obtained.

Proposition 2.3 There is an $ICF(A_3)$ which generates the set of n^2 nodes without edges where n is any natural number.

Proof. The $ICF(A_3)$ in Table 2.14 generates the set of n^2 nodes without edges where n is any natural number.

[Example 2.8] We show an example of moves of the $ICF(A_3)$ of Table 2.14, in Table 2.15. In Tables 2.14 and 2.15, at each token, node numbers are from 1 to n, from left to right. And in Table 2.15, parenthesized rule numbers show the rules which are permitted to apply.

Proposition 2.4 There is no CS which generates the set of n^2 nodes without edges where n is any natural number.

Proof. Let there be a CS which generates the set of n^2 nodes

Table 2.14 The *ICF*(A_3) which generates n^2 nodes without edges.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
a'	S ≡> a	(1)		S ≡> a'	(1')
a"	S ≡> a a a a	(2)		S ≡> a"	(2')
A	S ≡> A C C C C C C E B	(3)		S ≡> A C C C C C C E B	(3')
a	A ≡> a	(4)		A ≡> a	(4')
a	C ≡> a	(5)	a C	C ≡> a	(5')
a	E ≡> a	(6)	a E	E ≡> a	(6')
a	B ≡> a	(7)	a B	B ≡> a	(7')
A' C B*	C ≡> C_1	(8)		A ≡> A'	(8')
C' C B*	C ≡> C_1	(9)	C_1 B* E	B* ≡> E B**	(9')
C' E B^*_1	E ≡> E_3	(10)	C_1 B**	E ≡> E'	(10')
A" C C'_3	C'_3 ≡> C	(11)	. B**	C ≡> C'	(11')
A' C B	B ≡> E B*	(12)	E_3 B^{**}_1	E ≡> E'	(12')
B**	B* ≡> B**	(13)	C'_3	E' ≡> C'_3	(13')
	B** ≡> B	(14)	C'_3	E ≡> C'_3	(14')
C' C B	B ≡> E B*	(15)	A"	A' ≡> A"	(15')
C' E B	B ≡> E B^*_1	(16)	A" C	C" ≡> C	(16')

continue

$\underline{E\ \ E\ \ E'}\quad \underset{\cdot}{E_3} \Rrightarrow \underset{\cdot}{C'_3}$	(17)
$\underline{C'_3\ \ E}\quad \underset{\cdot}{B^{**}_1}\quad \underset{\cdot}{E} \Rrightarrow \underset{\cdot}{C'_3}$	(18)
$\underset{\cdot}{B}\quad \underset{\cdot}{C'_3}\quad \underset{\cdot}{A'}\quad \underset{\cdot}{A} \Rrightarrow \underset{\cdot}{A''}$	(19)
$\underset{\cdot}{A''}\quad \underline{E\ \ C'_3}\quad \underset{\cdot}{C'_3} \Rrightarrow \underset{\cdot}{C}$	(20)
$\underset{\cdot}{A''}\quad \underline{C\ \ C'}\quad \underset{\cdot}{C'} \Rrightarrow \underset{\cdot}{C}$	(21)
$\underline{A''\ \ C\ \ C}\quad \underset{\cdot}{A''} \Rrightarrow \underset{\cdot}{A''_1}$	(22)
$\underline{A''\ \ C'\ \ C}\quad \underset{\cdot}{C'} \Rrightarrow \underset{\cdot}{C}$	(23)
$\underset{\cdot}{A}\quad \underset{\cdot}{A''_1} \Rrightarrow \underset{\cdot}{A}$	(24)
$\underset{\cdot}{B^{***}}\underset{\cdot}{C'}\quad \underset{\cdot}{C_1} \Rrightarrow \underset{\cdot}{C'}$	(25)
$\underset{\cdot}{B^{***}}\underset{\cdot}{C'}\quad \underset{\cdot}{C_2} \Rrightarrow \underset{\cdot}{C'}$	(26)
$\underset{\cdot}{B^{**}_1}\quad \underset{\cdot}{B^{*}_1} \Rrightarrow \underset{\cdot}{B^{**}}$	(27)
$\underline{C'_3\ \ C'_3}\quad \underset{\cdot}{B^{**}_1}\quad \underset{\cdot}{B^{**}_1} \Rrightarrow \underset{\cdot}{B}$	(28)

$\underset{\cdot}{A''_1}\quad \underset{\cdot}{C'}\quad \underset{\cdot}{C'} \Rrightarrow \underset{\cdot}{C}$	(17')
$\underset{\cdot}{A''_1}\quad \underset{\cdot}{A''} \Rrightarrow \underset{\cdot}{A}$	(18')
$\underset{\cdot}{C_2}\quad \underset{\cdot}{B^*}\quad \underset{\cdot}{E}\quad \underset{\cdot}{B^*} \Rrightarrow \underline{E\ \ B^{**}}$	(19')
$\underset{\cdot}{E_3}\quad \underset{\cdot}{B^*_1}\quad \underset{\cdot}{E}\quad \underset{\cdot}{B^*_1} \Rrightarrow \underline{E\ \ B^{**}_1}$	(20')
$\underset{\cdot}{B^*}\quad \underset{\cdot}{E}\quad \underset{\cdot}{B} \Rrightarrow \underset{\cdot}{B^*}$	(21')
$\underset{\cdot}{B}\quad \underset{\cdot}{B^{**}} \Rrightarrow \underset{\cdot}{B^{***}}$	(22')
$\underset{\cdot}{B}\quad \underset{\cdot}{B^{***}} \Rrightarrow \underset{\cdot}{B}$	(23')
$\underset{\cdot}{C_2}\quad \underset{\cdot}{B^{**}}\quad \underset{\cdot}{C} \Rrightarrow \underset{\cdot}{C'}$	(24')
$\underset{\cdot}{B^*_1}\quad \underset{\cdot}{E}\quad \underset{\cdot}{B^{**}} \Rrightarrow \underset{\cdot}{B^*_1}$	(25')
$\underset{\cdot}{C'_3}\quad \underset{\cdot}{B}\quad \underset{\cdot}{B^{**}} \Rrightarrow \underset{\cdot}{B}$	(26')
$\underset{\cdot}{A''}\quad \underset{\cdot}{C}\quad \underset{\cdot}{C'_3} \Rrightarrow \underset{\cdot}{C}$	(27')

Table 2.15 Moves of the interactive system of Table 2.14.

Applied production	token of token system 0	Applied production	token of token system 1
	S		S
	S	3' (1',2')	A C C C C C C C E B
3	A C C C C C C C E B	8' (4')	A'C C C C C C C E B
12	A C C C C C C C E E B*	21'	A'C C C C C C C E B*
8	A C_1C C C C C C E E B*	9'	A'C C C C C C C E E B**
13	A C_1C C C C C C E E B**	10'	A'C'C C C C C C E E B**
14	A C_1C C C C C C E E B	22'	A'C'C C C C C C E E B***
25	A C'C C C C C C E E B	23'	A'C'C C C C C C E E B
15	A C'C C C C C C E E E B*	21'	A'C'C C C C C C E E B*
6	A C'C_2C C C C C E E E B*	19'	A'C'C C C C C C E E E B**
α — 13	A C'C_2C C C C C E E E B**	β — 11'	A'C'C'C C C C C E E E B**
14	A C'C_2C C C C C E E E B	22'	A'C'C'C C C C C E E E B***
26	A C'C'C C C C C E E E B	23'	A'C'C'C C C C C E E E B
α^4	A C'C'C'C'C'C'E E E E E E E B	β^4	A'C'C'C'C'C'C'E E E E E E E B
16	A C'C'C'C'C'C'E E E E E E E E B^*_1	25'	A'C'C'C'C'C'C'E E E E E E E B^*_1
10	A C'C'C'C'C'C'E_3E E E E E E E B^*_1	20'	A'C'C'C'C'C'C'E E E E E E E E B^{**}_1
27	A C'C'C'C'C'C'E_3E E E E E E E B^{**}_1	12'	A'C'C'C'C'C'C'E'E E E E E E E B^{**}_1
17	A C'C'C'C'C'C'C_3E E E E E E E B^{**}_1	13'	A'C'C'C'C'C'C'C'_3E E E E E E E B^{**}_1

continue

18	A C'C'C'C'C'C'C'$C'_3C'_3$E E E E E E B**$_1$	14'	A'C'C'C'C'C'C'C'$C'_3C'_3$E E E E E E B**$_1$
28	A'C'C'C'C'C'C'C'$C'_3C'_3$E E E E E E B	26'	A'C'C'C'C'C'C'C'$C'_3C'_3$E E E E E E B
19	A"C'C'C'C'C'C'C'$C'_3C'_3$E E E E E E B	15'	A"C'C'C'C'C'C'C'$C'_3C'_3$E E E E E E B
20	A"C'C'C'C'C'C'C'C'_3C E E E E E E E B	27'	A"C'C'C'C'C'C'C'C'_3C E E E E E E E B
11	A"C'C'C'C'C'C'C'C C E E E E E E E B	27'	A"C'C'C'C'C'C'C'C C E E E E E E E B
21[5]	A"C'C C C C C C C E E E E E E E B	16'[5]	A"C'C C C C C C C E E E E E E E B
23	A"C C C C C C C C E E E E E E E B	16'	A"C C C C C C C C E E E E E E E B
22	A_1"C C C C C C C C E E E E E E E B	18'	A C C C C C C C C E E E E E E E B
24	A C C C C C C C C E E E E E E E B	4' (8')	a C C C C C C C C E E E E E E E B
4	a C C C C C C C C E E E E E E E B	5' (6',7')	a C a C C C C C C E E E E E E E B
5	a C a C C C C C C E E E E E E E B	7' (5',6')	a C a C C C C C C E E E E E E E a
7	a C a C C C C C C E E E E E E E a	5'[7]+6'[6]	a a a a a a a a a a a a a a a a a
5[7]+6[6]	a a a a a a a a a a a a a a a a a		a a a a a a a a a a a a a a a a a

without edges where n is any natural number. And let a set of rules of the CS be

$$P = \{ p_1, p_2, \dots , p_l \},$$

then any CS does not generate an egde. Because any CS can not delete edges according to Definition 2.9. So, if an edge is generated then it can not be deleted. Then, let a set of non-terminal symbols of the CS be

$$V_n = \{ v_1, v_2, \dots , v_m \}$$

and a set of terminal symbols be

$$V_t = \{ t \}.$$

Then we can represent a situation of the CS by

$$(a_1, a_2, \dots , a_m, a_{m+1}),$$

where a_i is the number of nodes whose labels are v_i, and a_{m+1} is the number of nodes whose labels are t. And using this representation, a production p_i can be represented as below.

$$(b_{i1}, b_{i2}, \dots , b_{im}, b_{im+1}) \Rightarrow (c_{i1}, c_{i2}, \dots , c_{im}, c_{im+1}),$$

where, b_{ij} or c_{ij} is the number of nodes whose labels are v_i and b_{im+1} or c_{im+1} is the number of nodes whose labels are t. And it represents that if there are more than b_{ij} nodes whose labels are v_j and b_{im+1} nodes whose labels are t in a situation of the CS, then those nodes are replaced by c_{ij} nodes whose labels are v_j and c_{im+1} nodes whose labels are t. In this formalism under two formulas are satisfied.

$$0 \le b_{im+1} \le c_{im+1}$$

$$0 \le \sum_{j=1}^{m+1} b_{ij} \le \sum_{j=1}^{m+1} c_{ij}$$

We represent a situation in a process of generating n^2 nodes as

$$(g_{k1}, g_{k2}, \cdots , g_{km}, g_{km+1}).$$

Then let

$$\max_{j=1,m} g_{kj} \le \alpha$$

be satisfied where α is a natural number. So the *CS* has only finite states and it can not generate n^2 nodes without edges. Then let there be n where

$$\max_{j=1,m} g_{kj} > \alpha$$

for any natural number α. And let sequences of productions which are applied in processes of generateing n^2, n'^2 and n''^2 be

$$(p_{11}, p_{12}, \cdots , p_{1k}),$$

$$(p_{21}, p_{22}, \cdots , p_{2k'}),$$

and

$$(p_{31}, p_{32}, \cdots , p_{3k''})$$

respectively, where only the numbers of nodes whose labels are t are different. So there is a sequence of applications of productions which increase the number of nodes whose labels are t by a constant c. So there is a sequence of applications of productions which generates n^2+c nodes whose labels are t. So there is no CS which generates n^2 nodes whose labels are t without edges. So this proposition is proved.

Theorem 2.3

$$ICFL(\mathrm{A}_i) \nsupseteq CSL \qquad i \geq 3$$

Proof. From Propositions 2.3 and 2.4, we have $ICFL(\mathrm{A}_3) \not\subset CSL$. So from Definition 2.10 we have

$$ICFL(\mathrm{A}_i) \not\subset CSL \qquad i \geq 3 .$$

From Proposition 2.1 we have $ICFL(\mathrm{A}_i) \not\supset CSL$.

So we have this theorem.

2.5 Interactive Systems of mode B

We will define an interactive system of mode B which has no correspondences between vertices of a token of token system T_0 and them of token system T_1. We express the width of the pass between token system T_0 and token system T_1 by a suffix i such as mode B_i. In the definition the width of the pass between token system T_0 and token system T_1 is represented by the number of vertices of sub-tokens which are referred by interaction functions. A difference between an interactive system of mode A and one of mode B is as follows.

When a vertex and a token are given, the value of an interactive function of an interactive system of mode A is defined by the sub-tokens whose center corresponds to the vertex, but the value of an interaction function of an interactive system of mode B depends on only the token, but the vertex. Therefore an interaction function of an interactive system of mode B take the same value for any vertex when given tokens are same.

Definition 2.11. We say that an interactive system which has no correspondence between vertices of a token of token system T_0 and them of token system T_1 is mode B. More formally, it is mode B_j that an interactive system whose interaction functions, f_0 and f_1, are both have a form of (2.8).

$$f(n, t) = \bigcup_{k \leq j} \bigcup_{m \in N} f'({}_m t_k) \qquad (2.8)$$

where, f' is a function: $\{{}_m t_k\} \to 2^P$.

In this section we will show,

$$CFL \subset ICFL(B_1) \subseteq ICFL(B_2) \subseteq \ldots \subseteq ICFL(B_n)$$

and $$CSL \subset ICSL(B_1) = ICSL(B_2) = \ldots = ICSL(B_n),$$

where $ICFL(B_i)$ is the family of the sets of outputs of $ICF(B_i)$ (interactive systems of mode B_i whose token systems are both context-free), $ICSL(B_i)$ is the family of the sets of outputs of $ICS(B_i)$ (interactive systems of mode B_i whose token systems are both context-sensitive).

Here we show one example of $ICF(B_1)$.

[Example 2.9] The interactive system which is shown in Example 2.3 is $ICF(B_1)$. Because interaction functions of the interactive system is defined as follows.

$$f_0(i,\ t) = \bigcup_{k \le 1} \bigcup_{n \in N} f'_0({}_n t_k)$$

where $$f'_0(\ S\ ) = \{\ S \Rrightarrow \underline{A\ b\ B}\ \},$$

$$f'_0(\ A\ ) = \{\ A \Rrightarrow \underline{a\ A}\ \},$$

$$f'_0(\ B\ ) = \{\ B \Rrightarrow \underline{a\ B}\ \},$$

and $$f'_0(\ d\ ) = \{\ A \Rrightarrow a\ ,\ \ B \Rrightarrow a\ \}.$$

And $$f_1(i,\ t) = \bigcup_{k \le 1} \bigcup_{n \in N} f'_1({}_n t_k)$$

where $$f'_1({}_n t_0) = \{\ S \Rrightarrow A\ ,\ \ A \Rrightarrow B\ ,\ \ B \Rrightarrow A\ ,\ \ B \Rrightarrow d\ \}.$$

Therefore the interactive system in Example 2.3 is of mode B_1.

Theorem 2.4.

$$CFL \subset ICFL(B_1) \subseteq ICFL(B_2) \subseteq \cdots \subseteq ICFL(B_n)$$

Proof. We shall show that $CFL \subseteq ICFL(B_n)$. From Definitions 2.1, 2.2, 2.8 and 2.11, for any context-free web grammar system:

$$CF = (\ s,\ V_n,\ V_t,\ P\ ),$$

let $$T_0 = (\ s,\ V_n,\ V_t,\ P\ ),$$

$$f_0(n,\ t) = P,$$

where n is any natural number and t is any token, T_1 is any context-free token system and f_1 is any interaction function of mode B_1.

Then

$$S = (T_0, T_1, f_0, f_1)$$

is $ICF(B_1)$ and simulates the CF. Therefore $CFL \subseteq ICFL(B_1)$ is proved. From Definition 2.11 $ICF(B_i)$ is also $ICF(B_{i+1})$, therefore

$$ICFL(B_i) \subset ICFL(B_{i+1})$$

is apparent.

Next we shall show

$$CFL \neq ICFL(B_1).$$

In [1], only the graphs as in Fig. 2.5 can not be generated by CF (any context-free web grammar system), but the interactive system in Example 2.3 generates only the graphs as in Fig. 2.5, and it is of mode B_1 and whose token systems are context-free. Therefore $CFL \neq ICFL(B_1)$ is apparent. Thus $CFL \subset ICFL(B_1)$ is proved.

Theorem 2.5.

$$CSL \subset ICSL(B_1) = ICSL(B_2) = \dots = ICSL(B_n)$$

Proof. We shall show that $CSL \subset ICSL(B_1)$. From Definitions 2.1, 2.2, 2.9 and 2.11 $CSL \subseteq ICSL(B_1)$ is apparent. Next we show that $CSL \neq ICSL(B_1)$. The set of all separable graphs is not included by CSL from [1]. The interactive system of Table 2.16 generates all separable graphs, and it is $ICS(B_1)$ (an interactive system of mode B_1 and whose token systems are context-sensitive). Therefore

Table 2.16 The *ICS*(B_1) which generates all separable graphs.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
	S ≡> A' E	(1)		S ≡> A E'	(1')
	E ≡> B' B"	(2)	(B")	E' ≡> E B	(2')
	B" ≡> B	(3)	(B")	B ≡> B' B	(3')
(B")	B' B' ≡> B' B'	(4)	(B)	B ≡> B"	(4')
	B" ≡> B' B"	(5)	(C')	B" ≡> C	(5')
(B")	B B' ≡> B B'	(6)	(C')	C B' ≡> C" C	(6')
	B ≡> C'	(7)	(C')	C E ≡> C" D	(7')
(C)	B' ≡> C	(8)	(E)	D A ≡> C" A E'	(8')
(D)	C' A' ≡> C A E	(9)	(t')	D A ≡> C" t'	(9')
(D)	C' A ≡> C A E	(10)			
(D)	C' A ≡> C t'	(11)			
(t')	t' C ≡> t' C	(12)			
(t')	t' C ≡> t' t	(13)			
(E')	E A' ≡> E A t	(14)			
(E')	E A ≡> E A t	(15)			
	t C ≡> t t	(16)			

$$CSL \subset ICSL(B_1)$$

is proved. We shall explain moves of the interactive system of Table 2.16 in Table 2.17.

Next we show $ICSL(B_i) = ICSL(B_{i+1})$. From Definition 2.11 $ICSL(B_i) \subseteq ICSL(B_{i+1})$ is apparent. So we show

$$ICSL(B_1) \supseteq ICSL(B_i).$$

In order to prove this we shall show that $ICS(B_1)$ can simulate $ICS(B_i)$ (an interactive system of mode B_i whose token systems are context-sensitive). $ICS(B_i)$ is expressed by two sets of the rules of (2.9) as shown in Table 2.16.

$$(t_1) \quad t_2 \Rightarrow t_3 \tag{2.9}$$

$$(y^+) \quad t_2^\times \Rightarrow t_3^\circ \tag{2.10}$$

$$(y^+) \quad t_2^\times \Rightarrow t_3^{01} \tag{2.11}$$

$$(x^\circ) \quad y^+ \Rightarrow y^\times \tag{2.12}$$

$$(x^\times) \quad t_1^\circ \Rightarrow t_1^+ \tag{2.13}$$

Basically the rule of (2.9) is simulated by the rules of (2.10)-(2.13) of $ICS(B_1)$. If the rule of (2.9) belongs to token system T_0 of $ICS(B_i)$, then the rules of (2.10) and (2.11) belong token system T_0, and the rules of (2.12) and (2.13) belong to token system T_1, and vice versa. The rules of (2.13) fill the role of checking whether token t_1 in (2.9) is contained in the token of the token system which contains

Table 2.17 Moves of the interactive system of Table 2.16.

Applied productions	Tokens of token system 0	Applied productions	Tokens of token system 1
	S		S
1	A' E	1'	A E'
2	A' B' B"	2'	A E B
5 (3)	A' B' B' B"	3'	A E B' B
3 (5, 7)	A' B' B' B	4'	A E B' B"
4 (6, 7)	A' B' B' B		A E B' B"
6 (7)	A' B' B' B		A E B' B"
6 (7)	A' B' B' B		A E B' B"
7	A' B' B' C'	5'	A E B' C
8	A' B' C C'	6'	A E C C"
8	A' C C C'	7'	A D C" C"
9	E A C C C	8'	E' A C" C" C"

the rule of (2.9). We shall explain differences between (2.3) of $ICS(A_i)$ and (2.9) of $ICS(B_i)$.

In the rule of (2.3) of $ICS(A_i)$, the concept of a sub-token is important. A sub-token has its center which defines correspondence between t_1 and t_2 in the rule of (2.3). But in the rule of (2.9), t_1, t_2 and t_3 are not sub-tokens, but tokens. In $ICS(B_i)$ there is no correspondence between vertices of a token of token system T_0 and those of token system T_1. Therefore it is not necessary to express correspondence between t_1 and t_2 in (2.9) which is expressed by a center of a sub-token. So in (2.9) t_1, t_2 and t_3 are tokens. And in Table 2.16 there is no mark of a center.

In the rule of (2.13) t_1° is constructed with only t_1 of (2.9) and x° (any ove vertex token whose label is superfixed with '$\circ$'). t_1 may contain x°. For any t_1 there is the rule of (2.13) which contains the t_1°. And t_1^+ in (2.13) is the token which is obtained by changing x° in t_1° in (2.13) to x^+. If the rule of (2.13) is applied, then the rules of (2. 10) and (2.11) are tried to apply when they know that they are allowed to apply by seeing the vertex whose label is superfixed with '+'. In the rules of (2.10) $t_2^\times$ is any token which is constructed with only t_2 in (2.9) and $x^\times$ (any one vertex token whose label is superfixed with '$\times$'). t_2 may contain $x^\times$. For any t_2 there is the rule of (2.10) which contains the $t_2^\times$. And t_3° in the rule of (2.10) is the token which is obtained by changing $x^\times$ in $t_2^\times$ to x° and t_2 in $t_2^\times$ to t_3 in the rule of (2.9). The rule of (2.12) fills the role of handing over a move to the other token system, when it is informed of application of the rule of (2.10) by the vertex whose label is supersixed with

'∘'. The rule of (2.11) is the same as the rule of (2.10) except t_3^{01}. t_3^{01} in the rule of (2.11) is obtained by changing the superfix '∘' of t_3° in the rule of (2.10) to '01'. The rule of (2.11) has the role of requiring the other token system to perform an empty move (preservation of the token) which is defined in Definition 2.6. We shall later show an example of changing the rule of (2.9) of $ICS(B_3)$ to the rules of (2.10) and (2.13).

It is necessary to confirm that no productions can be applied when an empty move is required by the rule of (2.11). To confirm that no productions can be applied, the rules in Table 2.18 or Table 2.19 are necessary. If it is confirmed that no productions can be applied to the token which is in token system T_0 by the rules in Table 2.18 and Table 2.19, then a move is handed over to token system T_1, and vice versa. We shall later explain Table 2.18 and Table 2.19.

The interactive system which is constructed from the rules of (2.10)-(2.13) and in Tables 2.18 and 2.19 is of mode B_1. And it simulates the moves of the original interactive system. Therefore $ICS(B_i)$ can be simulated by $ICS(B_1)$. Then we have

$$ICSL(B_1) \supseteq ICSL(B_i).$$

Therefore $$ICSL(B_1) = ICSL(B_i)$$

is apparent. So

$$ICSL(B_i) = ICSL(B_{i+1})$$

is proved.

Table 2.18 Productions which confirms that no rules can be applied.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
$(x^{\times})$	$t_2^{01} \Rightarrow n.s.$	(A)	(x^{01})	$t_1^{+} \Rightarrow t_1^{\times}$	(a)
(x^{+})	$x^{01} \Rightarrow x^{02}$	(B)	$(n.s.)$	$x^{\times} \Rightarrow n.s.$	(b)
			(x^{01})	$x^{\times} \Rightarrow x^{+}$	(c)

Table 2.19 Rules which confirms that no rules can be applied.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
$(x^{\times})$	$t_2^{01} \Rightarrow n.s.$	(A)	(x^{01})	$x^{+} \Rightarrow x^{\times}$	(a)
(x^{+})	$x^{01} \Rightarrow x^{02}$	(B)	$(n.s.)$	$x^{\times} \Rightarrow n.s.$	(b)
			(x^{01})	$x^{\times} \Rightarrow x^{+}$	(c)

We shall show an example of changing the rules of (2.9) of $ICS(B_3)$ to the rules of (2.10) and (2.13) of $ICS(B_1)$.

[Example 2.10] If the rule of (2.9) of $ICS(B_3)$ is Fig. 2.7 (1), then the rules of (2.10) of $ICS(B_1)$ are Figs. 2.7 (2)-(7) and the rules of (2.13) are Figs. 2.7 (8)-(18).

We explain the rules in Tables 2.18 and 2.19. In Tables 2.18 and 2.19, 'x^{α}' expresses one vertex token whose lablel is superfixed with 'α', and '*n. s.*' does the token whose labels are the nonterminal symbols which can not be rewritten, and t_i^{α} expresses the token which is constructed with only one vertex token x^{α} and t_i which is in the rule of (2.9). t_i may contain x^{α}.

Next we shall explain moves of the rule in Table 2.18. First, (a) in Table 2.18 is notified by the vertex whose label is superfixed with '01' that an empty move is required. Then (a) in Table 2.18 is tried to apply. If t_1 in the rule of (2.9) exists in a token of token system t_1, then (a) in Table 2.18 is applied. If (a) is applied, then (A) is tried to apply. If t_2 in the rules of (2.9) exists in a token of token system T_0, then (A) is applied. So the nontermial symbols which can not be rewritten are generated. If t_2 in the rules of (2.9) does not exist in a token of token system T_0, then there is no rules which can be applied. So in token system T_1, (c) in Table 2.18 is applied. Then (B) is applied. If (A) is applied, then (b) is applied. So in token system T_0 and token system T_1 the nonterminal symbol which can not be rewritten is generated in a token. Therefore the interactive system can not generate an output. If (a) in Table 2.18

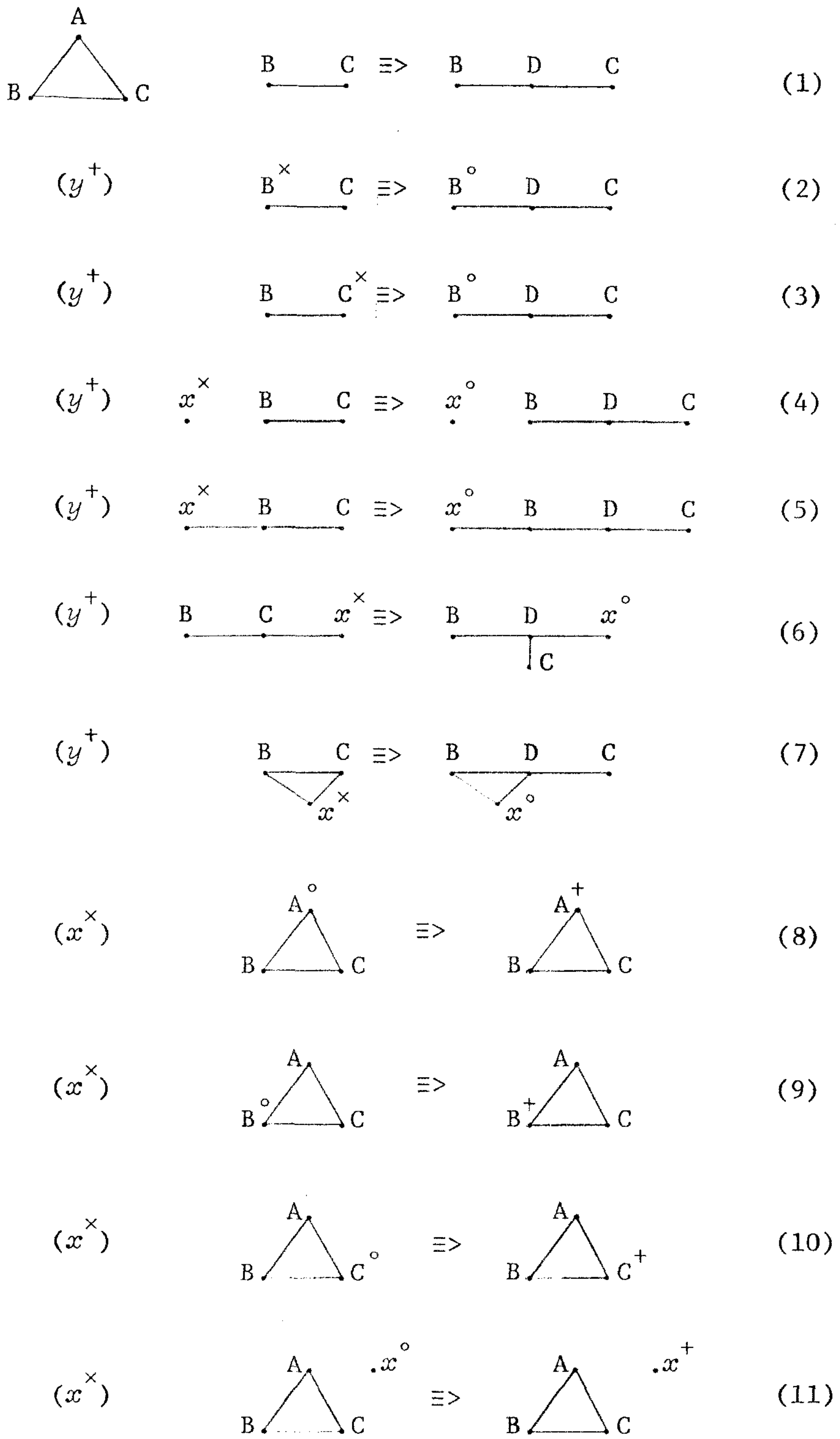


Fig. 2.7 Examples of productions transformed from $ICS(B_3)$ (partial)

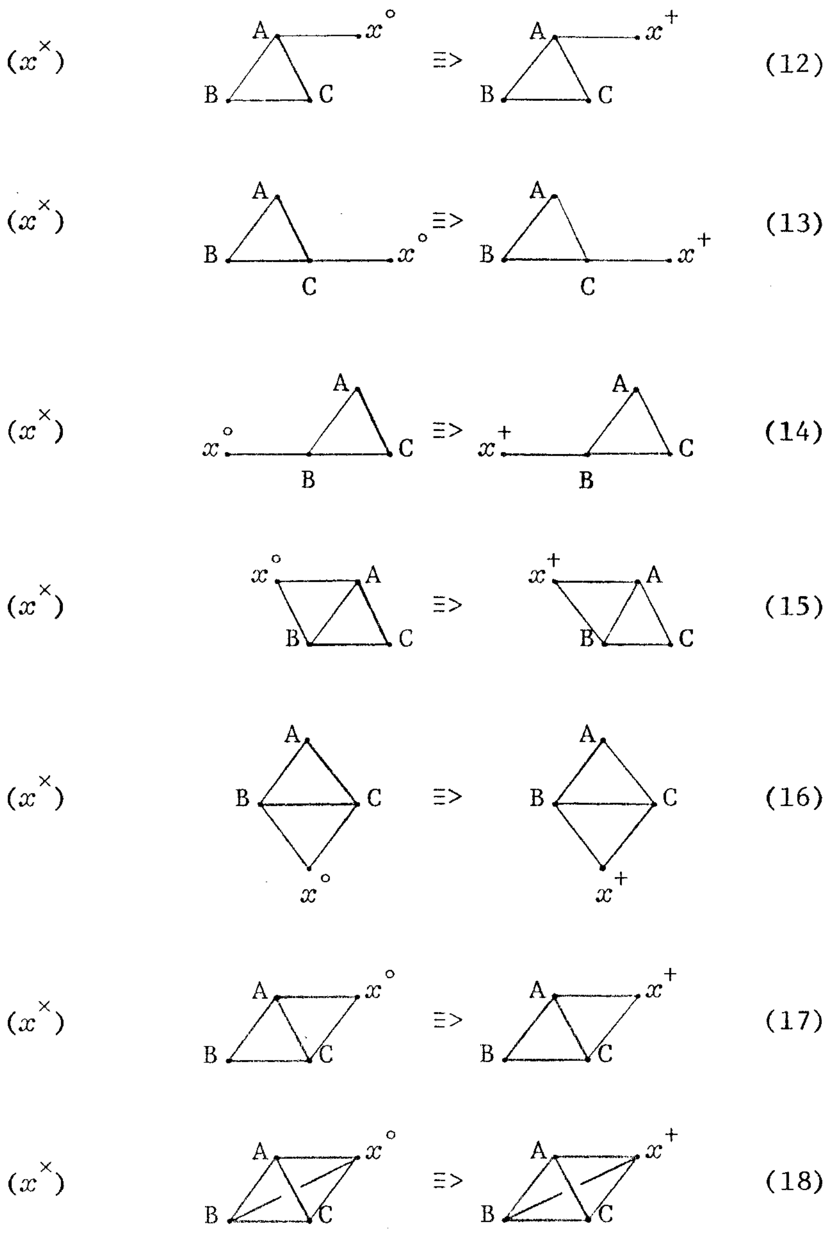


Fig. 2.7 Continued.

is not applied, then (B) is applied. (B) notifies that this check ends by the vertex whose label is superfixed with '02'.

In Table 2.19 (a) is applied when it is tried to apply. Other rules in Table 2.19 are the same as them in Table 2.18.

Proposition 2.5.

$$ICFL(B_i) \not\supseteq CSL$$

Proof. Ezawa [14] shows that CSL includes the set of all complete graphs. In any $ICF(B_i)$ (an interactive system of mode B_i whose token systems are context-free) token system T_0 is a context-free web grammar system, therefore it can not generate all complete graphs. So $ICFL(B_i) \not\supseteq CSL$ is apparent.

We shall show a relation between $ICSL(B_i)$ and $ICFL(B_i)$

Proposition 2.6.

$$ICSL(B_1) \supset ICFL(B_i)$$

Proof. From Definitions 2.8, 2.9 and 2.11 $ICSL(B_i) \supseteq ICFL(B_i)$ is apparent. From Proposition 2.5 and Theorem 2.5 we have this proposition.

2.6 Relations between Interactive Systems of mode A and those of mode B

We shall show relations between interactive systems of mode A whose token systems are context-free and those of mode B whose token systems are context-free.

Theorem 2.6

$$ICFL(B_i) \not\supseteq ICFL(A_3)$$

Proof. To prove this theorem we show that $ICF(A_3)$ can generate only the graphs such as of Fig. 2.8, and $ICF(B_i)$ can not do. On Table 2.20 there is $ICF(A_3)$ which generates only the graphs such as of Fig. 2.8. To generate only the graphs such as of Fig. 2.8, an interactive system must know that all the paths from the root to leeves are same length. But in $ICF(B_i)$, each token systems can not know the connection among vertices. Therefore $ICF(B_i)$ can not generate only the graphs such as of Fig. 2.8. So we have this theorem.

We show an example of moves of the interactive system of Table 2.20. In Table 2.21 we show moves of the interactive system.

We shall show relations between interactive systems of mode A whose token systems are context-sensitive and them of mode B whose token systems are context-sensitive.

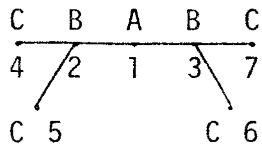
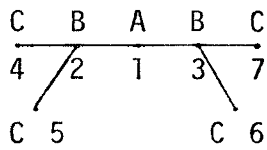
Theorem 2.7

$$ICSL(A_1) \supseteq ICSL(B_i)$$

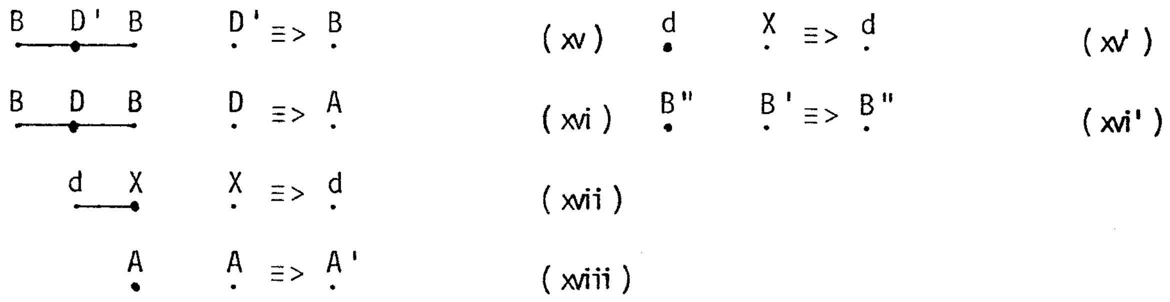
Proof. We shall prove this theorem by showing that the moves of $ICS(B_1)$ can be simulated by moves of $ICS(A_2)$.

$$(\ x\ ) \qquad t_2 \ \equiv\!> \ t_3 \tag{2.14}$$

Table 2.20 The *ICF*(A_3) which generates the graphs as in Fig. 2.8.

Interaction function 0	Productions of token system 0		Interaction function 1	Productions of token system 1	
	S ≡> 	(i)		S ≡> 	(i')
A	A ≡> d	(ii)	A'	A ≡> A'	(ii')
A' B	B ≡> B'	(iii)	d	A ≡> d	(iii')
B' B	B ≡> B'	(iv)	B'	B ≡> B'	(iv')
B' C	C ≡> C'	(v)	C'	C ≡> C'	(v')
C'	C' ≡> C" B" C" (2 1 3)	(vi)	B"	C' ≡> C" B" C" (2 1 3)	(vi')
B" B' B"	B' ≡> B"	(vii)	A"	A' ≡> A"	(vii')
B" A' B"	A' ≡> A"	(viii)	d	A" ≡> d	(viii')
A"	A" ≡> d	(ix)	D	A" ≡> D	(ix')
A"	A" ≡> D	(x)	D'	B" ≡> D'	(x')
D B"	B" ≡> D'	(xi)	D"	C" ≡> D"	(xi')
D' B"	B" ≡> D'	(xii)	B	D" ≡> C B C (2 1 3)	(xii')
D' C"	C" ≡> D"	(xiii)	B	D' ≡> B	(Xiii')
D"	D" ≡> C B C (2 1 3)	(xiv)	A	A" ≡> A	(xiv')

continue



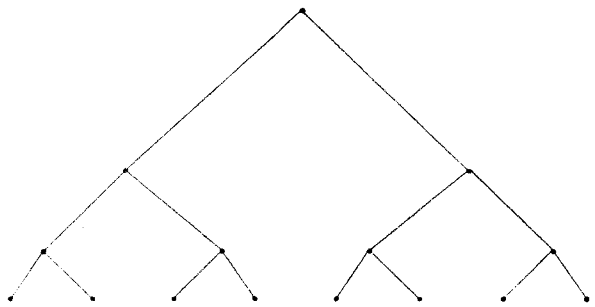
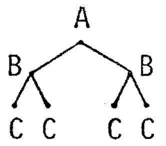
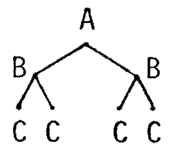
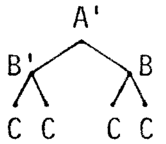
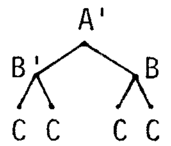
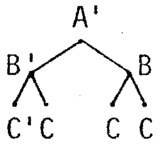
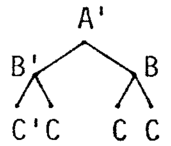
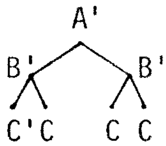
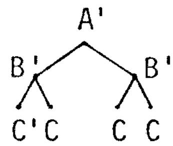
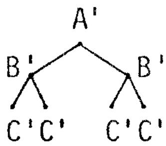
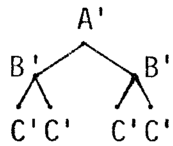
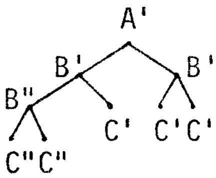
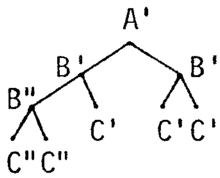
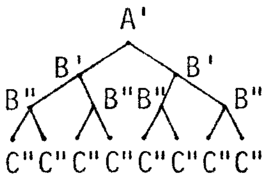
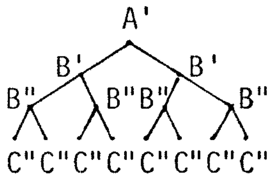


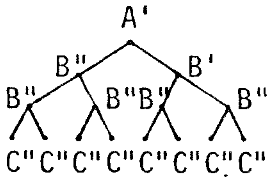
Fig. 2.8 The complete binary tree of depth 3.

Table 2.21 Moves of the interactive system of Table 2.20.

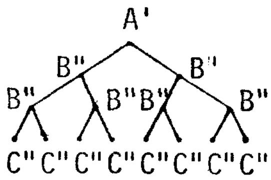
Applied productions	Token of token system 0	Applied productions	Token of token system 1
	.S		.S
i		i'	
xviii		ii'	
v		v'	
iii		iv'	
v^3		v'^3	
vi		vi'	
vi^3		vi'^3	

continue

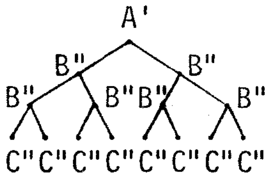
vii



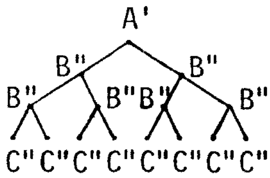
xvi'



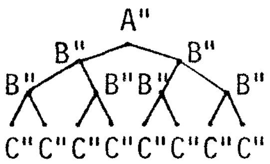
vii



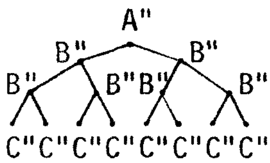
xvi'



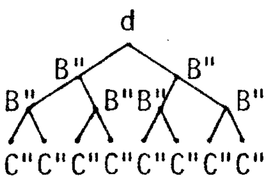
viii



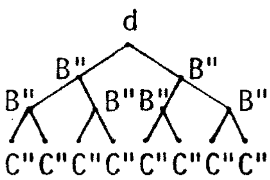
vii'



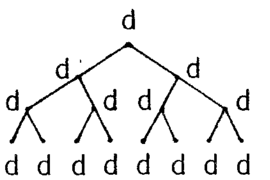
ix



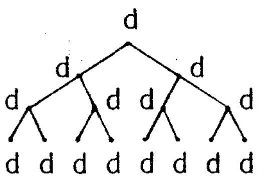
viii'



xvii [14]



xv [14]



$$(x) \quad z\text{-}t_2 \ \Rightarrow \ z\text{-}t_3 \tag{2.15}$$

$$(x\ y) \quad z\text{-}t_2 \ \Rightarrow \ z\text{-}t_3 \tag{2.16}$$

$$(x\text{-}y) \quad z\text{-}t_2 \ \Rightarrow \ z\text{-}t_3 \tag{2.17}$$

$ICS(B_1)$ is expressed by two sets of the rules of (2.14). The rule of (2.14) of $ICS(B_1)$ is simulated by the rules of (2.15)-(2.17) of $ICS(A_2)$. If the rule of (2.14) of $ICS(B_1)$ belongs to token system T_0, then the rules of (2.15)-(2.17) belongs to token system T_0 of $ICS(A_2)$, and vice versa. In the rules of (2.14) and (2.15) 'x' is one vertex token. In (2.15)-(2.17) z-t_2 is any sub-token which is constructed with only t_2 in the rule of (2.14) and z (any one vertex token), and whose center is z. t_2 may contain z. In (2.15)-(2.17) z-t_3 is the sub-token which is obtained by changing t_2 in z-t_2 to t_3 in the rule of (2.14). In the rule of (2.16) 'x y' expresses the sub-token "$\dot{x}$ $\dot{y}$" whose center is y. y expresses any label. We shall later show an example of changing the rules of (2.14) of $ICS(B_1)$ to the rules of (2.15)-(2.17) of $ICS(A_2)$. $ICS(B_1)$ can be simulated by $ICS(A_2)$. Therefore $ICSL(A_2) \supseteq ICSL(B_1)$ is apparent. So from Therorem 2.2 and Theorem 2.4 we have $ICSL(A_1) \supseteq ICSL(B_i)$.

We shall show an example of changing the rules of (2.14) to the rules of (2.15), (2.16) and (2.17).

[Example 2.11] If the rule of (2.14) of $ICS(B_1)$ has the form of Fig. 2.9 (a), then the rules of (2.15) of $ICS(A_2)$ which simulates the $ICS(B_1)$ are Figs. 2.9 (b)-(g); the rules of (2.16) of the $ICS(A_2)$ are

Figs. 2.9 (h)-(j); the rules of (2.17) of the $ICS(A_2)$ are Figs. 2.9 (k)-(m). In Fig. 2.9 the rules of (2.16) of the $ICS(A_2)$ which correspond to the rules of Figs. 2.9 (d)-(f) of (2.15) are omitted and the rules of (2.17) of the $ICS(A_2)$ which correspond to the rules of Figs. 2.9 (d)-(f) of (2.15) are omitted.

2.7 Relations among Interactive Systems which are constructed with two kinds of Token Systems

We shall show relations among interactive systems of mode A. An interactive system of mode A which is constructed with a context-free token system T_0 and a context-sensitive token system T_1 is expressed with $ICFS(A_i)$. We express the family of the sets of outputs of $ICFS(A_i)$ with $ICFSL(A_i)$. With $ICSF(A_i)$ we express an interactive system of mode A_i which is constructed with a context-sensitive token system T_0 and a context-free token system T_1. The family of the sets of outputs of $ICSF(A_i)$ is expressed with $ICSFL(A_i)$.

Theorem 2.8

$$ICSL(A_1) \supseteq ICSFL(A_i) \supset ICFL(A_i)$$

Proof. (2.18) is apparent from Theorem 2.2.

$$ICSL(A_1) \supseteq ICSFL(A_i) \supseteq ICFL(A_i) \qquad (2.18)$$

$$ICSFL(A_i) \neq ICFL(A_i) \qquad (2.19)$$

$ICSFL(A_i)$ includes the set of all complete graphs, and $ICFL(A_i)$ does

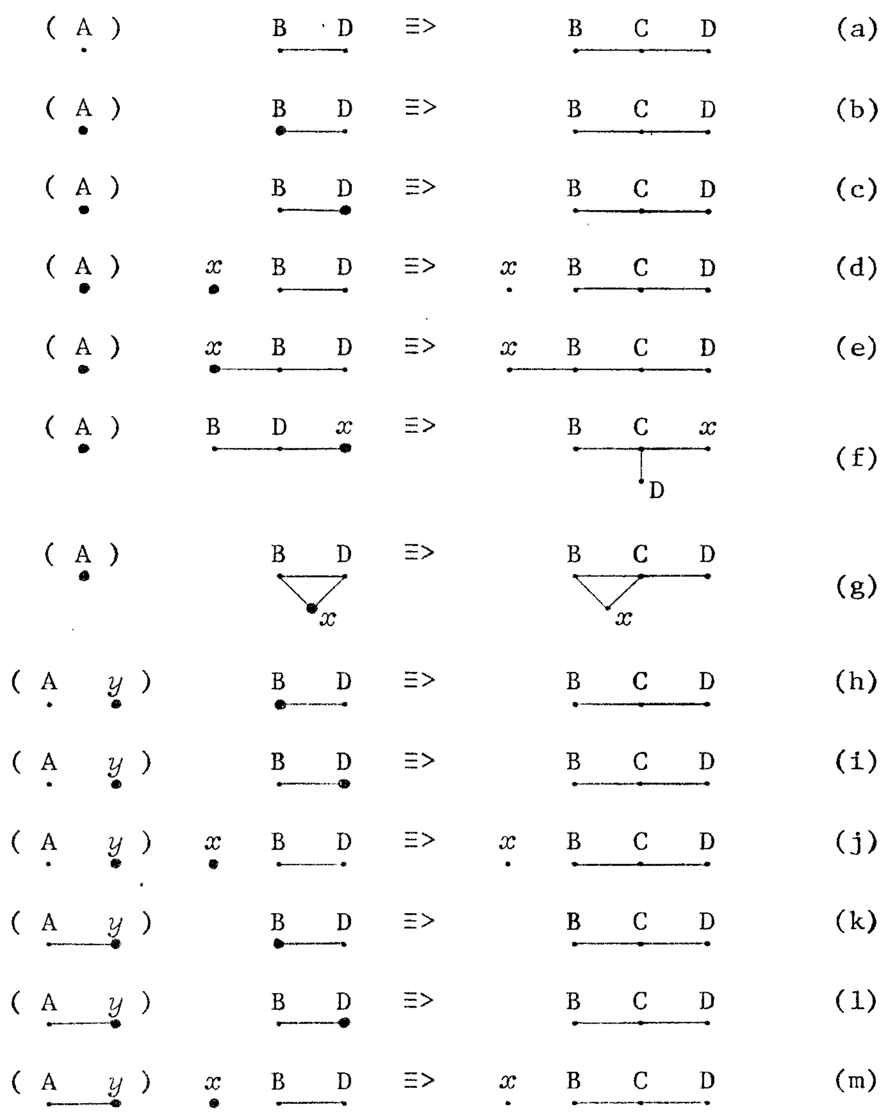


Fig. 2.9 Examples of productions transformed from $ICS(B_1)$ to $ICS(A_2)$.

not include the set of all complete graphs. Therefore (2.19) is apparent. So we have this theorem.

Theorem 2.9

$$ICSL(A_1) \supset ICFSL(A_i) \supseteq ICFL(A_i)$$

Proof. (2.20) is apparent from Theorem 2.2 and Definition 2.10.

$$ICSL(A_1) \supseteq ICFSL(A_i) \supseteq ICFL(A_i) \qquad (2.20)$$

$$ICSL(A_1) \neq ICFSL(A_i) \qquad (2.21)$$

$ICSL(A_1)$ includes the set of all complete graphs, and $ICFSL(A_i)$ does not include the set of all complete graphs. Therefore (2.21) is aparent. So we have this theorem.

We shall show relations among interactive systems of mode B. An interactive system of mode B_i which is constructed with a context-free token system T_0 and a context-sensitive token system T_1 is expressed with $ICFS(B_i)$, and $ICFSL(B_i)$ expresses the family of the sets of outputs of $ICFS(B_i)$. An interactive system of mode B_i which is constructed with a context-sensitive token system T_0 and a context-free token system T_1 is expressed with $ICSF(B_i)$, and $ICSFL(B_i)$ expresses the family of the sets of outputs of $ICSF(B_i)$.

Theorem 2.10

$$ICSL(B_1) \supset ICFSL(B_i) \supseteq ICFL(B_i)$$

Proof. (2.22) is apparent from Theorem 2.4 and Definition 2.11.

$$ICSL(B_1) \supseteq ICFSL(B_i) \supseteq ICFL(B_i) \qquad (2.22)$$

$$ICSL(B_1) \neq ICFSL(B_i) \qquad (2.23)$$

$ICSL(B_1)$ includes the set of all complete graphs, and $ICFSL(B_i)$ does not include the set of all complete graphs. Therefore (2.23) is apparent. So we have this theorem.

Theorem 2.11

$$ICSL(B_1) \supseteq ICSFL(B_i) \supset ICFL(B_i)$$

Proof. (2.24) is apparent from Theorem 2.4 and Definition 2.11.

$$ICSL(B_1) \supseteq ICSFL(B_i) \supseteq ICFL(B_i) \qquad (2.24)$$

$$ICSFL(B_i) \neq ICFL(B_i) \qquad (2.25)$$

$ICSFL(B_i)$ includes the set of all complete graphs, and $ICFL(B_i)$ does not include the set of all complete graphs. Therefore (2.25) is apparent. So this theorem is proved.

Proposition 2.7

$$ICFSL(A_1) \subseteq ICFSL(A_2) \subseteq \ldots \subseteq ICFSL(A_n)$$

$$ICSFL(A_1) \subseteq ICSFL(A_2) \subseteq \ldots \subseteq ICSFL(A_n)$$

Proof. From Definition 2.10 an interactive system of mode A_i is also of mode A_{i+1}. Therefore this proposition is apparent.

Proposition 2.8

$$ICFSL(B_1) \subseteq ICFSL(B_2) \subseteq \ldots \subseteq ICFSL(B_n)$$

$$ICSFL(B_1) \subseteq ICSFL(B_2) \subseteq \ldots \subseteq ICSFL(B_n)$$

Proof. From Definition 2.11 an interactive system of mode B_i is also of mode B_{i+1}. Therefore this proposition is apparent.

Proposition 2.9

$$ICFSL(A_i) \not\supseteq ICSFL(A_j)$$

$$ICFSL(B_i) \not\supseteq ICSFL(B_j)$$

Proof. $ICSFL(A_j)$ includes the set of all complete graphs, and $ICFSL(A_i)$ does not. $ICSFL(B_j)$ includes the set of all complete graphs and $ICFSL(B_i)$ does not. Therefore we have this proposition.

Proposition 2.10

$$ICSFL(A_i) \supset CSL \qquad i \geq 3$$

$$ICFSL(A_i) \not\supseteq CSL \qquad i \geq 3$$

Proof. From Theorem 2.3 $ICFL(A_i) \not\supseteq CSL$, and $CSL \subseteq ICSFL(A_i)$ is apparent. $ICFSL(A_i)$ can not generate all complete graphs so $CSL \not\subseteq ICFSL(A_i)$ is apparent. So we have this proposition.

Proposition 2.11

$$ICFSL(B_i) \not\supseteq CSL$$

$$ICSFL(B_i) \supseteq CSL$$

Proof. $ICFSL(B_i)$ can not generate all complete graphs so

$ICFSL(B_i) \not\supseteq CSL$ is apparent. $ICSFL(B_i) \supset CSL$ is apparent. So we have this proposition.

2.8 Conclusion

We have proposed a new formal system which models a system that is constructed with two sub-systems interacting each other, and defined two kinds of interaction functions: mode A and mode B. Then we have constructed 8 kinds of interactive systems: $ICS(A_i)$, $ICSF(A_i)$, $ICFS(A_i)$, $ICF(A_i)$, $ICS(B_i)$, $ICSF(B_i)$, $ICFS(B_i)$ and $ICF(B_i)$.

We have studied the abilities of interactive systems. From Theorems 2.2 and 2.4 we may say as follows. If the abilities of token systems which are parts of an interactive system are high, then the abilities of the interactive systems do not depend on the complexities of interaction functions. In the real world, by exchanges of simple informations high-able one can keep in enough communication. We show results of this chapter in Fig. 2.10. As shown in Fig. 2.10, it has been shown that the well known quotation from Homer's

Iliad: *"Two heads are better than one."*

is true for formal systems, too.

And it is has been shown that increase of ability from *CF* to *CS* has difference in quality from increase of ability from *CF* to $ICF(A_i)$.
So we may say that for modelling systems which is constructed with sub-systems interacting each other interaction is an important element and it is difficult to model a system which has interaction by a formal

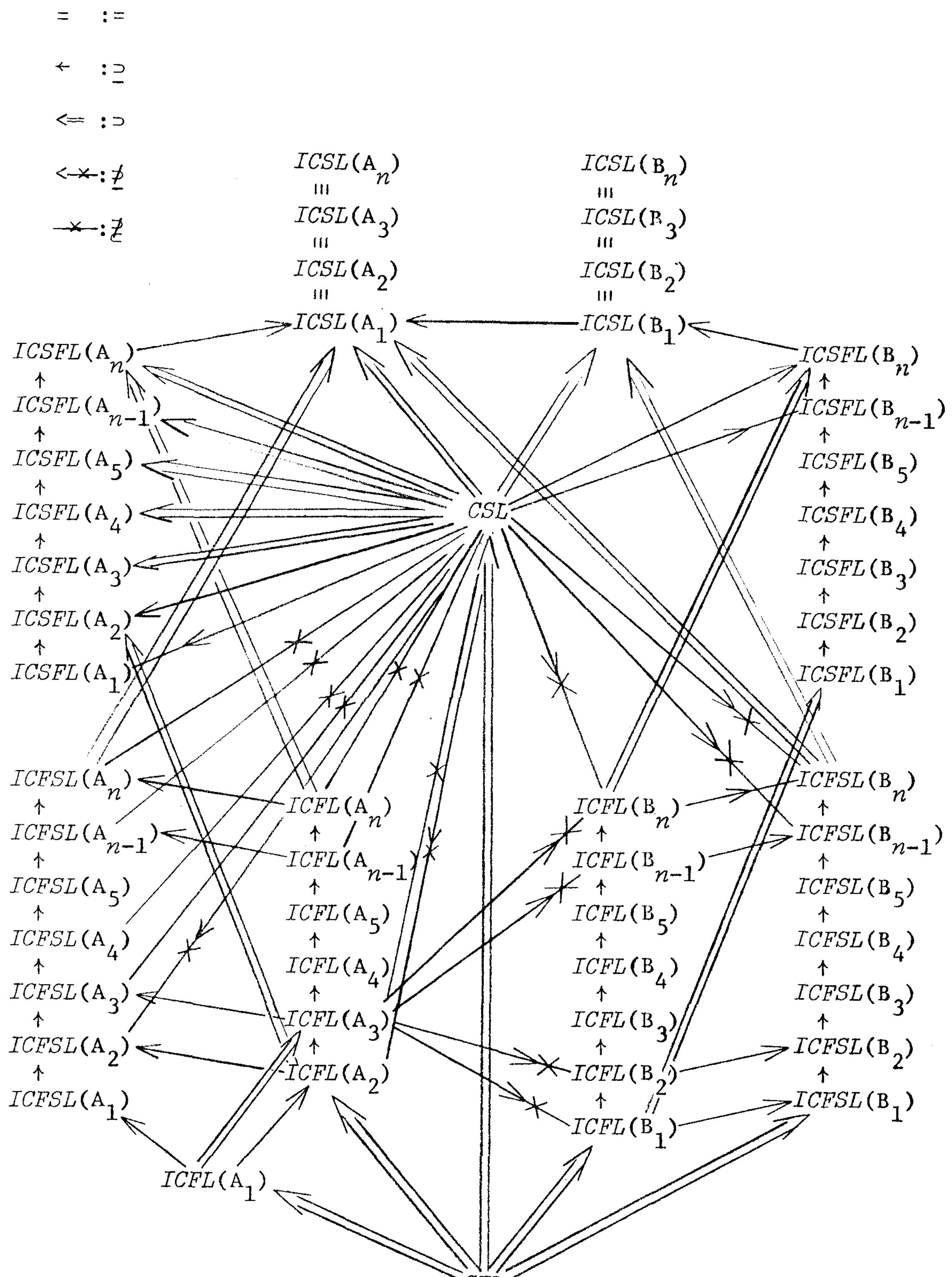


Fig. 2.10 The hierarchy of the abilities of interactive systems.

model which represents no interaction.

For the future, we will investigate relations between *ICFSL*(A_i) and *ICFL*(A_i), and between *ICFSL*(B_i) and *ICFL*(B_i). In this chapter we have shown the relations, but they have not been shown whether *ICF*(A_i) is equal to *ICFSL*(A_i) or not, and whether *ICF*(B_i) is equal to *ICFSL*(B_i) or not. And we will investigate relations between *ICFSL*(A_i) and *ICFSL*(A_{i+1}), and relations between *ICSFL*(A_i) and *ICSFL*(A_{i+1}), and etc.. Investigating those relations we may construct bases for comparing complexities of many problems.

In next chapter we will propose a new production system which is constructed by extending descriptive power of the interactive systems of mode B_i.

CHAPTER 3

Interactive Graph Production System

3.1 Introduction

Since production systems (PS) were first proposed by Post [34] as a general computational mechanism, the methodology has seen a great deal of development and has been applied to a diverse collection of problems. A production system may be viewed as consisting of three components: a set of rules, a data base, and an interpreter for the rules. In the simplest design, a rule is an ordered pair of symbol strings, with a left and right hand side, the rule set has a predetermind total ordering; and the data base is simply a collection of symbols.

Throughout much of the work reported, there appears to be two major views of PSs, as characterized on one hand by the psychological modelling efforts (PSG, PAS II, VIS, etc.) [27, 29] and on the other by the performance oriented, knowledge-based expert systems (e.g. MYCIN, DENDRAL) [13, 18]. For the psychological modellers, production rules offer a clear, formal, and powerful way of expressing basic symbol processing acts, which form the primitives of information processing psychology. For the designer of knowledge-based systems, production rules offer a representation of knowledge that is relatively easily accessed and modified, making it quite useful for systems designed for incremental apploaches to competence.

Now we have trend to apply PSs to more and more complex problems. Those problems need more complex knowledge-bases. For instance, the DENDRAL system uses a literal pattern match, but its patterns are graphs representing chemical classes. For expressing complex situations graphs are better than collections of literals for human understandability. In many cases graphs are used for describing situations, and we see many usages of graphs for explanations while collections of assertions are used for an internal representation of a system. So we want to construct a PS which treats not symbol strings or collections of assertions, but graphs.

In complex problems we find two situations interacting each other. For instance, in a problem solving for controlling a robot we find a robot and its environment interacting each other. We think that it is better than a description by a single situation which represents the robot and its environment, to describe the world by the set of two sub-situations: one represents the robot and the other does its environment; and changes of the world by interactive changes of two sub-sitautions. So we want to construct a PS which can express systems that interact each other also.

In this chapter we will propose a new graph production system that has interactions called Interactive Graph Production System (IGPS). IGPS represents a situation by a set of two labelled directed graphs and changes of situations by rewriting rules of graphs. So we will construct IGPS based on the graph grammar system which is discussed in chapter 2. In this chapter we will first describe definitions of IGPS, next show some examples of IGPSs for making obvious the method of descriptions and moves, and then discuss execution of IGPS.

3.2 Interactive Graph Production System

In this section we will describe definitions of *Interactive Graph Production System* (IGPS). An IGPS is constructed by two labelled directed graphs and two sets of production rules which control interactions and changing of situations.

3.2.1 Situations of IGPS

A situation of an IGPS is represented by a tuple,

$$(\sigma_0 , \sigma_1),$$

where, σ_0 and σ_1 are sub-situations which are described by labelled directed graphs. More formally, a sub-situation σ_i is represented by 3-tuple,

$$(N_i , L_i , E_i),$$

where, N_i is a set of nodes, L_i is a function: $N_i \rightarrow$ a set of labels; and E_i is a set of edges and is included by $N_i \times N_i$.

[Example 3.1] Here we show an example of a sub-situation.

Let $N_i = \{ 1, 2, 3 \}$,

$L_i(1) = \text{he}$,

$L_i(2) = \text{is}$,

$L_i(3)$ = diligent ,

and $E_i = \{ (1, 2), (2, 3) \}$.

Then a sub-situation (N_i, L_i, E_i) is the graph of Fig. 3.1.

3.2.2 Structure of an Interactive Graph Production System

An IGPS is a 7-tuple:

$$S = (C, V, R, i_0, i_1, P_0, P_1),$$

where, C is a set of labels of sub-situations, V is a set of variables whose ranges are sub-sets of C, R is a function: $V \rightarrow 2^C$; which defines the ranges of variables, i_0 and i_1 are two finite initial sub-situations whose sets of labels are C, and P_0 and P_1 are sets of productions which are applied to two sub-situations: σ_0 and σ_1 respectively.

In this thesis we will describe elements of C by strings of lower case latters and elements of R by strings of upper case letters.

3.2.3 Variables

An element of V is v_i, whose range is $R(v_i) \subset C$. The variable v_i can have a value of an element of $R(v_i)$.

[Example 3.2] Here we show an example of a variable.

Let C = { station, school, boy, girl },

$V \ni$ PLACE ,

and, R(PLACE) = { station, school }.

Then the value of 'PLACE' can be 'station' or 'school', and it can not be 'girl' or 'boy'.

3.2.4 Productions

We describe a production of P_0 or P_1 by the form:

$$(\ G_1\ )\qquad G_2 \equiv\!> G_3\ ,$$

where, G_1 is a lebelled directed graph whose $|N_1|$ is a non-negative integer and the set of labels is $C \cup V$, G_2 is a labelled directed graph whose $|N_2|$ is a natural number and the set of labels is $C \cup V$, and G_3 is a labelled directed graph whose set of labels is $C \cup V \cup \{\text{null}\}$ and satisfies conditions listed below.

$$N_3 \supset N_2 \quad \text{and} \quad E_3 \supset E_2\ ,$$

and $$\text{any } n_3 \in N_3 : L_3(n_3) \in V\ ;$$

there exists n_2 or n_1 : $n_2 \in N_2$ and $L_2(n_2) = L_3(n_3)$,

and/or $$n_1 \in N_1 \text{ and } L_1(n_1) = L_3(n_3).$$

At an application of a production each variable has one value. When two labelled directed graphs are compared, a variable v and a constant c are compatible if the value of v is c.

[Example 3.3] We show an example of a production in Fig. 3.2. In this example C, V and R of the IGPS are same as those in Example 3.2.

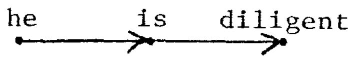


Fig. 3.1 An example of a sub-situation of IGPS.

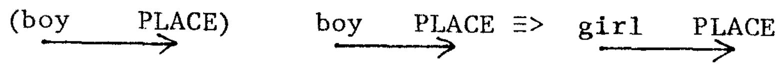


Fig. 3.2 An example of a production of IGPS.

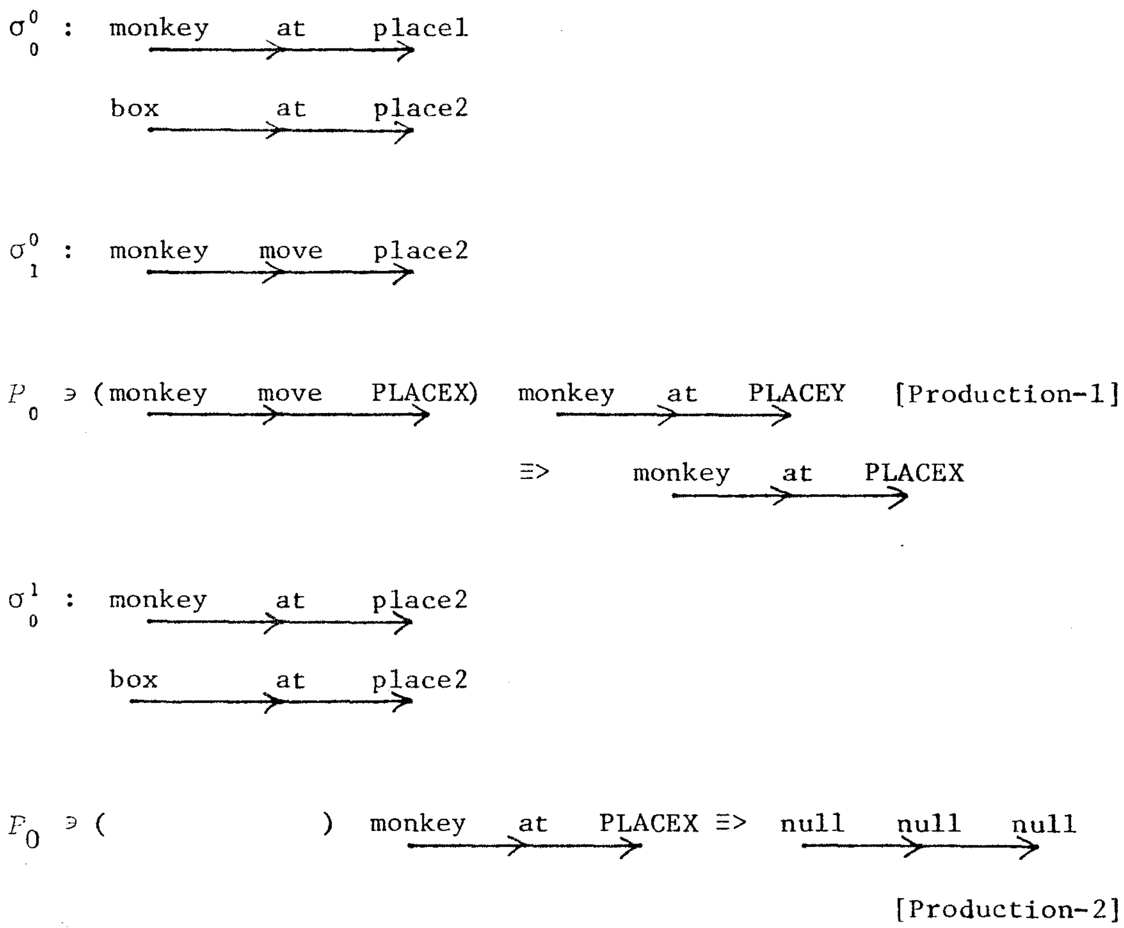


Fig. 3.3 Examples of productions and their applications.

3.2.5 Effect of an application of a production

Let a situation of an IGPS be (σ_0, σ_1), and let a production p_{ij}: $(G_1)\ G_2 \Rrightarrow G_3$; be included in P_i (i=0, 1). Then the production p_{ij} may be applied, if under two conditions are satisfied.

[Condition 3.1] Let σ be σ_{1-i} , and let G be G_1, then Condition 3.1' is satisfied. If G is an empty graph, then this condition is satisfied.

[Condition 3.1']

Let, $\sigma = (N , L , E)$,

then there exists $N' \subset N$: $(N', L, E \cap N' \times N')$ and G are compatible.

When Condition 3.1 is satisfied, the production p_{ij} can be applied if Condition 3.2 is satisfied.

[Condition 3.2] Let G_2' be a graph which is made by rewriting labels of G_2 which are elements of V and used in G_1, to the value of the variable. Let σ be σ_i and G be G_2', then Condition 3.1' is satisfied.

If a production which satisfies Condition 3.1 and Condition 3.2 is applied, then a sub-graph of σ_i which matches G_2 is rewritten to G_3'. Here G_3' is a graph which is made by rewriting labels which are elements of V to the value of the label.

In an IGPS, productions have structure which is described in 3.2.4, so deleting of nodes or edge is not available. But in an IGPS the special label 'null' expresses that the node will be not rewritten nor referred. So an IGPS interpreter can delete a node whose label is null and edges which connect the node, there are no defferences in moves of the IGPS.

[Example 3.4] We show an example of a production and its application in Fig. 3.3. In this example,

$$C = \{ \text{monkey, at, place1, place2, box, move} \},$$

$$V = \{ \text{PLACEX, PLACEY} \},$$

and $$R(\text{PLACEX}) = R(\text{PLACEY}) = \{ \text{place1, place2} \}.$$

When a situation of the IGPS is (σ_0^0, σ_1^0) of Fig. 3.3, if production-1 of Fig. 3.3 is an element of P_0 of the IGPS, then in production-1 PLACEX's value is place2 and PLACEY's value is place1, and the production satisfies Condition 3.1 and Condition 3.2, so the production can be applied to σ_0^0. If the production is applied to σ_0^0, then a situation of the IGPS becomes to be (σ_0^1, σ_1^0) of Fig. 3.3. And production-2 of Fig. 3.3 has the same effect of deleting of the graph.

3.2.6 Moves of an IGPS

One move of an IGPS is constructed with two sub-moves. When a situation of an IGPS is (σ_0^0, σ_1^0), one sub-move is an application of an element of P_0 to σ_0^0, or preservation of σ_0^0 when no elements of P_0 can not be applied to σ_0^0. Let the result be (σ_0^1, σ_1^0). Next one sub-move is an application of an element of P_1 to σ_1^0, or preservation of σ_1^0 when no elements of P_1 can not be applied to σ_1^0. Let the result be (σ_0^1, σ_1^1).

3.3 Some examples of IGPS

In this section we show some examples of IGPS for making obvious the structure of an IGPS and the moves of an IGPS.

3.3.1 The three coin problem

Here we show the three coin problem of Jackson [21].

Problem. Given three coins initially HHT (i.e., heads, heads, tails), in exactly three moves make all coins show the same face. A move consists of flipping a coin over.

We show below elements of an IGPS which describes the three coin problem and chunks of knowledge for solving the problem.

C = { start, cont, flip, end, coin1, coin2, coin3, inc, count0, count1, count2, count3, head, tail, op, counter },

V = { COINX, COINY, COINZ, CONT, STATEX, STATEY, COUNTALL, COUNTNEXT, STATE, COUNT },

R(COINX) = R(COINY) = R(COINZ) = { coin1, coin2, coin3 },

R(CONT) = { cont, end },

R(STATEX) = R(STATEY) = R(STATE) = { head, tail },

R(COUNTALL) = { count0, count1, count2, count3 },

R(COUNTNEXT) = { count1, count2, count3 },

R(COUNT) = { count0, count1 },

i_0 = start $\rightarrow$ cont ,

i_1 = inc → count0 → count1   inc → count1 → count2

inc → count2 → count3   head ← op → tail

coin1 → head   coin2 → head   coin3 → tail

counter → count0 .

In Figs. 3.4 and 3.5, P_0 and P_1 are shown.

In this IGPS the sub-situation σ_1 which is initially i_1 expresses a situation of coins, and the sub-situation σ_0 which is initially i_0 expresses a situation of a process of solving. Productions of P_0 generate moves which fit for the situation with referring a sub-situation σ_1 by G_1 of a rule for getting knowledges of coins' situation and how many times the move is. A production of P_1 expresses the move which is generated by P_0. In i_1 "inc → count0 → count1" expresses the move of a counter. And "head ← op → tail" expresses that 'head' and 'tail' are opposite faces of each other.

In Table 3.1, we show changes of a sub-situation σ_0 of the IGPS. And in Table 3.2, changes of a sub-situation σ_1 of the IGPS are shown. In Table 3.2 we omit parts which do not change for making short.

3.3.2 Monkey/banana problem

The familiar monkey and banana problem is formulated as an IGPS. In three coin problem, that IGPS always generates correct answers, but here, the IGPS expresses a process of monkey's trial and error process. In the IGPS, monkey does not want to do a move which can not be carried out. And if monkey can take banana, he must take it. We show elements of the IGPS below.

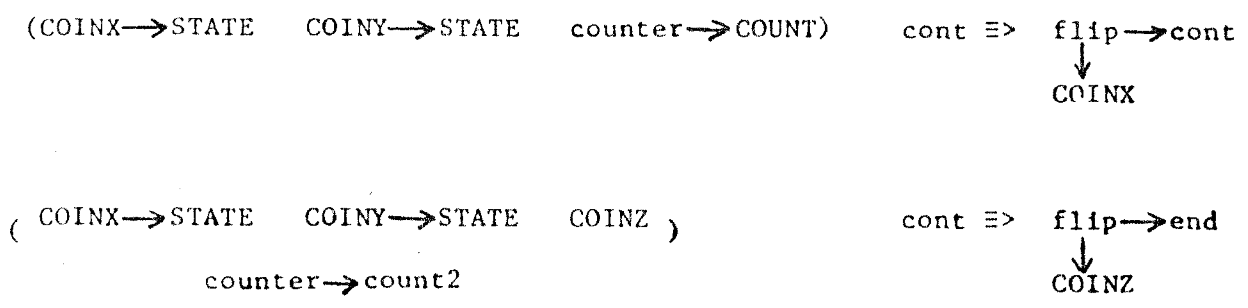


Fig. 3.4 P_0 of the IGPS which represents the three coin problem.

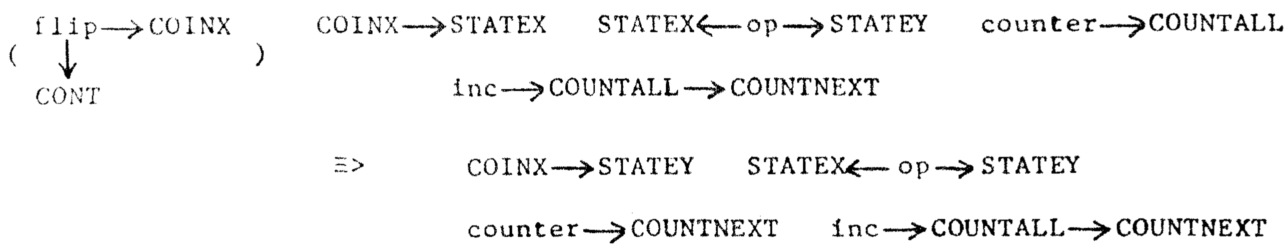


Fig. 3.5 P_1 of the IGPS which represents the three coin problem.

Table 3.1 Process of changing of σ_0.

applied production	σ_0 which is the result of application of the production
production-1	start → flip → cont ↓ coin1
production-1	start → flip → flip → cont ↓ ↓ coin1 coin1
production-2	start → flip → flip → flip → end ↓ ↓ ↓ coin1 coin1 coin3

Table 3.2 Process of changing of σ_1.

value of 'COINX'	σ_1 which is the result of application of the production		
coin1	coin1 → tail	coin2 → head	coin3 → tail
coin1	coin1 → head	coin2 → head	coin3 → tail
coin3	coin1 → head	coin2 → head	coin3 → head

C = { mky, box, ban, place1, place2, place3, at, over, on, has, move, push, climb, take, start, cont, cont', end },

V = { PLACEX, PLACEY, DO, DOX },

R(PLACEX) = R(PLACEY) = { place1, place2, place3 },

R(DO) = { climb, take, push, move, start },

R(DOX) = { push, move }.

i_0 and i_1 of the IGPS are shown in Table 3.3 and Table 3.4 respectively. P_0 and P_1 of the IGPS are shown in Fig. 3.6 and Fig. 3.7 respectively. We show sub-situations σ_0 and σ_1 in Table 3.3 and Table 3.4 respectively. A process of monkey's trial and error is shown in Table 3.3 by productions which can be applied simultaneously. In this IGPS, how to select a production is not represented, so selection of a production is a problem for an IGPS interpreter. Monkey who is expressed in this IGPS does trial and error, but he can recognize what move is not necessary. This ability is represented by production VI in Fig. 3.8.

3.3.3 Block world manipulations problem

As a final example, we demonstrate that an IGPS can describe the block world problem studied by Tate [40]. The intent is to describe the stacking and unstacking of cubic blocks of uniform size on a flat surface such as a table, and a process of the solution.

In this problem the robot can do only one kind of manipulation which is shown in Fig. 3.8. The IGPS which will be shown below de-

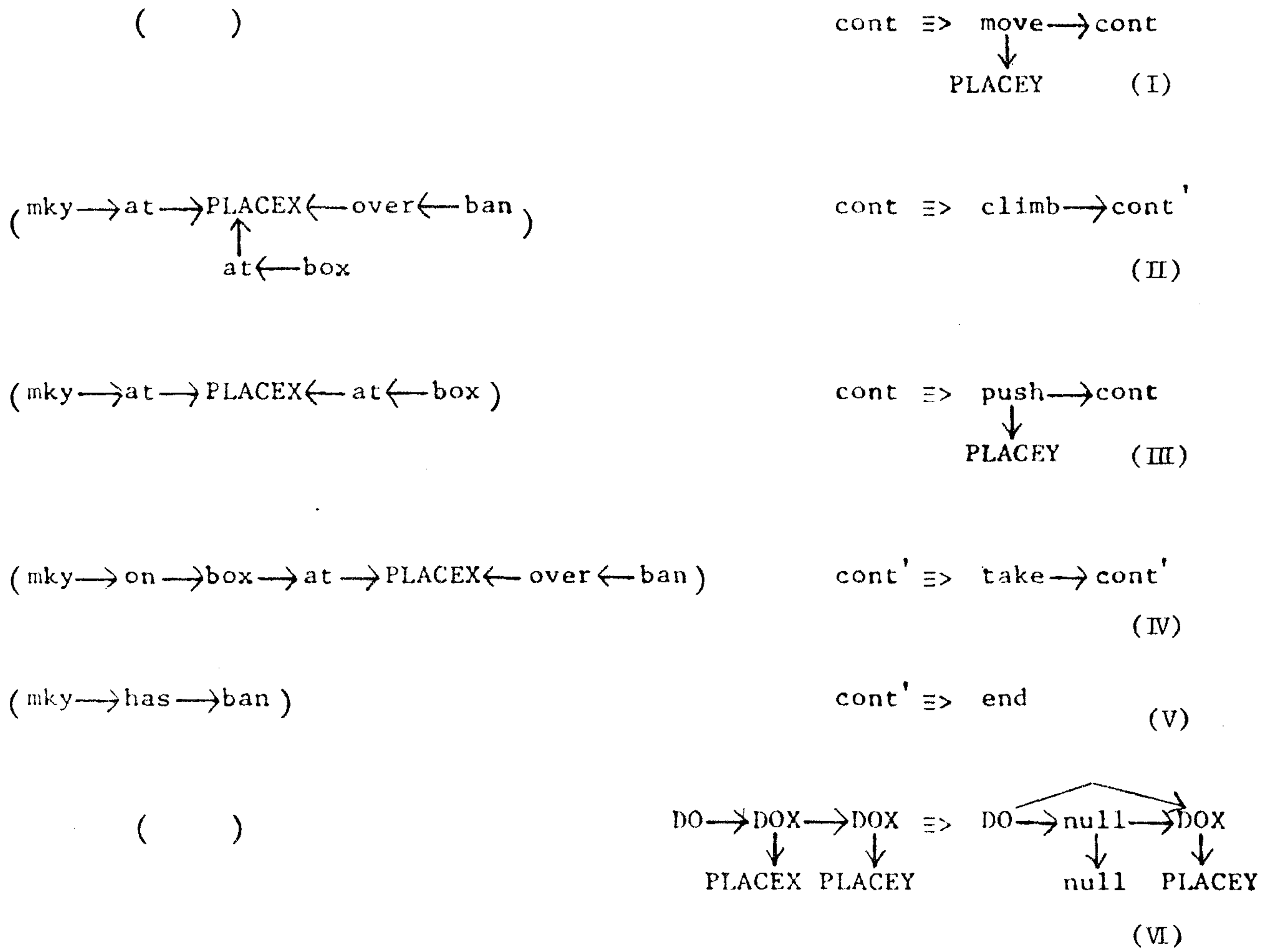


Fig. 3.6 P_0 of the IGPS which represents the monkey and banana problem.

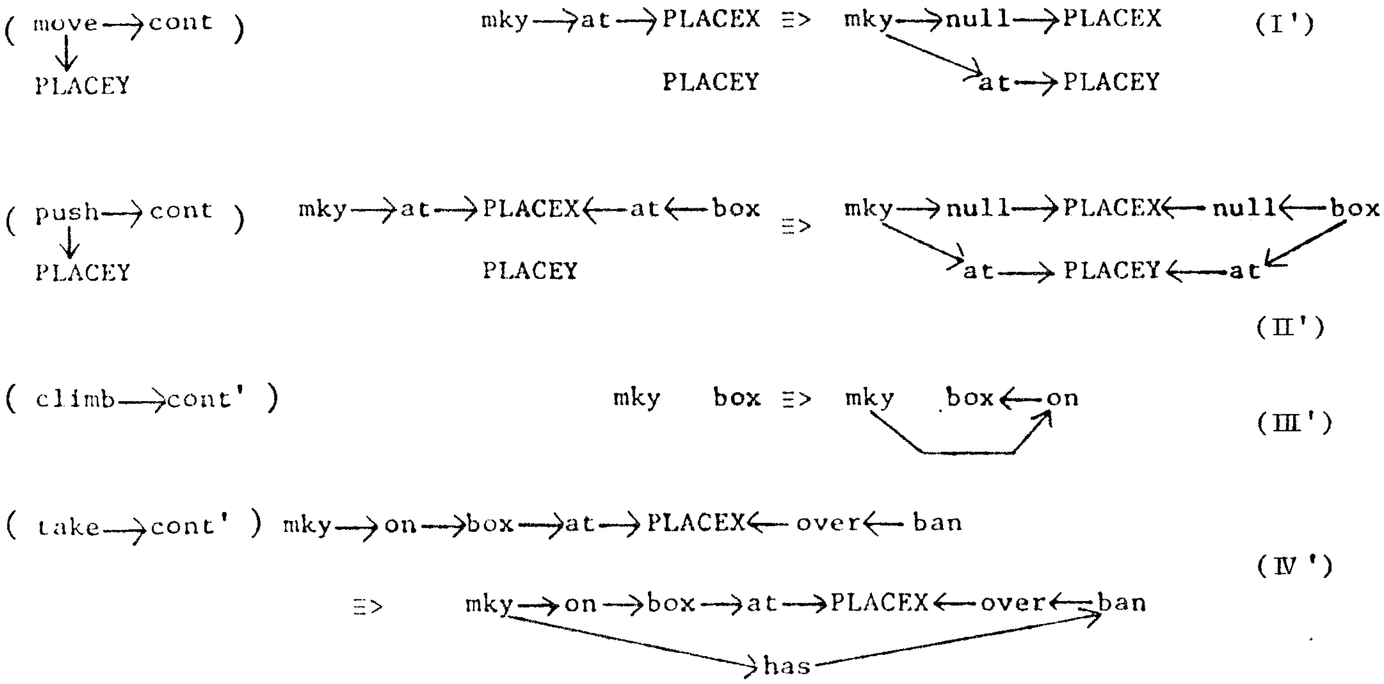


Fig. 3.7 P_1 of the IĜPS which represents the monkey and banana problem.

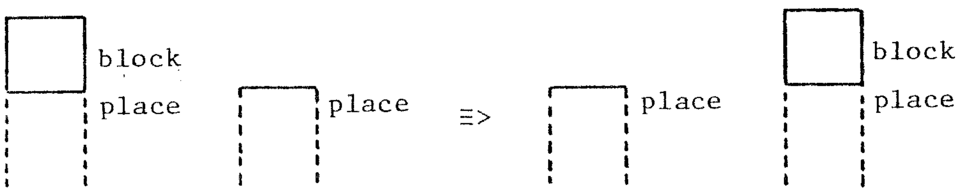
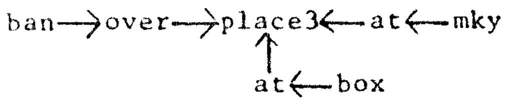
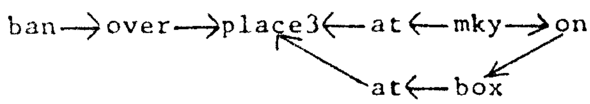
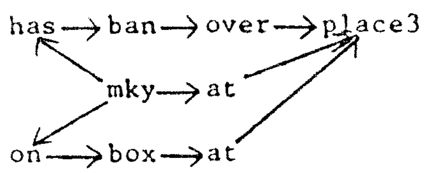


Fig. 3.8 The manipulation in the block world.

Table 3.3 Process of changing of σ^0.

applied production	σ_0 which is the result of application of the production
	start → cont
I	start → move → cont; move ↓ place3
I	start → move → move → cont; move ↓ place3, move ↓ place2
VI (I)	start → null → move → cont (arc start → move); null ↓ null, move ↓ place2
III (I)	start → move → push → cont; move ↓ place2, push ↓ place1
III (I)	start → move → push → push → cont; move ↓ place2, push ↓ place1, push ↓ place3
VI (I, III)	start → move → null → push → cont (arc move → push); move ↓ place2, null ↓ null, push ↓ place3
II (I, III)	start → move → push → climb → cont; move ↓ place2, push ↓ place3
IV	start → move → push → climb → take → cont; move ↓ place2, push ↓ place3
V (IV)	start → move → push → climb → take → end; move ↓ place2, push ↓ place3

Table 3.4 Process of changing of σ_1.

applied production	σ_1 which is the resul of application of the production
	mky→at→place1 box→at→place2 ban→over→place3
I'	mky→at→place3←over←ban box→at→place2 place1
I'	ban→over→place3 mky→at→place2←at←box place1
II'	ban→over→place3 mky→at→place1←at←box place2
II'	 place1 place2
III'	 place1 place2
IV'	 place1 place2

scribes a world where there are four blocks and three places where the robot can stack the blocks. But the IGPS can easily describe n blocks and m places problem by a little alteration. We show the IGPS which describes the block world problem below.

C = { blocka, blockb, blockc, blockd, floor1, floor2, floor3, on, start, start', cont, end, tgoal, tgoal*, ct, nct, clean, scont, send, put },

V = { PLACEX, PLACEY, PLACEZ, PLACEW, BLOCKX, BLOCKY, BLOCKZ, BLOCKX },

R(PLACEX) = R(PLACEY) = R(PLACEZ) = R(PLACEW)
= { floor1, floor2, floor3, blocka, blockb, blockc, blockd },

R(BLOCKX) = R(BLOCKY) = R(BLOCKZ)
= { blocka, blockb, blockc, blockd },

R(FLOORX) = { floor1, floor2, floor3 }.

And i_0 and i_1 of the IGPS are shown in Table 3.5 and Table 3.6 respectively, and P_0 and P_1 of the IGPS are shown in Fig. 3.9 and Fig. 3.10 respectively.

We show the initial situation and the final situation of the block world problem in Fig. 3.11. The final situation is the objective of this problem. In this IGPS the initial situation of the block world is described in i_1, and the final situation is described in i_0. In this IGPS " $X \leftarrow$ on $\leftarrow Y$ " expresses that Y is on X.

Next we explain productions of P_0 of this IGPS. We show pictorial representation of situations of G_1 of P_0 of the IGPS which represents the block world problem for making obvious the situations where each production is permitted to apply in Fig. 3.12. Production 1 of P_0 decides the first sub-goal. Production 2 decides that the sub-goal is attained. Production 3 decides a new sub-goal when an old sub-goal is attained. Production 4 generates a necessary move for attaining a sub-goal. Productions 5 and 6 generate necessary conditions for reaching a sub-goal when the sub-goal can not be attained directly. Production 7 confirms that the necessary conditions which are generated by productions 5 and 6 are enough. Production 8 generates necessary conditions which are needed for satisfying the necessary conditions generated yet when the conditions can not be satisfied directly. Productions 9 and 12 generate moves for satisfying the necessary conditions generated yet. Production 10 deletes the necessary conditions which are satisfied by a side effect of an application of other productions. Production 11 confirms that all necessary conditions for application of production 4 are satisfied.

The production of P_1 in Fig. 3.10 expresses the only one move of the robot in the block world, which is the transference of a block which is clear-top.

We show sequences of sub-situation σ_0 and σ_1 in Table 3.6 and Table 3.7 respectively. We show the productions which can be applied but are not applied in the column of applied production by parenthesized.

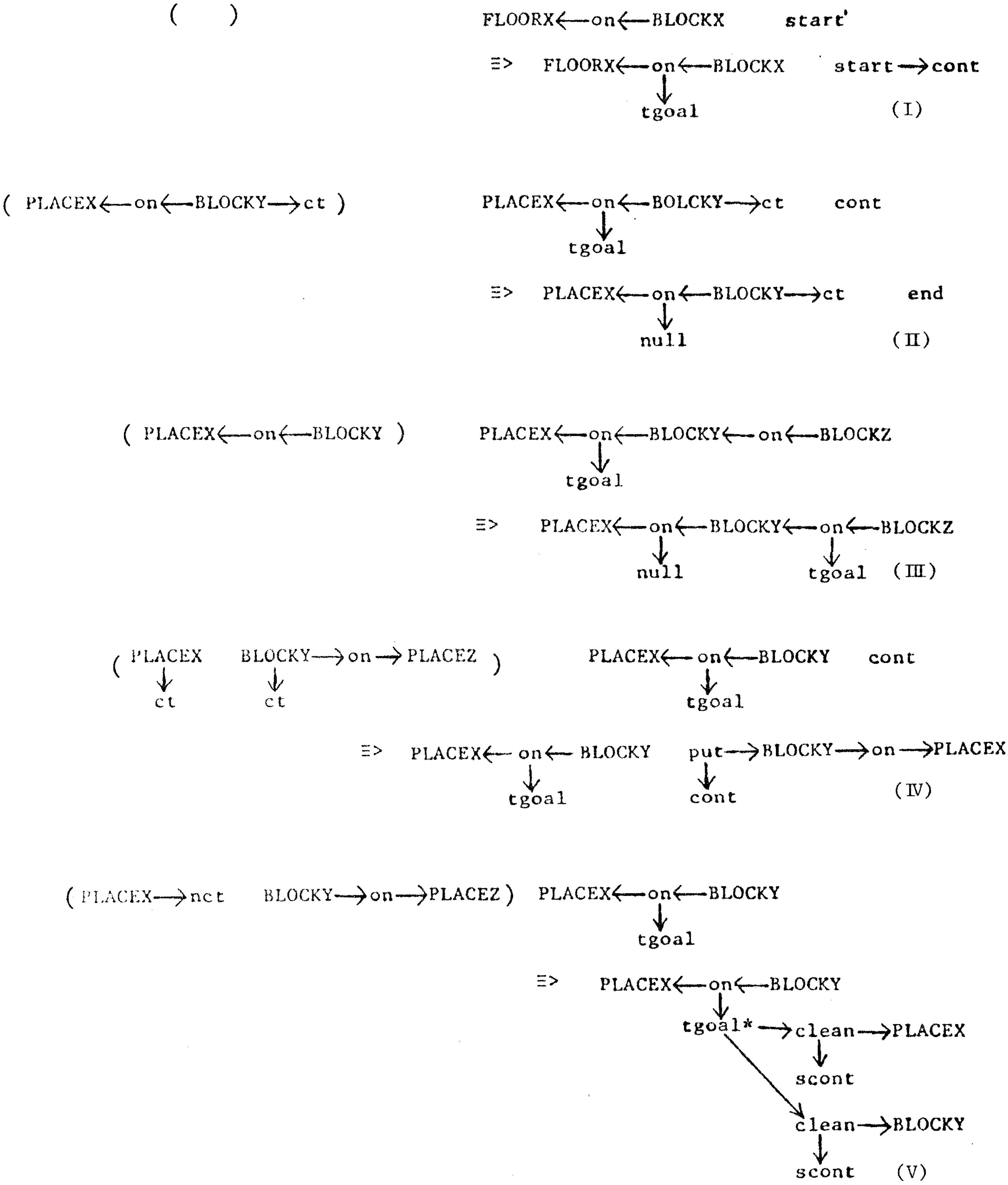


Fig. 3.9 P_0 of the IGPS which represents the block manipulations problem. (partial)

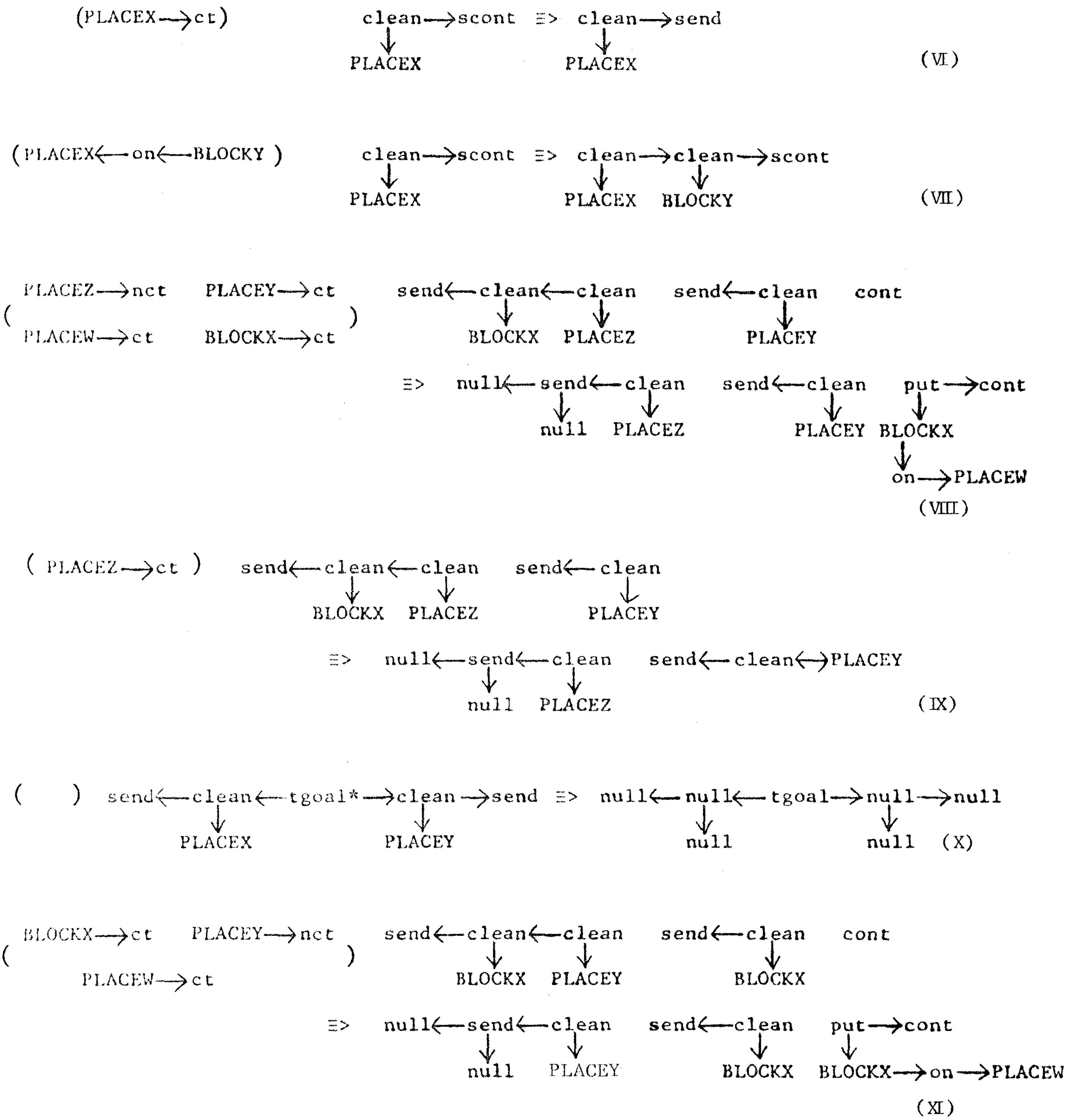


Fig. 3.9 Continued.

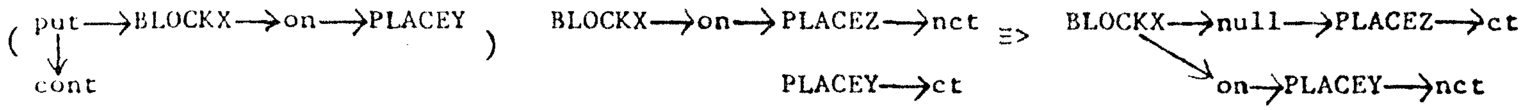
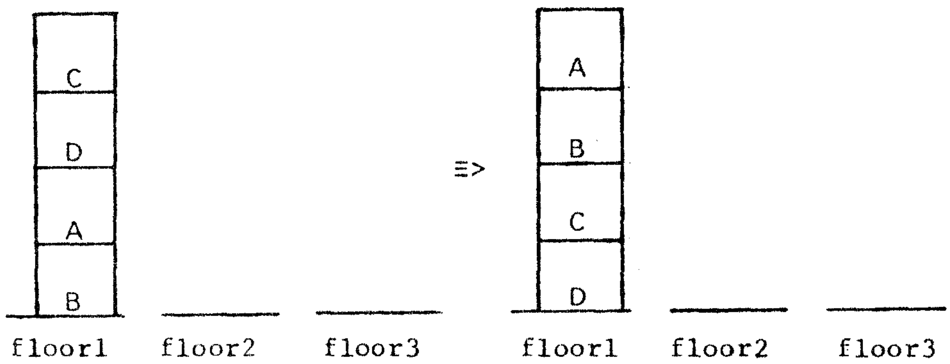


Fig. 3.10 P_1 of the IGPS which represents the block manipulations problem.



The initial situation The final situation

Fig. 3.11 The initial situation and the final situation of the block world manipulations problem.

Table 3.5 Process of changing of σ_0 of the IGPS which represents the block world problem.

applied production	σ_0 which is the result of application of the production
i_0	floorl←on←blockd←on←blockc←on←blockb←on←blocka→ct start→cont
I	floorl←on←blockd←on←blockc←on←blockb←on←blocka→ct on↓tgoal start→cont
V or VI	floorl←on←blockd←on←blockc←on←blockb←on←blocka→ct on↓tgoal*→clean→scont clean↓floorl tgoal*↘clean→scont clean↓blockd start→cont
VIII	floorl←on←blockd←on←blockc←on←blockb←on←blocka→ct on↓tgoal*→clean→clean→scont clean↓floorl, clean↓blockb tgoal*↘clean→scont clean↓blockd start→cont

continue

VIII

```
floorl<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
          |
          v
        tgoal*-->clean-->clean-->clean--> scont
           \       |       |       |
            \      v       v       v
             \   floorl  blockb  blocka
              \
               v
                 clean-->scont
                   |
                   v
                 blockd

start-->cont
```

VIII

```
floorl<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
          |
          v
        tgoal*-->clean-->clean-->clean-->clean-->scont
           \       |       |       |       |
            \      v       v       v       v
             \   floorl  blockb  blocka  blockd
              \
               v
                 clean-->scont
                   |
                   v
                 blockd

start-->cont
```

VIII

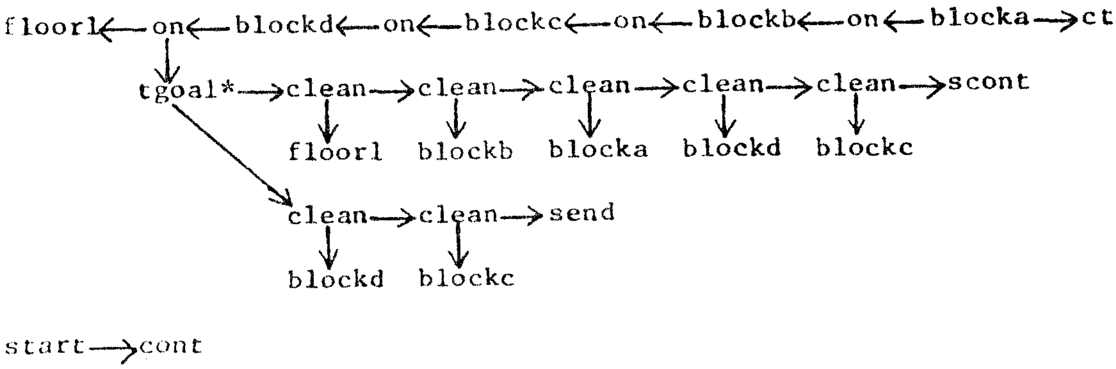
```
floorl<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
          |
          v
        tgoal*-->clean-->clean-->clean-->clean-->scont
           \       |       |       |       |
            \      v       v       v       v
             \   floorl  blockb  blocka  blockd
              \
               v
                 clean-->clean-->scont
                   |       |
                   v       v
                 blockd  blockc

start-->cont
```

VI (VIII)

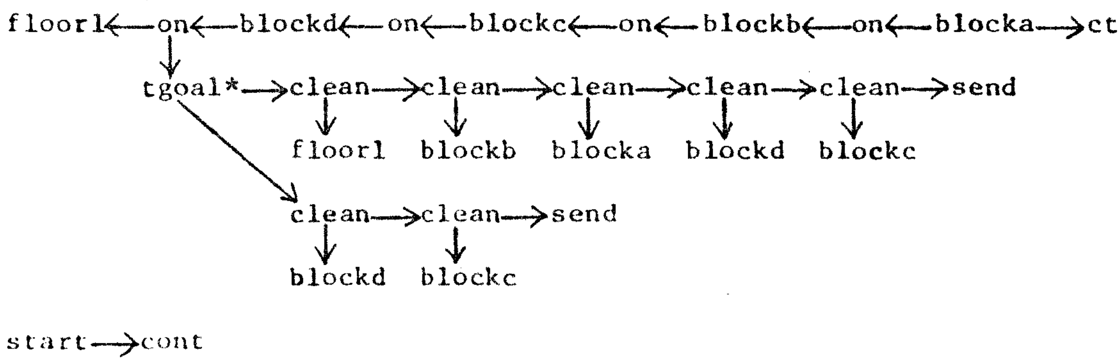
```
floorl<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
          |
          v
        tgoal*-->clean-->clean-->clean-->clean--> scont
           \       |       |       |       |
            \      v       v       v       v
             \   floorl  blockb  blocka  blockd
              \
               v
                 clean-->clean-->send
                   |       |
                   v       v
                 blockd  blockc

start-->cont
```

continue

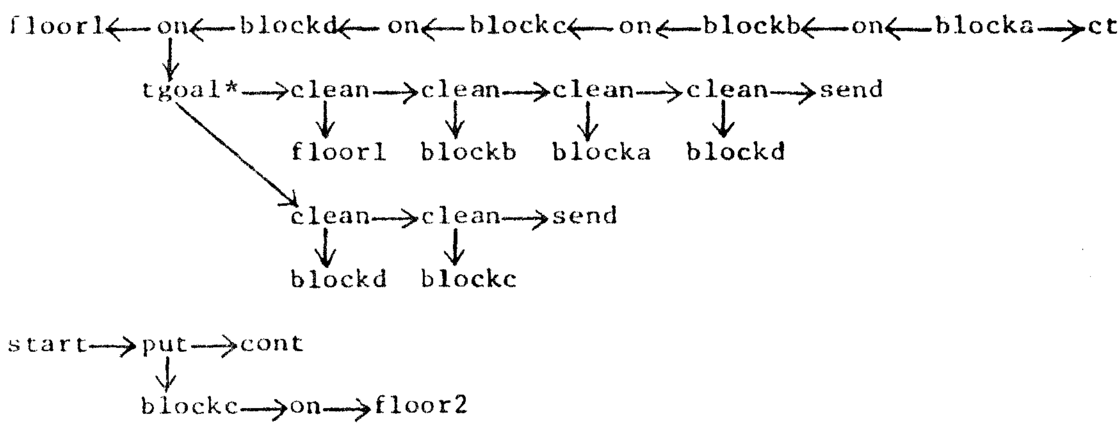
VIII



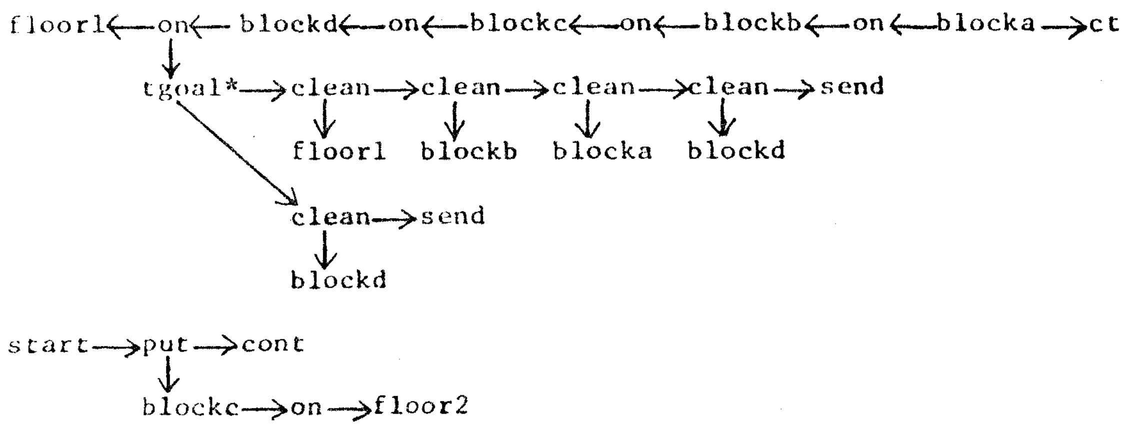
VII



XI

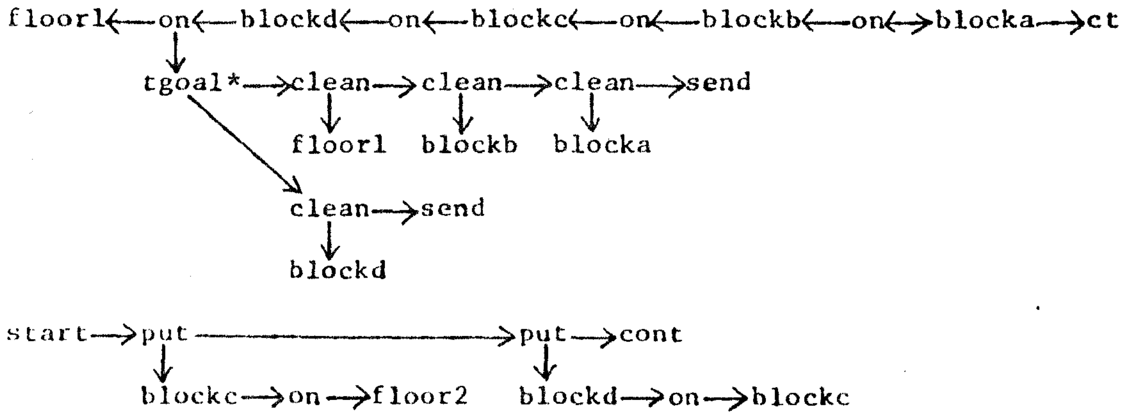


X

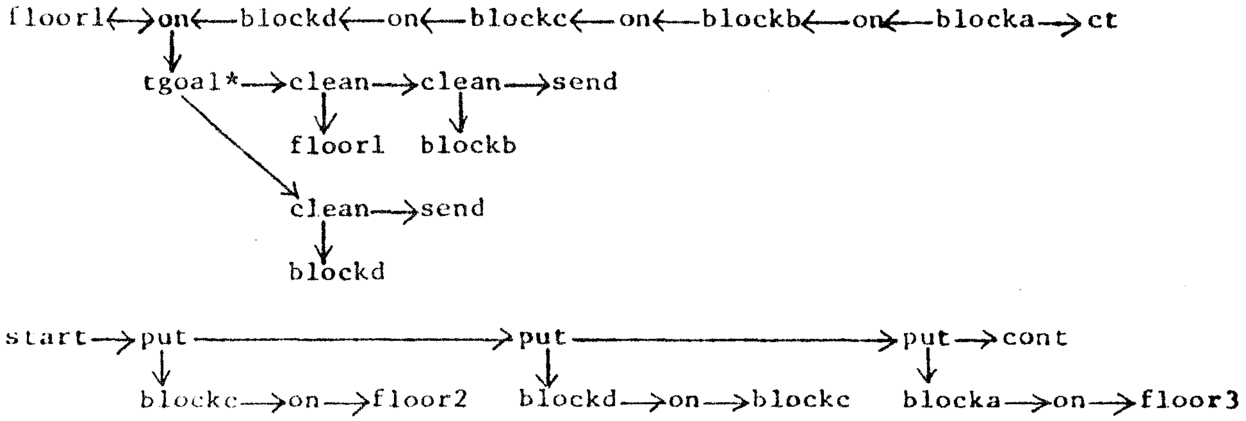


continue

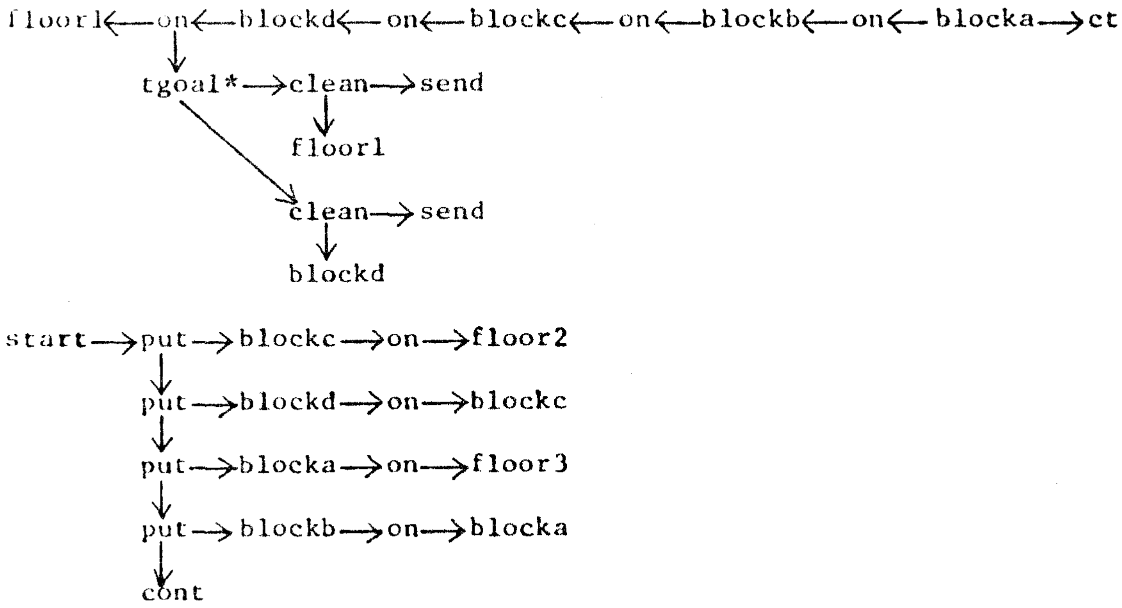
XI



IX



IX



continue

X

```
floor1<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
           |
           v
         tgoal

start-->put-->blockc-->on-->floor2
         |
         v
        put-->blockd-->on-->blockc
         |
         v
        put-->blocka-->on-->floor3
         |
         v
        put-->blockb-->on-->blocka
         |
         v
        cont
```

IV

```
floor1<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
           |
           v
         tgoal

start-->put-->blockc-->on-->floor2
         |
         v
        put-->blockd-->on-->blockc
         |
         v
        put-->blocka-->on-->floor3
         |
         v
        put-->blockb-->on-->blocka
         |
         v
        put-->blockd-->on-->floor1
         |
         v
        cont
```

III

```
floor1<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
                         |
                         v
                       tgoal

start-->put-->blockc-->on-->floor2
         |
         v
        put-->blockd-->on-->blockc
         |
         v
        put-->blocka-->on-->floor3
         |
         v
        put-->blockb-->on-->blocka
         |
         v
        put-->blockd-->on-->floor1
         |
         v
        cont
```

continue

IV

```
floor1<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
                        |
                        v
                      tgoal

start--> put-->blockc-->on-->floor2
          |
          v
         put-->blockd-->on-->blockc
          |
          v
         put-->blocka-->on-->floor3
          |
          v
         put-->blockb-->on-->blocka
          |
          v
         put-->blockd-->on-->floor1
          |
          v
         put-->blockc-->on-->blockd
          |
          v
         cont
```

III

```
floor1<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
                                      |
                                      v
                                    tgoal

start-->put-->blockc-->on-->floor2
         |
         v
        put-->blockd-->on-->blockc
         |
         v
        put-->blocka-->on-->floor3
         |
         v
        put-->blockb-->on-->blocka'
         |
         v
        put-->blockd-->on-->floor1
         |
         v
        put-->blockc-->on-->blockd
         |
         v
        cont
```

IV

```
floor1<--on<--blockd<--on<--blockc<--on<--blockb<--on<--blocka-->ct
                                      |
                                      v
                                    tgoal

start-->put-->blockc-->on-->floor2
         |
         v
        put-->blockd-->on-->blockc
         |
         v
        put-->blocka-->on-->floor3
         |
         v
        put--> blockb-->on-->blocka
         |
         v
        put-->blockd-->on-->floor1
         |
         v
        put--> blockc-->on--> blockd
         |
         v
        put-->blockb-->on-->blockc
         |
         v
        cont
```

continue

III

```
floor1 <-- on <-- blockd <-- on <-- blockc <-- on <-- blockb <-- on <-- blocka --> ct
                                                                 |
                                                                 v
                                                               tgoal

start --> put --> blockc --> on --> floor2
           |
           v
          put --> blockd --> on --> blockc
           |
           v
          put --> blocka --> on --> floor3
           |
           v
          put --> blockb --> on --> blocka
           |
           v
          put --> blockd --> on --> floor1
           |
           v
          put --> blockc --> on --> blockd
           |
           v
          put --> blockb --> on --> blockc
           |
           v
          cont
```

IV

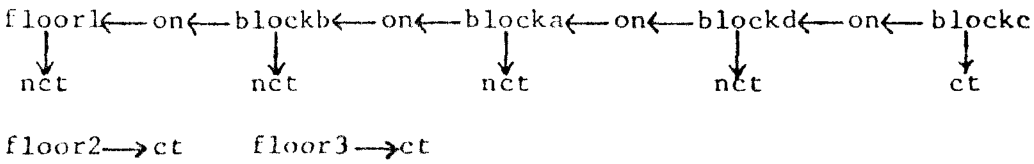
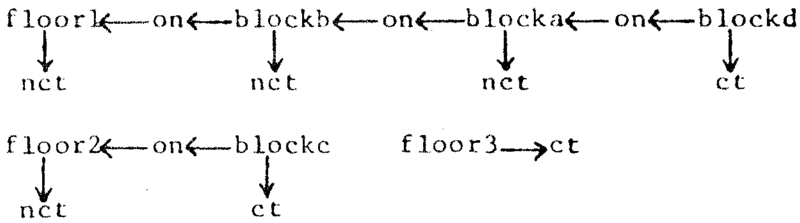
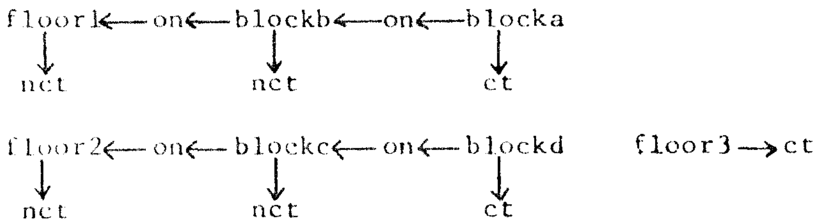
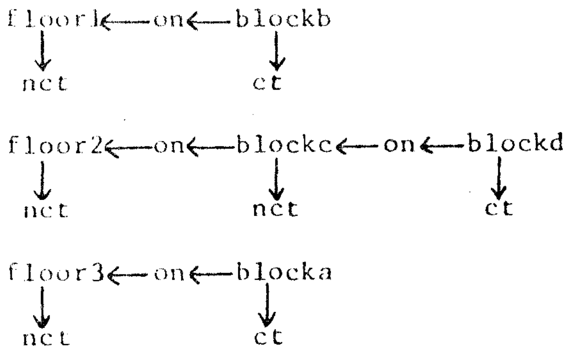
```
floor1 <-- on <-- blockd <-- on <-- blockc <-- on <-- blockb <-- on <-- blocka --> ct
                                                                 |
                                                                 v
                                                               tgoal

start --> put --> blockc --> on --> floor2
           |
           v
          put --> blockd --> on --> blockc
           |
           v
          put --> blocka --> on --> floor3
           |
           v
          put --> blockb --> on --> blocka
           |
           v
          put --> blockd --> on --> floor1
           |
           v
          put --> blockc --> on --> blockd
           |
           v
          put --> blockb --> on --> blockc
           |
           v
          put --> blocka --> on --> blockb
           |
           v
          cont
```

continue

II

```
floor1 ← on ← blockd ← on ← blockc ← on ← blockb ← on ← blocka → ct

start → put → blockc → on → floor2
         ↓
        put → blockd → on → blockc
         ↓
        put → blocka → on → floor3
         ↓
        put → blockb → on → blocka
         ↓
        put → blockd → on → floor1
         ↓
        put → blockc → on → blockd
         ↓
        put → blockb → on → blockc
         ↓
        put → blocka → on → blockb
         ↓
        end
```

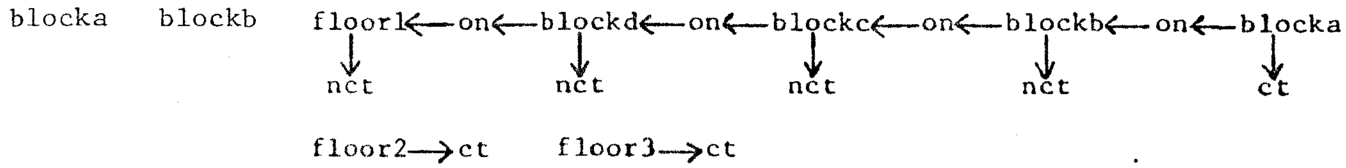
Table 3.6 Process of changing of σ_1 of the IGPS which represents the block world problem.

value of 'BLOCKX'	value of 'PLACEY'	σ_1 which is the result of application of the production
i_1		
blockc	floor2	
blockd	blockc	
blocka	floor3	

continue

```
blockb   blocka   floor1 ──→ ct

                  floor2 ←── on ←── blockc ←── on ←── blockd
                    ↓                 ↓                 ↓
                   nct               nct                ct

                  floor3 ←── on ←── blocka ←── on ←── blockb
                    ↓                 ↓                 ↓
                   nct               nct                ct


blockd   floor1   floor1 ←── on ←── blockd
                    ↓                 ↓
                   nct                ct

                  floor2 ←── on ←── blockc
                    ↓                 ↓
                   nct                ct

                  floor3 ←── on ←── blocka ←── on ←── blockb
                    ↓                 ↓                 ↓
                   nct               nct                ct


blockc   blockd   floor1 ←── on ←── blockd ←── on ←── blockc
                    ↓                 ↓                 ↓
                   nct               nct                ct

                  floor2 ──→ ct

                  floor3 ←── on ←── blocka ←── on ←── blockb
                    ↓                 ↓                 ↓
                   nct               nct                ct


blockb   blockc   floor1 ←── on ←── blockd ←── on ←── blockc ←── on ←── blockb
                    ↓                 ↓                 ↓                 ↓
                   nct               nct               nct                ct

                  floor2 ──→ ct

                  floor3 ←── on ←── blocka
                    ↓                 ↓
                   nct                ct
```

continue



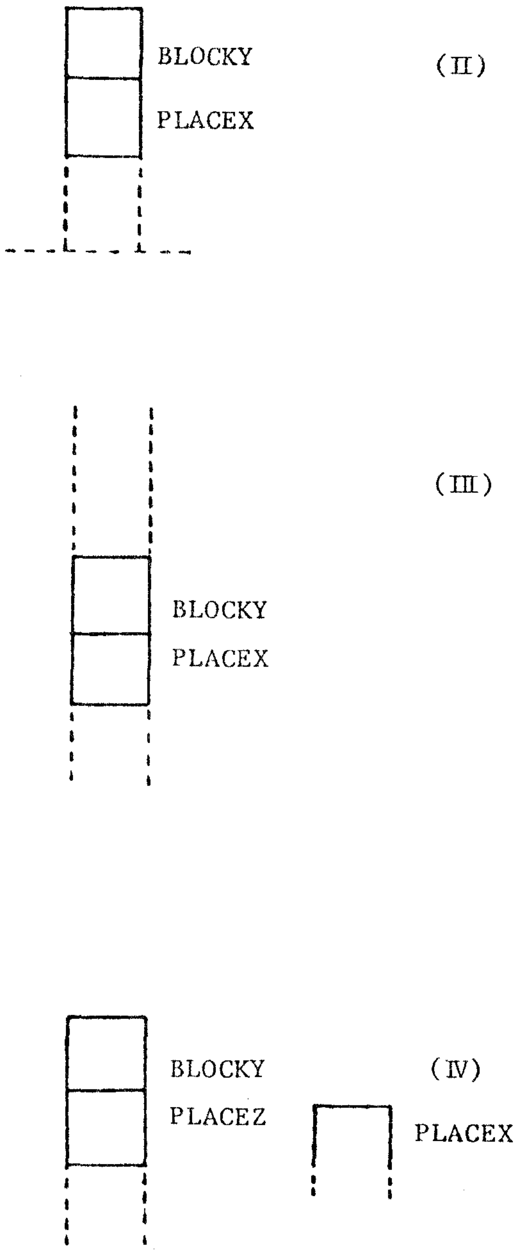


Fig. 3.12 Pictorial representation of situations of G_1 of P_0 which represents the block manipulations problem. (partial)

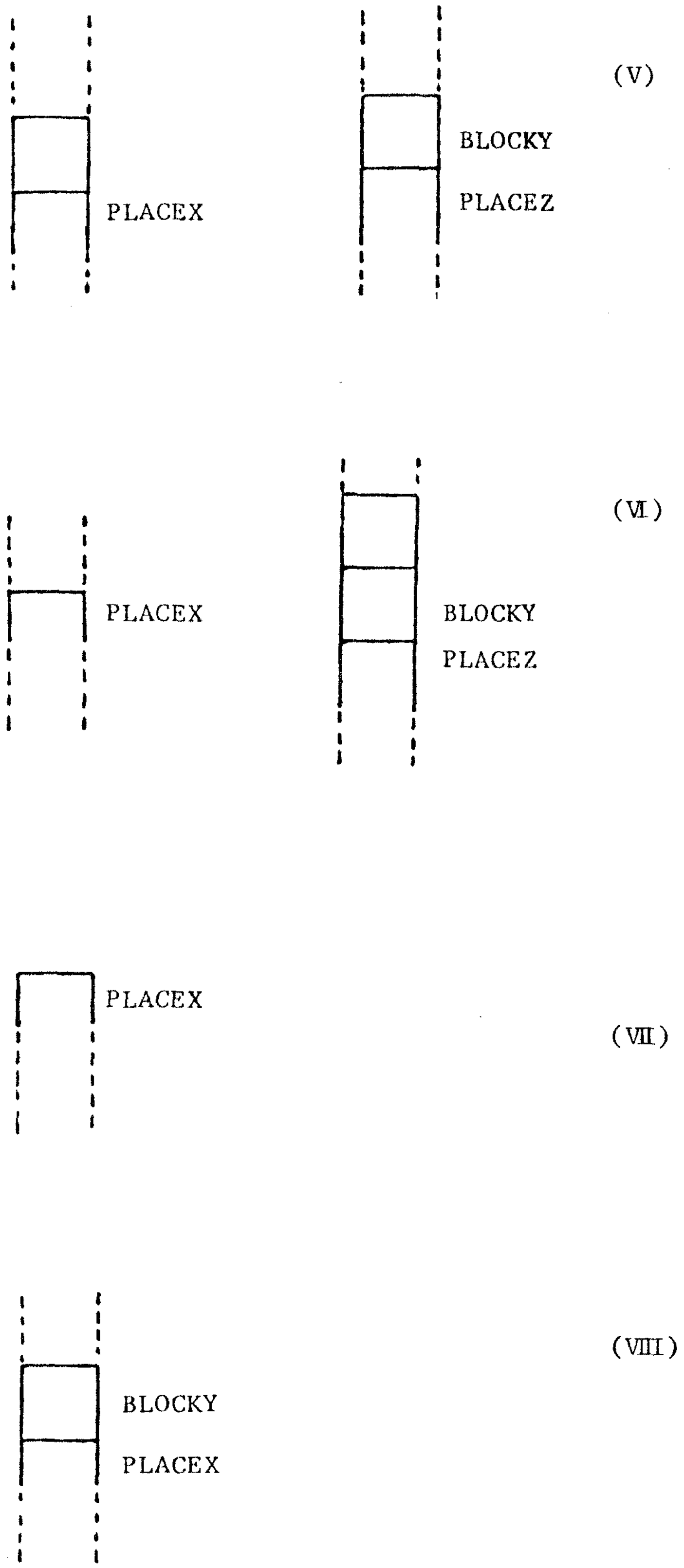


Fig. 3.12 Continued.

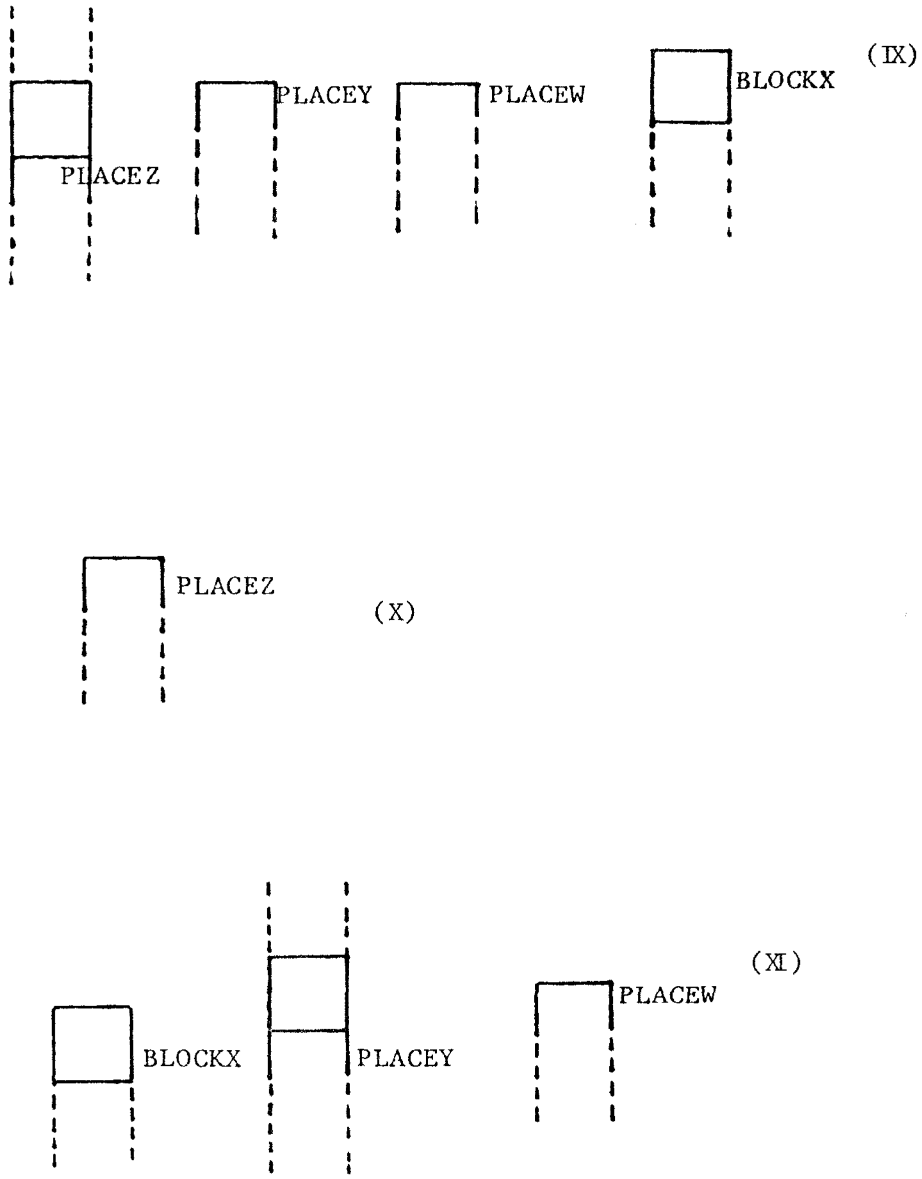


Fig. 3.12 Continued.

3.4 Execution of Interactive Graph Production System

In this section we discuss about execution of IGPS. The core of execution of IGPS is an IGPS interpreter, which receives an IGPS and execute the IGPS. And we need a production editing system for easy description of productions. The production editing system enables us to edit productions on a graphic display unit and generate set of productions.

3.4.1 IGPS interpreter

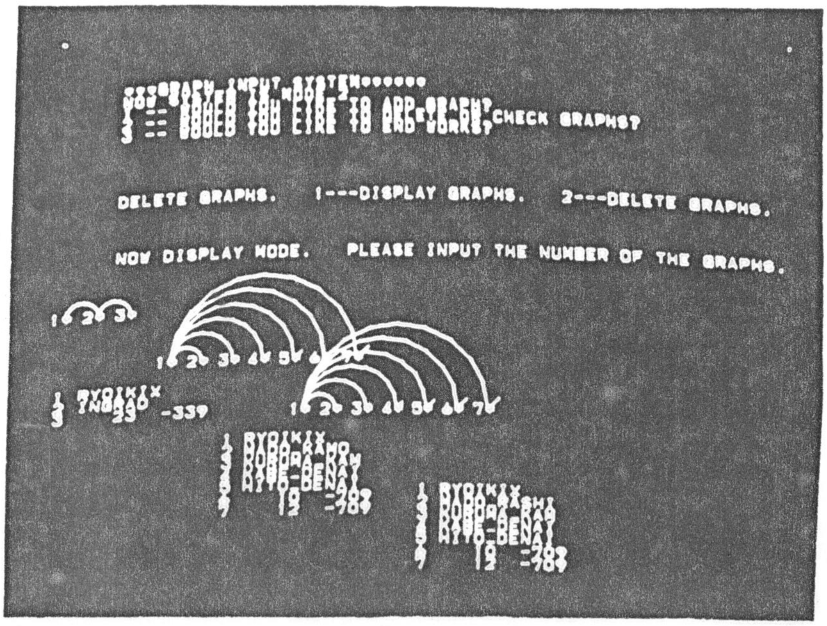
We implement an IGPS interpreter which executes an IGPS in accordance with the definitions of IGPS basically. But the IGPS interpreter has some fuctions for making processing fast and descriptions of production easy.

The IGPS interpreter has two modes except a basic mode. In the basic mode the IGPS interpreter applies one production at one time according to the definitions of IGPS. In expanded mode 1, the IGPS interpreter tries to apply all productions which are permitted to apply at one time. And in expanded mode 2, the IGPS interpreter tries to apply all productions to all sub-graphs at one time while productions can be applied. In those expanded modes, the number of decisions about permission of application of a production decreases at one application of a production, so processing time decreases at one application of a production.

In the definitions of IGPS we did not define how to select a production, so we must decide how to select a production here. In most of production system, for instance RPS [43], productions are ordered and the first production which matches the data-base is applied. But if we use rule ordering for selection of a production, we can not describe the monkey and banana problem as in 3.3.2. So the IGPS interpreter enables us to select a method of selection of a production from three methods. The first method is the conventional rule ordering. And the second one enables us to specify a priority of productions at each move. And the last one enables us to specify an algorithm which specifies a priority of productions at each move.

3.4.2 Production editing system

A production of IGPS is constructed by three tuples of labelled directed graphs. For editing productions efficiently, we need some functions, which enables us to define a set of constants C, a set of variables V and a range function R, and to input labelled directed graphs, and to check inputted productions. Input of labelled directed graphs can be done by inputting a label of each node and a tuple of a head and a tail of each edge using punched cards, but we can not inspect graphs efficiently using a list of labels of each node and a list of tuples of a head and a tail of each edge. So the production editing system must enable us to edit productions: three tuple of labelled directed graphs, using graphical description. Therefore the production editing system uses a graphic display unit for display of productions. And the production editing system enables us to edit productions interactively. We show a displayed image of the production editing system in Fig. 3.13.



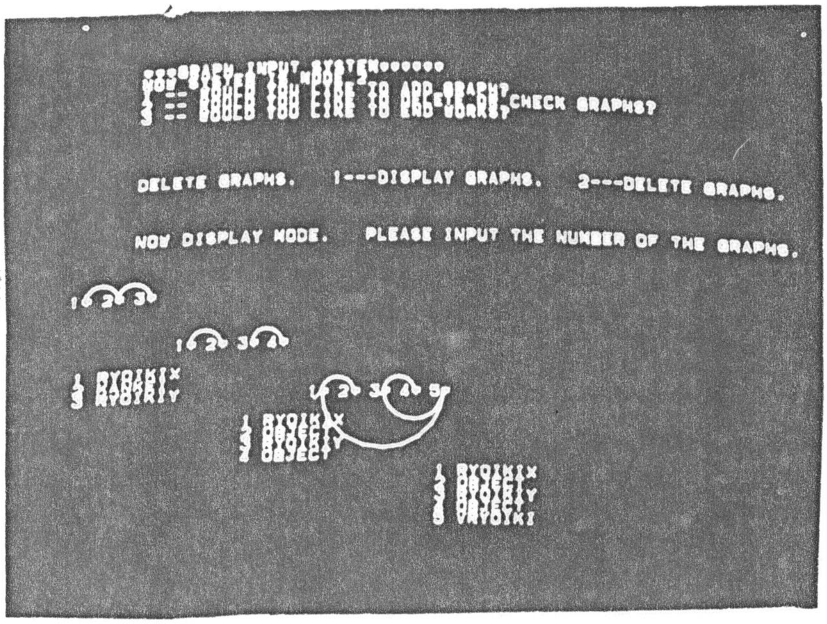


Fig. 3.13 Displayed images of the production editing system.

3.5 Conclusion

In this chapter, we have proposed a new production system: IGPS. IGPS is a formal system which expresses a structure and moves of a system which has interaction. And we have shown some examples of IGPSs for making obvious descriptive power and moves of IGPS.

IGPS has been developed by extending descriptive power of the interactive system of mode B by introducing variables. And we have shown three examples of IGPSs: the three coin problem, the monkey and banana problem, and the block world manipulations problem.

In the three coin problem, changing of situations is described in a data base. In the monkey and banana problem, monkey's try and error process is represented. And in the block world manipulations problem, complex situations are represented.

We have implemented the IGPS interpreter and the production editting system. They enable us to execute an IGPS easily.

In the field of artificial intelligence, we treat a diverse collection of problems. Some of them have interaction in their situations. For expressing those problem we need a production system which can express interaction. Of course conventional production systems can express a system which has interaction. But simplicity, modularity and other good PS's features are lost. IGPS can express a system which has interaction while increasing good PS's features. And using labelled directed graphs, IGPS can express very complex situations.

In future, we will apply IGPS to a diverse collection of problems and make obvious the merit and the demerit of IGPS and implement a more powerful IGPS interpreter which enables to treat larger problems.

CHAPTER 4

Image Interpretation using Interactive Graph Production System

4.1 Introduction

Since the first electric computer was born, the domain of computer applications has been expanding together with advances in basic computer science. At present, much of the research lies in the areas of pattern recognition and artificial intelligence. Researchers in these fields attempt to endow the computer not only with the calculating ability suggested by its name, but also with the ability of a perceptive and intelligent [12, 22, 30, 35, 44, 45].

Studies on computer vision are typical of those in pattern recognition and artificial intelligence, since their aim is to construct image understanding systems which can perceive and interpret visual stimuli automatically. Fig. 4.1 gives an overview of a computerrized image understanding system. This system is formally separable into low-level vision and high-level vision components [7]. The former, which is, in a sense, non-semantic information processing system, consists of a sensor and digitizer, a preprocessor, a feature extractor, and a symbolic descriptor. The latter consists of an understander and the semantic information processing associated with it.

The research here described is concerned with high-level computer

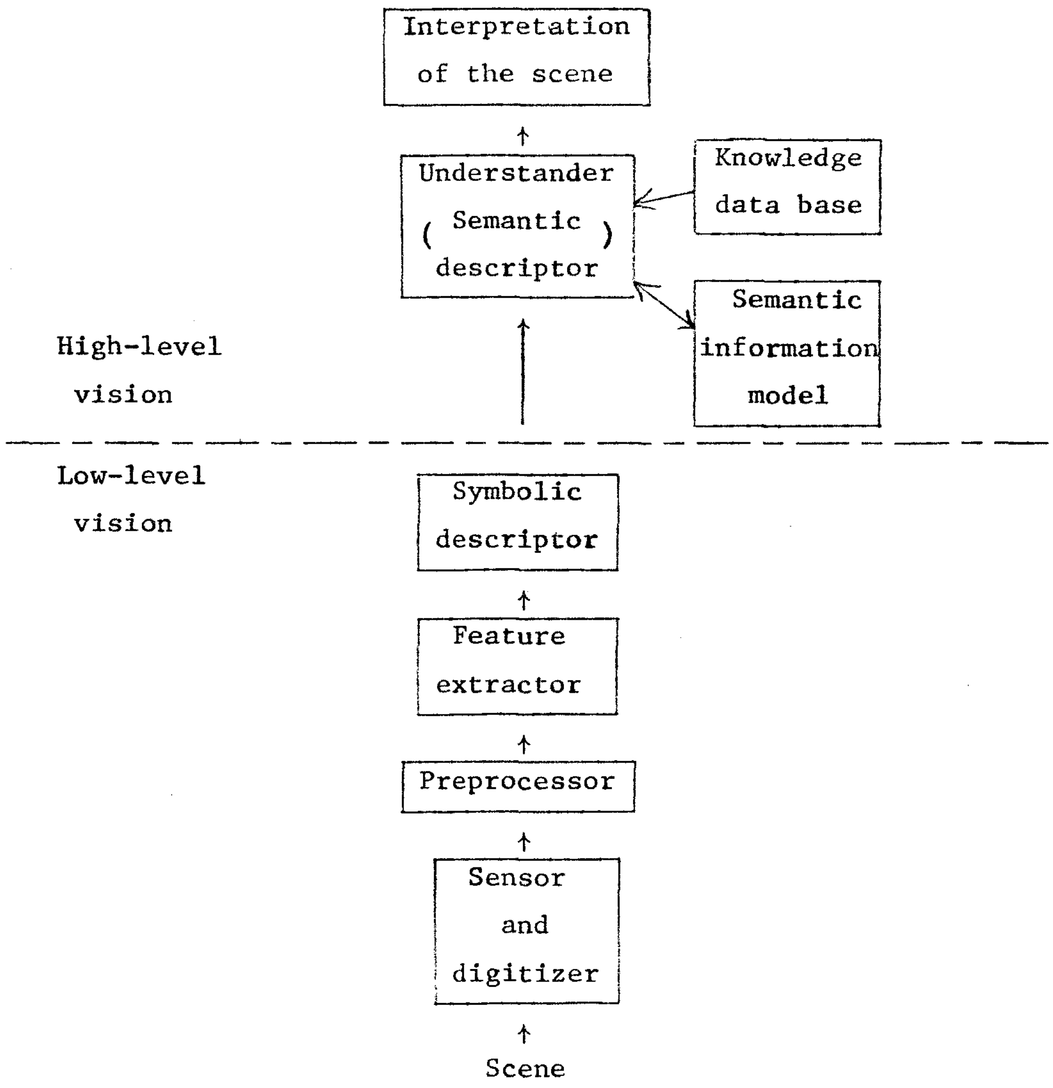


Fig. 4.1 Overview of an Image Understanding System.

vision and concentrated mainly on interpretation of images. To interpret an image, we need to describe the image and how to interpret the images. To describe a visual image graphs are better than literals. And we need easy method for describing the process of interpretation of an image. So we use Interactive Graph Production System to describe an image interpreting system. Using IGPS (Interactive Graph Production System) we can describe an image and its interpretation simultaneously, and we can describe relations between an image and its interpretation by interactive graph productions.

In the area of pattern recognition, syntactic method has been used where a pattern is described with simple sub-patterns and the structure of a pattern is described with complex of sub-patterns with grammar [10, 20, 25]. But in image analysis we can not hope that an image is described with simple sub-patterns in many cases. In image analysis segmentation to regions corresponds to division to sub-patterns in pattern recognition. But, now in many cases a region does not correspond to an object, but a uniform area. Some experiments of image interpretations have been done [11, 47]. For instance MSYS [11] handles a simple room scene which is segmented correctly. Most of experiments do not handle a complex out-door scene, where segmentation of a scene does not reflect the correct objects.

In this chapter first we will discuss description of an image and next description of semantics of images. Then we will discuss production rules which describe the relations between the description of an image and description of the semantics. Next we will discuss implementation of an image interpreting system. Then some experimental results will be shown.

4.2 Description of an image

Description of an image is realized with intra-region descriptions and inter-region relations. In this chapter, we will accept the three assumptions listed below.

[Assumption 4.1]

1. Two objects do not make one region.
2. One object can be divided into some regions.
3. An object which must be recognized is constructed with regions which are large enough for measuring their properties.

Assumptions 4.1.1 and 4.1.2 are related, and express that a region must be a part of an object. And Assumption 4.1.3 expresses that an object can be described.

One region is described with 11 properties listed below.

[Properties of a region]

1. Region number.
2. Center of gravity (described by X-distance and Y-distance).
3. Area (described the number of pixels).
4. Orientation.
5. Intensity (its average and dispersion).
6. Length of the border.
7. Compactness (contains two kinds of definitions).
8. Straight lines on the border (which described by its number

and orientation).

9. Limit of the existance (maxima and minima of X-distances and Y-distances of pixels in the region).

10. Straight lines in the region (described as 8).

11. Texture in the region‡.

These properties are expressed by the labelled directed graph as in Fig. 4.2. In Fig. 4.2 'value' expresses that the value of each properties is at the place, and 'type' expresses that the type of each properties is at the place.

Next we discuss description of inter-region relations. Among regions there are many relations, but only connections between regions are expressed in the image describing system. Most of other relations: is right of, is left of , is over of, etc., can be generated using properties of regions: the center of gravity or limit of existence. Connections between regions are expressed as in Fig. 4.3 by lengths of vorders of right and over. Fig 4.4 is the example of regions expressed as in Fig. 4.3.

Note: Compactnesses are defined as follows.

$$\text{Compactness-1} = 4\pi S \;/\; L^2$$

$$\text{Compactness-2} = 4S \;/\; \pi l^2,$$

where S is area, L is length of border and l is max(l') where $l'$is the length of the paths which pass through the center of gravity in the region.

‡ This feature is discussed in [24].

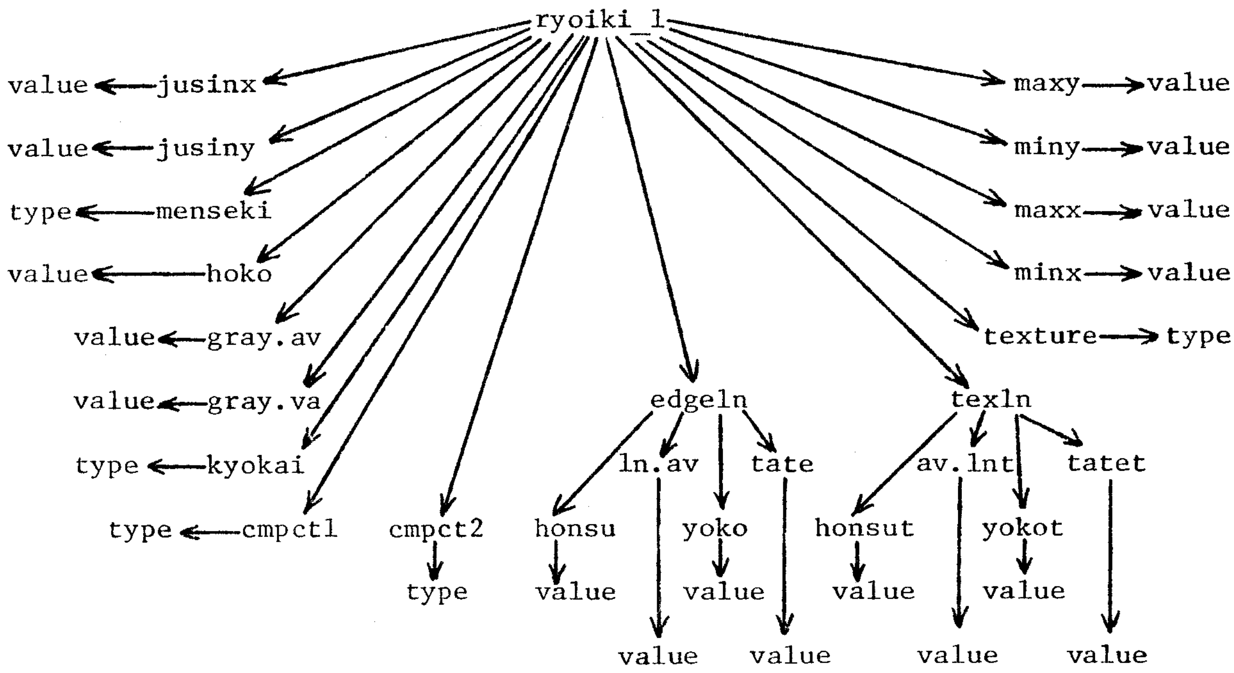


Fig. 4.2 Structure of graphs which represent features of a region.

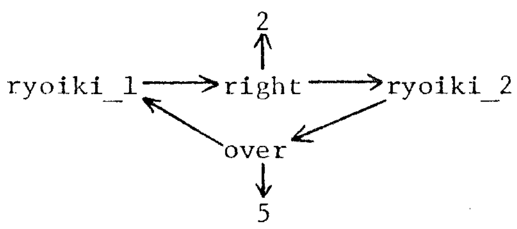


Fig. 4.3 An example of representation of inter-region relations.

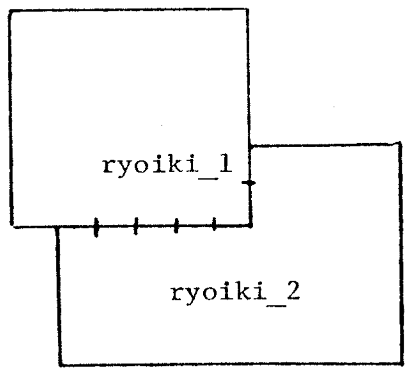
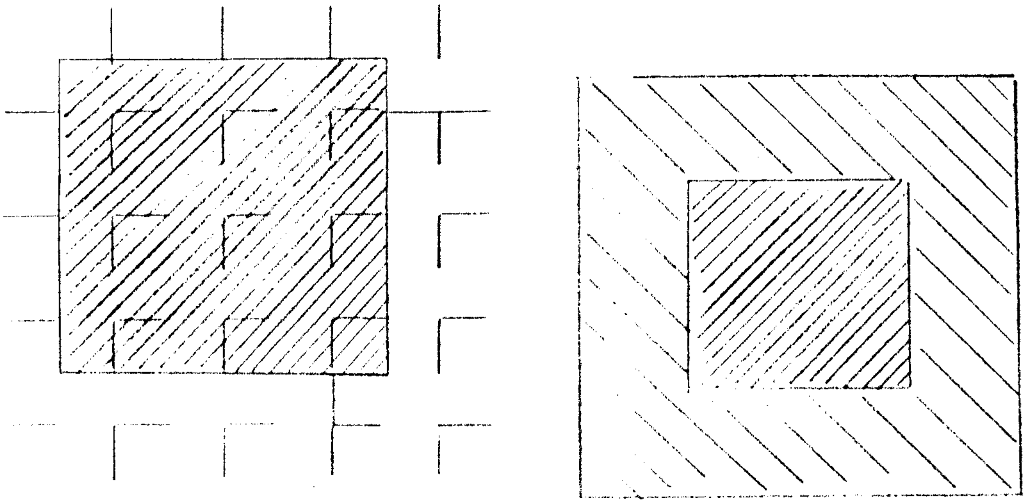


Fig. 4.4 An example of regions which have relations expressed as in Fig. 4.3.

4.3 Description of semantics of an image.

Description of semantics of an image depends on process of interpretation. In this chapter we handle simple out-door scenes where are five kinds of objects: a window, a wall, a man, a car and a road. Of course, we handle a real out-door scenes, so in a scene there may be other kinds of objects: trees, weeds, sky and etc.. And there can be a region which is one object and also other kind of object. This fact conflicts with Assumption 4.1. But the region is one edge between two object as in Fig. 4.5. So image interpreting system can interpret the region to be one of the two objects. We call those regions edges. So here we have 6-kinds of objects.

The labelled directed graph which expresses interpretation of an image is initially a collection of the graph of Fig. 4.6, and the label '*object*?' is rewritten to '*object*', '*object*-rashi', '*object*-kamo' or '*object*-denai' in accordance with inferences. And if two regions are infered to be parts of an object, then the label 'vryoiki', is used for expressing that those regions construct an object as in Fig. 4.7. The label 'mado' expresses that the region is a part of a window, the label 'mado-rashi' does that the region is probably a part of a window, the label 'mado-kamo' does that the region can be a part of a window, and the label 'mado-denai' does that the region can not be a part of a window. And the label 'kabe' expresses that the region is a part of a wall, the label 'hito' does that the region is a part of a man, the label 'kuruma' does that the region is a part of an automobile, the label 'michi' does that the region is a part of a road, and so on. The label 'fuchi' expresses that the region is an edge of objects.



(a) An original image.

(b) The digitalized image of (a).

Fig. 4.5 An example of an edge.

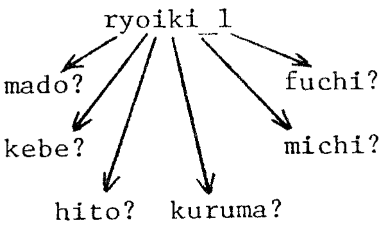


Fig. 4.6 An example of the initial sub-situation i_0.

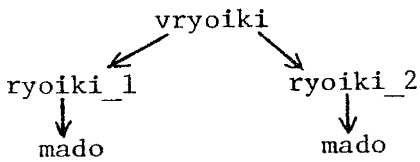


Fig. 4.7 An example of representation of an object.

4.4 Productions for interpretation of an image

Interactive Graph Production System is used for describing the process of interpretation of an image. An IGPS is defined as

$$S = (C, V, R, i_0, i_1, P_0, P_1)$$

where C is a set of constants which is defined in accordance with the structures of graphs describing an image and its interpretation, V is a set of variables and R is a function which defines the ranges of variables that are defined in accordance with the structures of productions, i_0 and i_1 are initial situations of graphs, the former expresses interpretation of an image and the latter describes an image.

Next we will discuss the set of productions which represent the process of interpretation. In this chapter a process of an image interpretation is divided into three parts. First at each region and each kinds of object, one of '*object*', '*object*-rashi', '*object*-kamo' and '*object*-denai' is selected in accordance with the probability of the region of the object. Next, inference is done using the result of former inference. And last, inference is done using relations among regions, and what kinds of objects the regions are decided.

4.4.1 Object-wise inference

At each kind of objects, features of regions are known using *Interactive Image Analizing System* [9]. In accordance with those features we construct the productions which decide the degree of belonging to each kind of objects. Here we use 84 productions. A part of which are collected using Interactive Image Analizing System is shown in Table 4.1 and Table 4.2. Table 4.1 shows averages of intensisty levels, are and compuctness of each kinds of objects. At each kinds of objects one of *'object'*, *'object*-rashi', *'object*-kamo' and *'object*-denai' is given to each region. 5 kinds of objects and one special object are treated in our image world, so each region takes 6 interpretations. Examples of productions which relate a window will be shown.

The production of Fig. 4.8 expresses that the region whose features satisfies the condition expressed in parenthisized graph is decided to be a window (mado). In figures, (m, n) expresses a internal expression of IGPS interpreter. That is used for expressing a constant or a variable which is not defined previously for its few usage. The label 'RYOIKIX' is defined as a variable whose range is the set of region numbers. And the labels 'hoko', 'egdeln' and etc. that are strings of lower-case letters are defined as constants previously. Next in Fig. 4.9 we show the production which expresses that the region can not be a window (mado_denai). Then in Fig. 4.10 we show the production which expresses that the region may be a window (mado_rashi). And we show the production which gives a region the interpretation 'mado_kamo' (the region may be a window) in Fig. 4.11. To each region, one of *'object'*, *'object*-rashi', *'object*-kamo' and *'object*-denai'si given about each kinds of objects.

We represent a set of productions which give the interpretation of *'object'* by Q_1, a set of productions which give the interpretation of *'object*-rashi' by Q_2, a set of productions which give the interpretation of *'object*-kamo' by Q_3, and a set of productions which give

Table 4.1 Features of each kinds of objects.

feature	window	man	wall	car
intensity	33.99	33.54	35.21	69.55
area	106.7	292.4	691.8	319.0
compactness-2	68.5	39.8	40.5	42.5

Table 4.2 Classification using compactness-2.

class	window	man	wall	car
window car wall man	96		28	23
man wall car window		50	56	47
wall man car window	2			
car wall man window	2	34	14	23

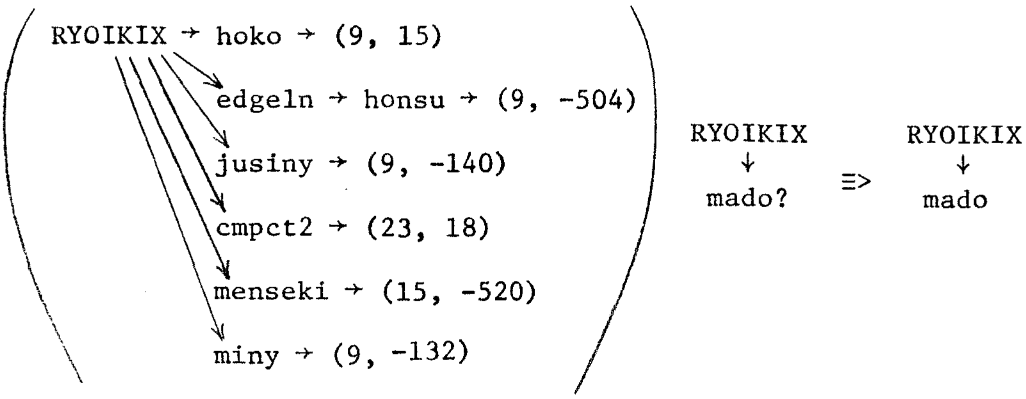


Fig. 4.8 An example of production which interprets the region as a window.

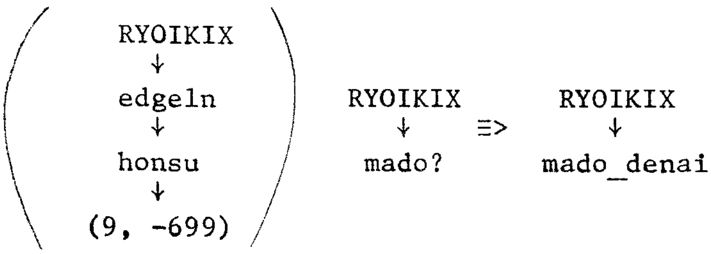
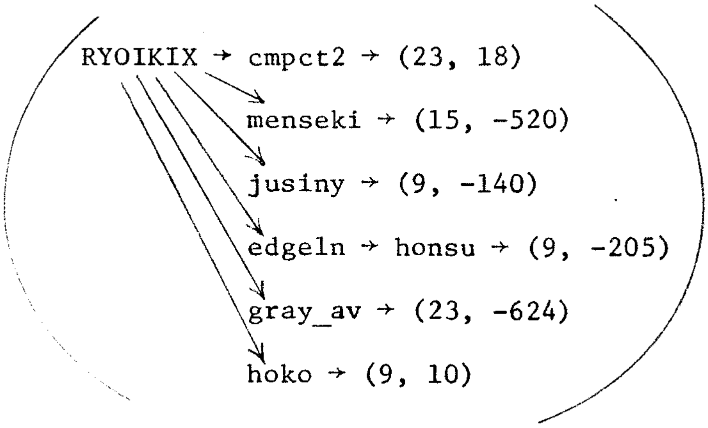


Fig. 4.9 An example of productions which interpret the region as not a window.



RYOIKIX ↓ mado? ≡> RYOIKIX ↓ mado_rashi

Fig. 4.10 An example of productions which interpret the region as a window probably.

(RYOIKIX) RYOIKIX ↓ mado? ≡> RYOIKIX ↓ mado_kamo

Fig. 4.11 An example of productions which interpret the region as a window.

the interpretation of '*object*-denai' by Q_4. Productions must be applied in accordance with the sequence Q_1, Q_2, Q_4, Q_3. In the definitions of IGPS there is no priority among productions. But we need to order the IGPS interpreter to preserve the sequence of applications of productions. Because a production in Q_2 is constructed by loosening an applicable condition of production in Q_1. So if a production in Q_2 was applied before an application of productions in Q_1, the region was interpreted as '*object*-rashi' while the region could be interpreted as '*object*'. And productions which give the interpretation of '*object*-kamo' must be applied when any productions in Q_1, Q_2 or Q_3 can not be applied. Then an example of a graph which represents interpretations is shown in Fig. 4.12.

4.4.2 Region-wise synthetic inference

Region-wise synthetic inference is done using a result of previous inference in accordance with 3 assumptions listed below.

[Assumption 4.2]

1. A region which is given only interpretations '*object*-denai', constructs an object which is not included our world models, so it can not be interpreted in our system, so it does not need more processing.

2. A region which is given an interpretation '*object*', can be decided to be the object.

3. A region which is given an interpretation '*object*-rashi' or '*object*-kamo', and an interpretation '*object*-denai' for other kinds of objects, is more probably the object.

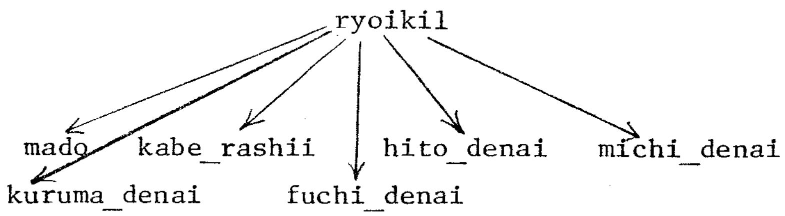


Fig. 4.12 An example of interpretation of a region.

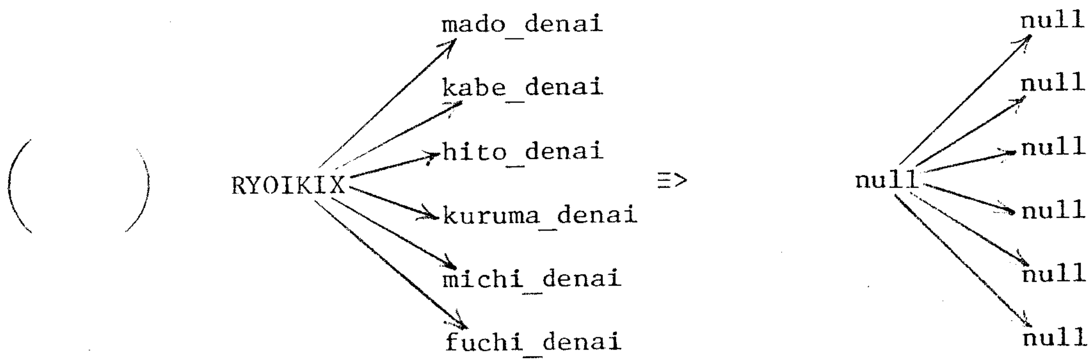


Fig. 4.13 An example of productions which are used in phase 2.

From those assumptions we obtain three productions which are shown in Figs. 4.13, 4.14 and 4.15. The production of Fig. 4.13 is derived from Assumption 4.2.1, and it deletes a sub-graph which describes interpretations of the region, the sub-graph holds the information about the region which can not be treated in our world model. In Figs. 4.13, 4.14 and 4.15, the label 'RYOIKIX' is a variable whose range is a set of region numbers, and the label 'FOBJECT' is a variable whose range is the set:

{ mado, kabe, hito, kuruma, michi, fuchi },

and 'FROBJECTA', 'FROBJECTB', 'FROBJECTC', 'FROBJECTD' and 'FROBJECTE' are variables whose ranges are the sets:

{ mado_rashi, mado_kamo, mado_denai, kabe_rashi, kabe_kamo, kabe_denai, hito_rashi, hito_kamo, hito_denai, kuruma_rashi, kuruma_kamo, kuruma_denai, michi_rashi, michi_kamo, michi_denai, fuchi_rashi, fuchi_kamo, fuchi_denai }.

And 'null' is the label which expresses that a node must be deleted by an IGPS interpreter. The production of Fig. 4.14 is derived from Assumption 4.2.3, and it rewrites the label 'mado_rashi'to the label 'mado', when other interpretations are only '*object*-denai'. The production of Fig. 4.15 is derived from Assumption 4.2.2, and it deletes labels '*object*-rashi', '*object*-kamo' and '*object*-denai', of the region when the region is interpreted as '*object*'. Because analysis about other kinds of objects is not needed. 12 productions which include the three productions shown in Figs. 4.13, 4.14 and 4.15 are used for de-

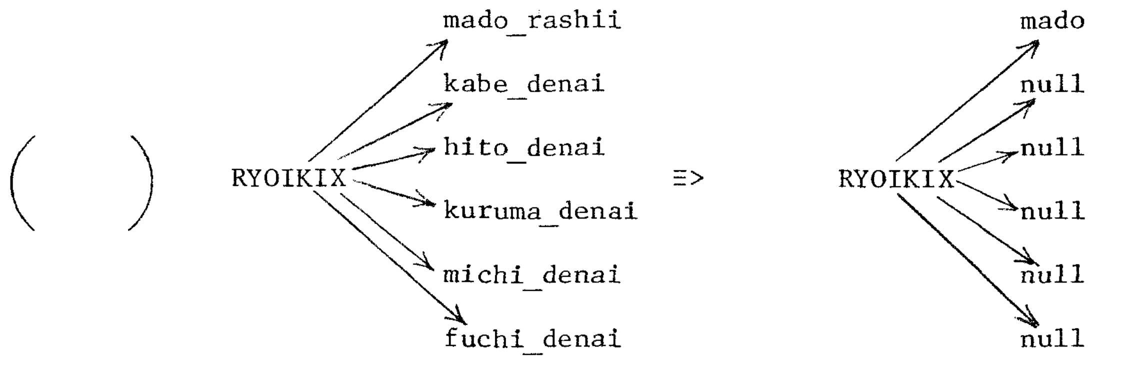


Fig. 4.14 An example of productions which are used in phase 2.

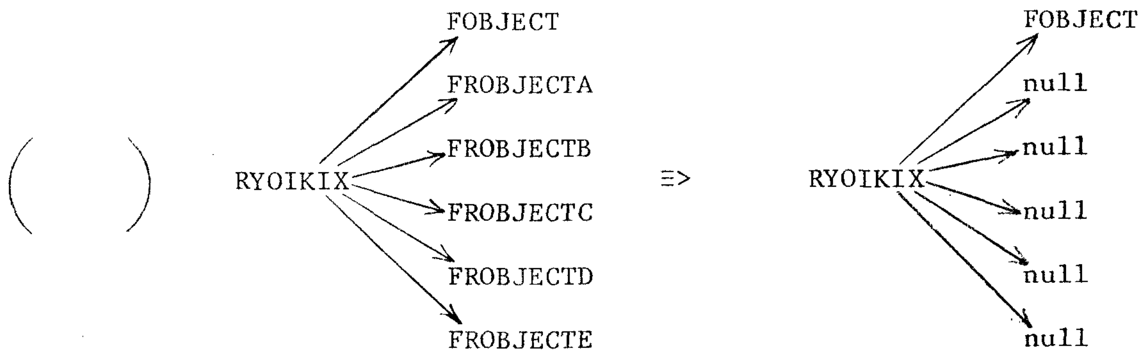


Fig. 4.15 An example of productions which are used in phase 2.

leting unnecessary informations and inferring synthetically.

4.4.3 Inference based on the relations among regions

Next inferences are done more synthetically using relations among regions. There are two relations explicitly in the graph which represents an image, and those are *'is over'* and *'is right'*. Other relations can be derived from the center of gravity of regions and etc. In this chapter, only connectivities between regions are used. Productions for inference using inter-region relations, are derived from the assumptions listed below.

[Assumption 4.3]

1. Edges exist between two object.
2. Same kind of objects do not contact.
3. Probability of them are large that a window contacts a wall, a road contacts a man or a car, and a wall contacts a car or a man.
4. A window does not contact a road.

From those assumptions the production of Figs. 4.16, 4.17, 4.18 and 4.19 are derived. In those productions the label 'KANKEI' is a variable whose range is the set:

$$\{ \text{over}, \text{right} \},$$

and the label 'vryoiki' is a constant which represents that regions are parts of an object. The label 'CYU' is a variable whose range is the set of numbers larger than 5. And the label 'OBJECTRX' is a variable whose range is the set:

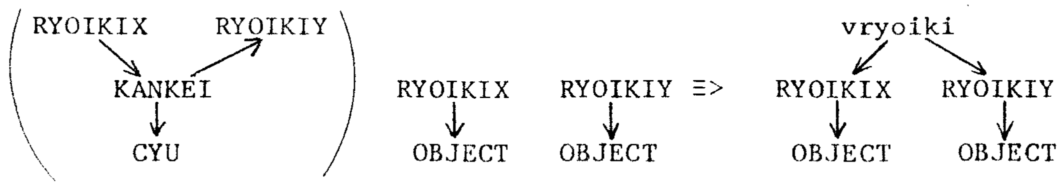


Fig. 4.16 A production used in phase 3.

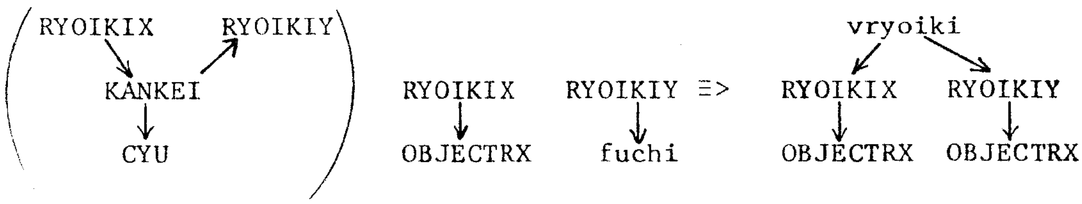


Fig. 4.17 A production used in phase 3.

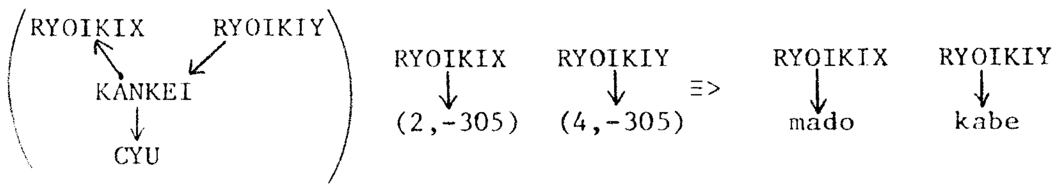


Fig. 4.18 A production used in phase 3.

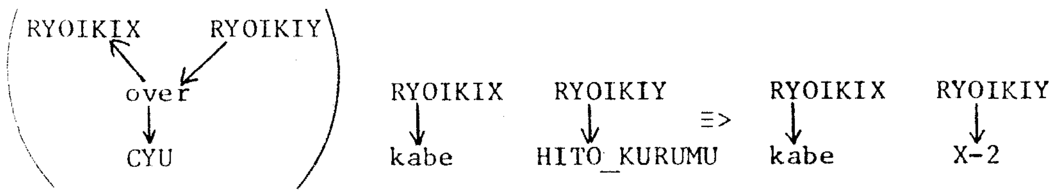


Fig. 4.19 A production used in phase 3.

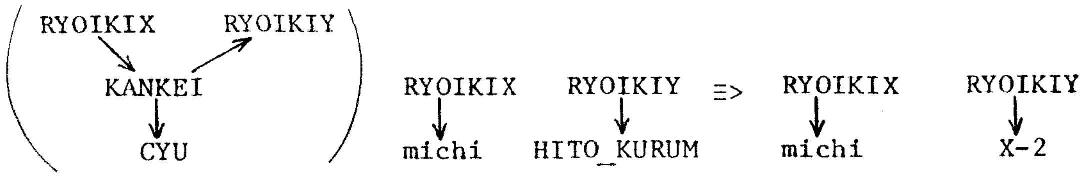


Fig. 4.20 A production used in phase 3.

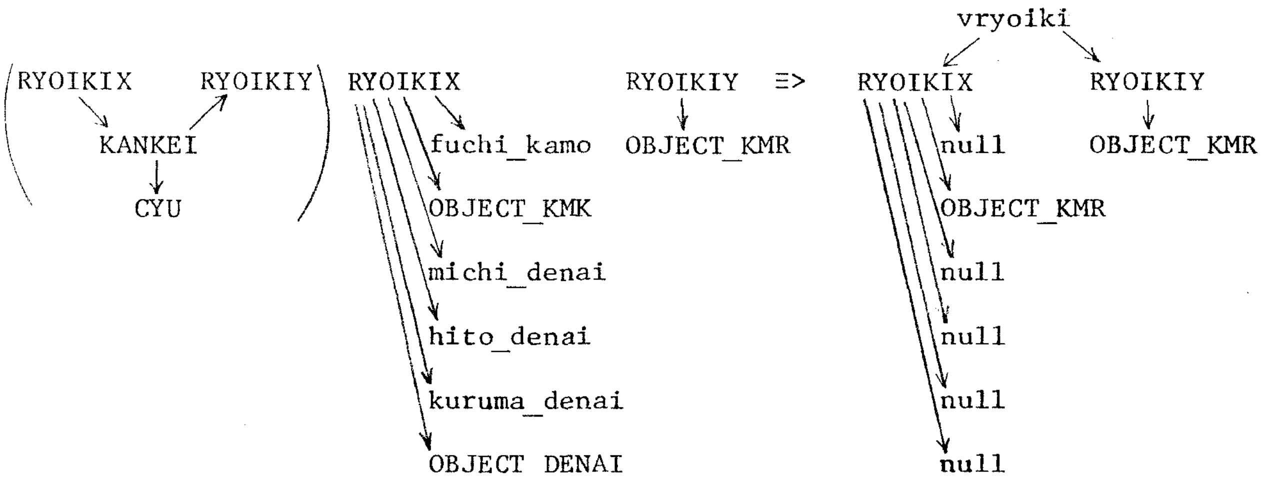


Fig. 4.21 A production used in phase 3.

{ mado, mado_rashi, kabe, kabe_rashi, hito, hito_rashi,

kuruma, kuruma_rasi, michi, michi_rashi }.

And the label 'HITO_KURUM' is a variable whose range is the set:

{ hito_rashi, kuruma_rashi }.

The production of Fig. 4.16 is derived from Assumption 4.3.2, and it attach the label 'vryoiki' to the regions which are parts of an object. The production of Fig. 4.17 is derived from Assumption 4.3.1, and it combines a region which is interpreted as an edge and a region which is given proper interpretation. The production of Fig. 4.18 derived from Assumption 4.3.3, and it decides that a region whose interpretation is 'mado_rashi' and a region whose interpretation is 'kabe_rashi' are a window and a wall respectively when these regions contact each other. The production of Fig. 4.19 is derived from Assumption 4.3.3 and it expresses that a region interpreted as 'hito_rashi' or 'kuruma_ rashi' contacts a region interpreted as 'kabe', then its interpretation is changed 'hito' or 'kuruma' respectively. And the production of Fig. 4.20 is deribed from Assumption 4.3.3 and it expresses that if a region interpreted as 'hito_rashi' or 'kuruma_rashi' contacts a region interpreted as 'michi', then its interpretation is changed 'hito' or 'kuruma' respectively.

In Figs. 4.19 and 4.20, the label 'x-2' is not defined in the definitions of IGPS. That instructs the IGPS interpreter to rewrite a label of a node which relates the node of the palce of the label '*object*' if a value of 'HITO_KURUM' is '*object*-rashi'. If a value of the label 'HITO_KURUM' is 'hito_rashi', then a value of the label 'x-2' is 'hito', and if that is 'kuruma_rashi', then this is 'kuruma'. This function of the IGPS interpreter enables to describe some productions of IGPS by one production form. The production of Fig. 4.21 is derived from Assumptions 4.3.2 and 4.3.3. In the production, 'OBJECT_KMK' is a variable whose range is the set:

{ mado_kamo, kabe_kamo },

and 'OBJECT_KMR' is a variable whose range is the set:

{ mado_rashi, kabe_rashi }.

And the label 'OBJECT_DENAI' is a variable whose range is the set:

{ mado_denai, kabe_denai, hito_denai,
kuruma_denai, michi_denai, fuchi_denai }.

The production of Fig. 4.21 connects two regions whose region numbers are 'RYOIKIX' and 'RYOIKIY', if the region whose region number is 'RYOIKIX', is given an interpretation of 'fuchi_kamo', and one of 'mado_kamo' and 'kabe_kamo' and '*object*-denai' to other objects, and the region whose region number is 'RYOIKIY', is given interpretations of 'mado_rashi', 'mado', 'kabe_rashi' or 'kabe', and those two regions are contacted. The productions listed above connect regions which are parts of an object and make obvious interpretations whose ambiguities are large.

4.5 Implementation of Image Interpreting System

The Image Interpreting System discussed in this chapter is constructed with three sub-systems. In Fig. 4.22 we show structure of the Image Interpreting System. The core of the Image Interpreting System is the IGPS interpreter, which receives a set of productions and a set of initial sub-situations from others: the Image Describing System and the Production Editing System. The Image Describing System receives a segmented image and generates graphical description of the segmented image. The Production Editing System enables to edit production on a graphic display unit and generates a set of productions.

The Image Describing System recieves a segmented image and its output is a labelled directed graph which describes the segmented image. An output of the Image Describing System has the structure discussed in 4.2, and it is inputted into the IGPS interpreter.

4.6 Experimantal examples

In this section we will show experimental examples. Fig. 4.23 shows a digitalized input image on a line printer by 8 levels. We show an interpretation of the image of Fig. 4.23, in Fig. 4.24.

In a figure expressing an interpretation, a window is printed by 'マ', a wall by 'カ', a man by 'ヒ', a car by 'ク', a road by 'ミ', and an edge by 'フ'. And a region which is printed nothing is given some interpretations or no interpretations and can not be decided to be a kind of objects.

The image of Fig. 4.23 is segmented into 43 regions. Shown in

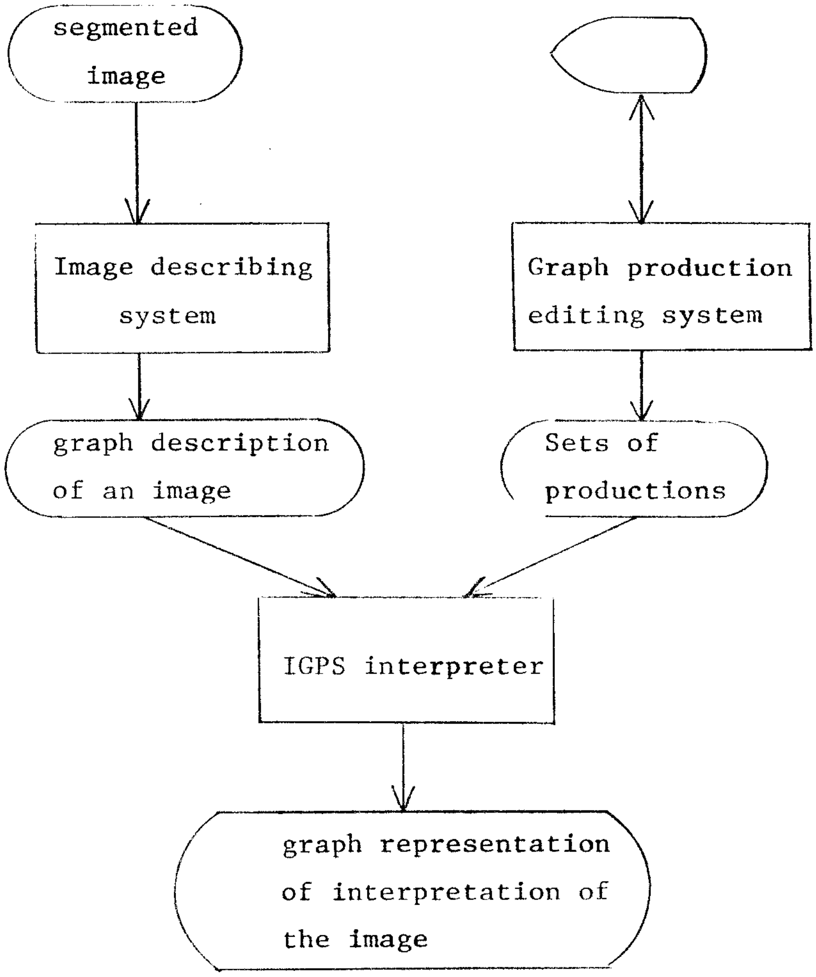


Fig. 4.22 Structure of the Image Interpreting System.

Fig. 4.24 the Image Interpreting System recognizes the collection of regions which construct an object effectively.

We show other examples in Fig. 4.25.

4.7 Conclusion

In this chapter, image interpretation using Interactive Graph Production System is discussed. As shown in section 4.6, we obtain good results in spite of a little set of productions.

From the examples we conclude that the Image Interpreting System can interpret segmented images effectively. The Image Interpreting System uses only two inter-region relations. Using other relations, more powerful interpretation can be done. And for more powerful interpretation other kinds of objects must be introduced in the world model.

Using Interactive Graph Production System for describing the process of image interpretation, we can simplify a description of a complex process which can not be described simply using regular programming languages.

From the view point of Interactive Graph Production System, it is shown that IGPS can describe a complex system. The IGPS interpreter used here is almighty type and does not implement enough processing speed.

IGPS enables us to represent a chunk of knowledge about images and their interpretations. In MSYS, we must treat total knowledge about images at once for representing how to interpret images, so about complex images we can not represent knowledges easily. Using IGPS, we can represent chunks of knowledge one by one, so we can treat complex images easily.

In future, we will develop a parallel IGPS interpreter which enables us to interpret IGPS more efficiently. And we will develop a simple IGPS interpreter which enables to interpret restricted IGPS more efficiently. For treating more complex images, more complex an image is, more and more processing time the IGPS interpreter consumes. And using IGPS's abilities fully, we will construct dynamic image interpreting systems.

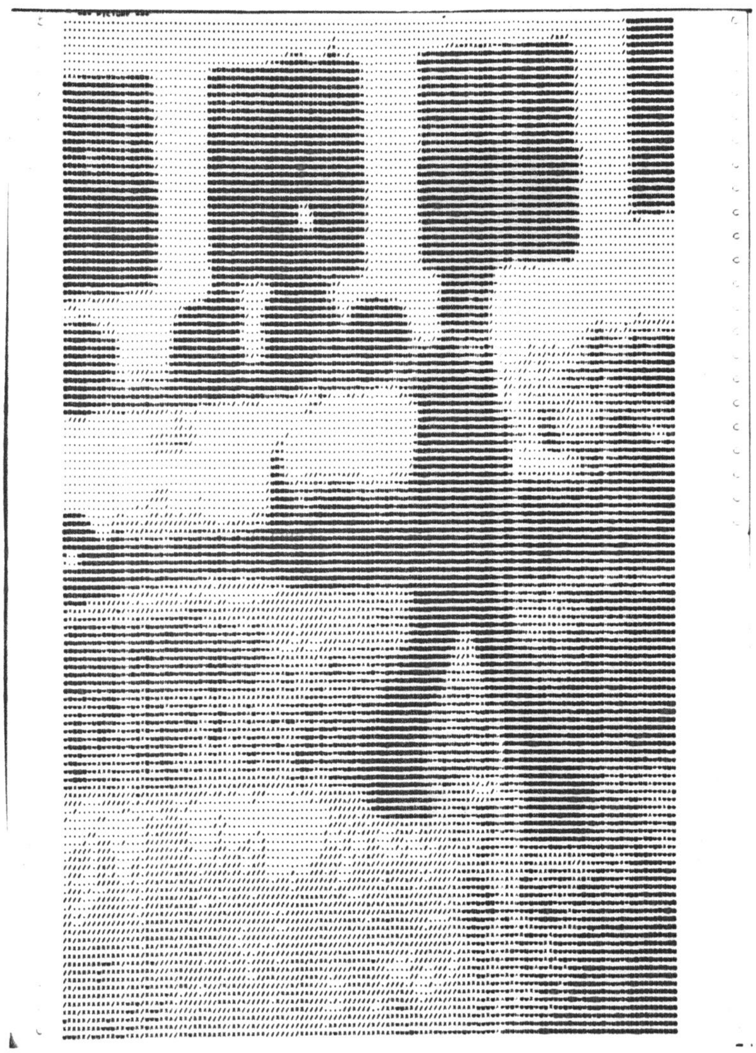


Fig. 4.23 An example of digitalized input images

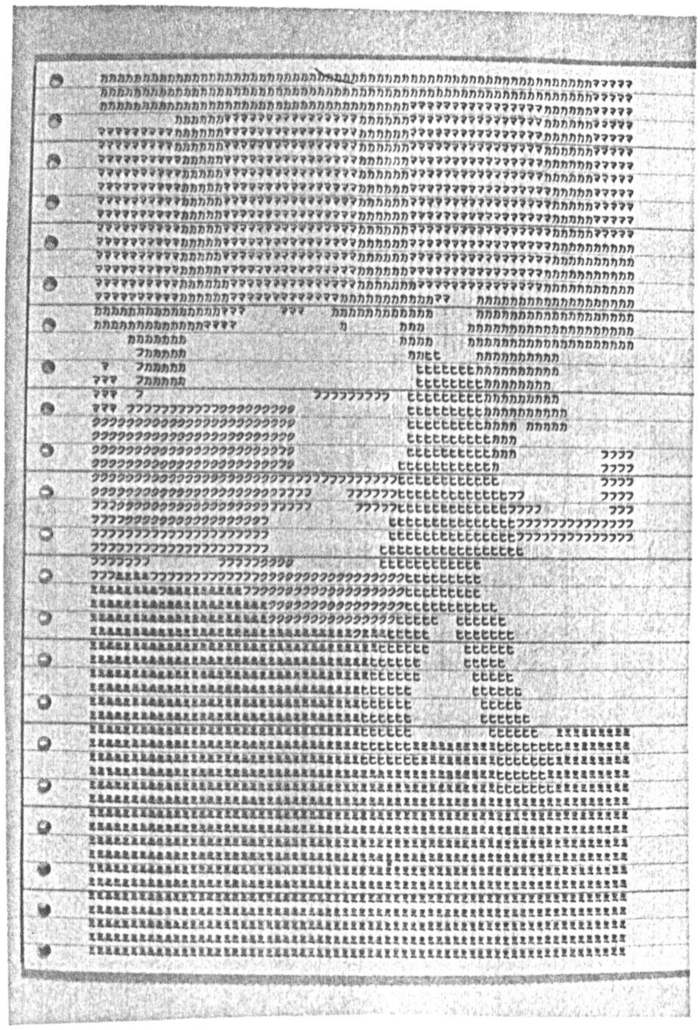
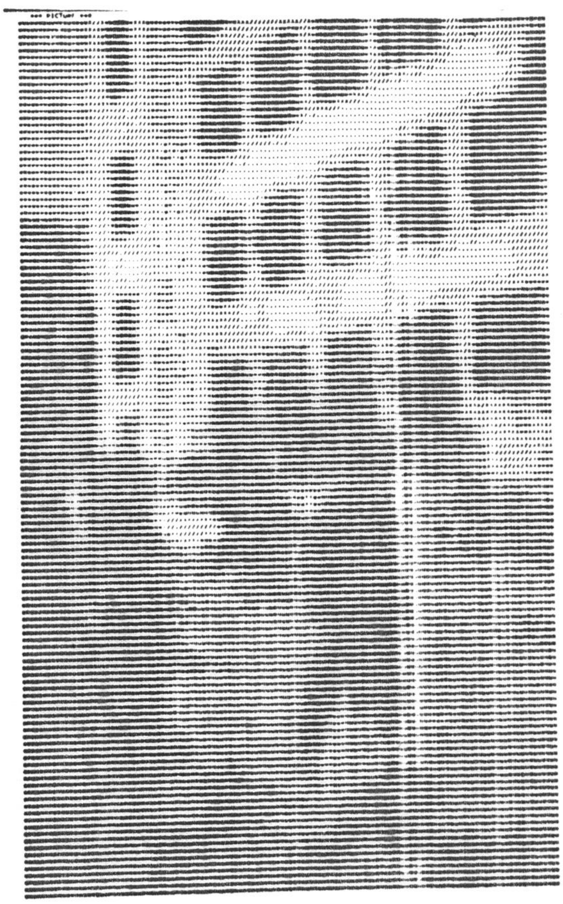
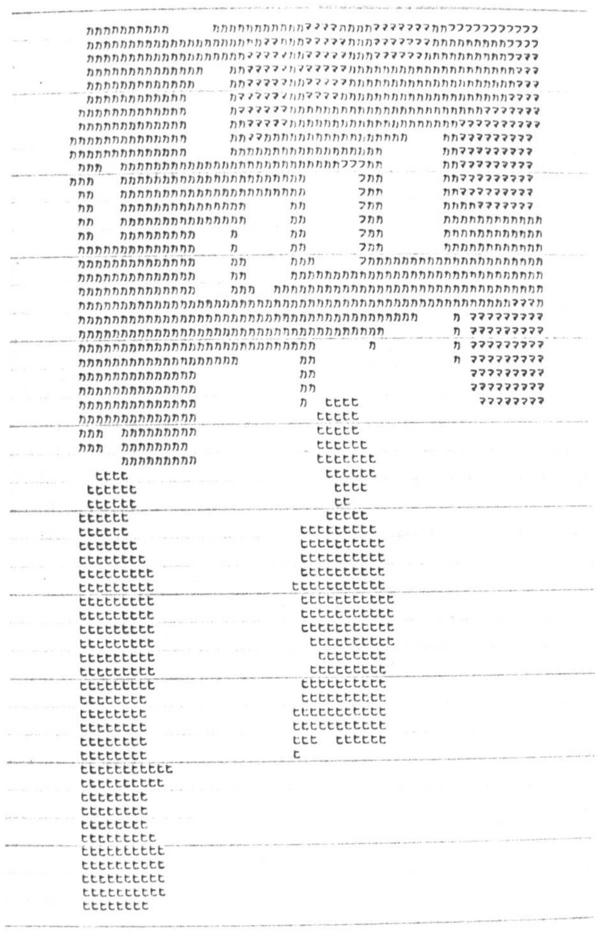


Fig. 4.24 Interpretation of the image of Fig. 4.23.

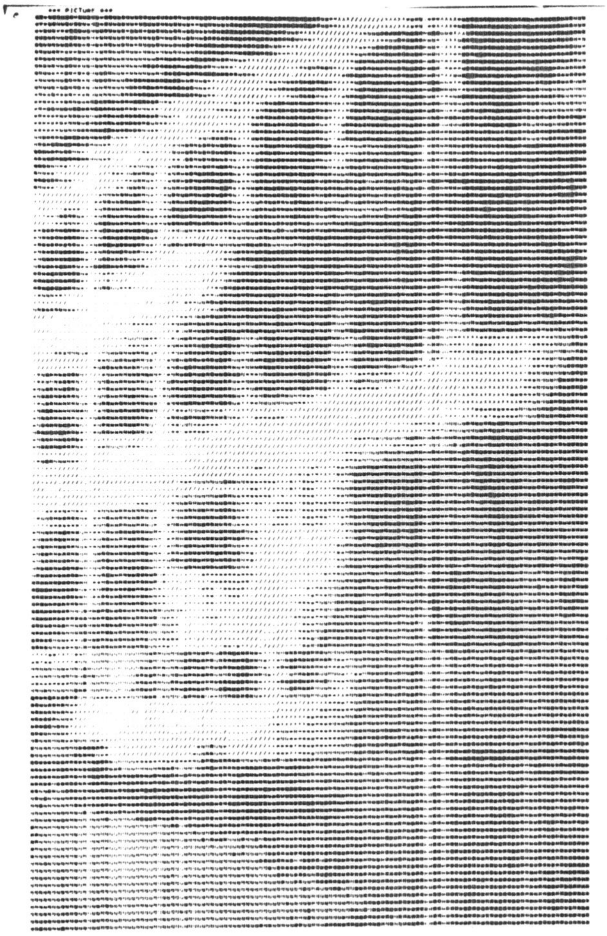


(a) A digitalized input image.



(b) Interpretation of the image of (a).

Fig. 2.25 Examples of image interpretations. (partial)



(c) A digitalized input image.

(d) Interpretation of the image of (c).

Fig. 2.25 Continued.

CHAPTER 5

Conclusions

In this thesis a new formal system has been proposed, that models systems interacting each other, and a new production system has been proposed that is constructed by extending descriptive power of the interactive system. And using the production system image interpretation has been discussed.

In chapter 2, interactive systems have been proposed and their abilities have been studied. An interactive system is constructed with two web grammar systems and two interaction functions which express interaction. It is shown that the well-known quotation from Homer's Iliad: *"Two head are better than one."* is true for formal systems. And if the powers of two systems interacting each other are high, then the abilities of interactive systems do not depend on the power of interaction functions. Interpreting this result in the real world, high-able ones can keep in enough communication by exchanges of simple information.

In chapter 3, an interactive graph production system has been proposed. IGPS (Interactive Graph Production System) is developed by extending descriptive power of the interactive system by introducing variables. And IGPS is simplified by constricting interactive systems' formation. Three examples have been shown. Three examples are three coin problem, monkey and banana problem and block world manipulations problem. In the three coin problem, changing of situations in a data base is described. In the monkey and banana problem, try and error process is represented. And in the block world manipulations problem, complex situations are represented.

In chapter 4, using the IGPS an image interpreting system has been constructed. The image interpreting system has been represented simply using the IGPS, and can be easily modified. The image interpreting system contains about 120 productions, but it is experimental one. The image interpreting system treats over 500 nodes in examples. Now its processing speed is not enough for processing more complex image descriptions.

For the future, about interactive systems we will investigate relations between $ICFSL(A_i)$ and $ICFS(A_i)$, and between $ICFSL(B_i)$ and $ICFL(B_i)$, and also between $ICFSL(B_i)$ and $ICFSL(B_{i+1})$, and etc. Investigating those relations we may construct bases for comparing complexities of many porblems.

About Interactive Graph Production System, we will apply IGPS to a diverse collection of problems and make obvious the merit and the demerit of IGPS. Representing probelms by IGPS, we may have a new measure of complexities of problems using relations among abilities of interactive systems. And we will implement a more powerful IGPS interpreter.

About image interpretation, introducing more kinds of objects into the world model we will increase the ability of the Image Interpreting System.

REFERENCES

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

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).

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).

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).

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

6. Aoki, K., Tanaka, K.,"Interactive systems and their abilities", Trans. Inst. Elect. Commun. Engineers of Japan, 61-D, 1, 1978, pp. 41-48 (in Japanese).

7. Aoki, K., Tanaka, K.,"The abilities of the interactive systems of context-free form", Trans. Inst. Elect. Commun. Engineers of Japan, 61-D, 8, 1978, pp. 525-532 (in Japanese).

8. Aoki, K., Tanaka, K.,"Interactive graph production system", Trans. Inst. Elect. Commun. Engineers of Japan, 61-D, 12, 1978, pp. 893-900 (in Japanese).

9. Aoki., K., Miyoshi, H., Tanaka, K.,"Interactive image analizing system", 1977 National Convention Records on Information Processing of the Inst. Elect. Commun. Engineers of Japan, 1977, p. 150 (in Japanese).

10. Barrow, H. G., Popplestone, J. R.,"Relational descriptions in picture processing", Machine Intelligence 6, Edinburgh Univ. Press, Edinburgh, 1971, pp. 377-396.

11. Barrow, H. G., Tenenbaum, J. M.,"MSYS: A SYSTEM FOR REASONING ABOUT SCENES", A. I. Center Tech. Note 121, SRI, 1976.

12. Chen, C. H. (Ed),"Patter recognition and artificial intelligence", Academic Press, New York, 1976.

13. Davis, R., Buchanan, B. G., Shortliffe, E. H.,"Production rules as a representation for a knowledge-based consultation program", A. I. Memo 266, Comp. Science Dept., Stanford Univ., 1975.

14. Ezawa, Y.,"Web languages and interactive languages", Ph. D. Thesis, Osaka univ., Feb., 1975.

15. 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).

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

17. 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, 1973, No. 1186 (in Japanese).

18. Feigenbaum, E. A., et. al.,"On generality and problem solving - a case study involving the DENDRAL program", Machine Intelligence 6, Edinburgh Univ. Press, Edinburgh, 1971, pp. 165-190.

19. Fikes, R. E., Nilsson, N. J.,"STRIPS: A new approach to the application of theorem proving to problem solving", Artificial Intelligence, 2, 3/4, 1971, p. 189.

20. Fu, K. S.,"Syntactic methods in pattern recognition", Academic Press, New York, 1974.

21. Jackson, P. C.,"Introduction to artificial intelligence", Petrocelli, New York, 1974.

22. Lipkin, B. S., Rosenfeld, A.,"Picture processing and psychopictorics", Academic Press, New York, 1970.

23. Milgram, D. L.,"Web automata", Univ. of Maryland Comp. Science Center Technical Reports 181, 1972.

24. Miyoshi, H., Aoki, K., Tanaka, K.,"Simple texture analysis", 1978 National Convention Records of the Inst. Elect. Commun. Engineers of Japan, 1978, p. 5-39 (in Japanese).

25. Moayer, B., Fu, K. S.,"A syntactic approach to fingerprint pattern recognition", Technical Report TR-EE 73-18, School of Elec. Eng., Purdue Univ., Indiana, 1973.

26. Montanari, U. G.,"Separable graphs, planar graphs, and web grammars", Information and Control, 16, 1970, pp. 243-267.

27. Moran, T. P.,"The symbolic hypothesis: aproduction system model", Comp. Science Dept., Carnegie-Mellon Univ., 1973.

28. Mylopoulos, J.,"On the relation of graph automata and graph grammars", Technical Report 34, Dept. of Comp. Science, Univ. of

Toronto, 1971.

29. Newell, A.,"Production systems: models of control structures", Visual Information Processing, Academic Press, 1973, pp. 463-526.

30. Nilsson, N. J.,"Problem solving methods in artificial intelligence", McGraw-Hill, New York, 1971.

31. Pavidis, T.,"Linear and context-free graph grammars", Journal of ACM, 19, 1972, pp. 11-22.

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

33. Pfaltz, J. L., Rosenfeld, A.,"Web grammars", Proc. 1st IJCAI, 1969, pp. 609-619.

34. Post, E.,"Formal reductions of the general combinatorial problem", Am. Jnl. Math., 65, 1943, pp. 197-263.

35. Rosenfeld, A.,"Picture processing by computer", Academic Press, New York, 1969.

36. Rosenfeld, A., Strong, J. P.,"A grammar for maps", in Software Engineering, 2, J. T. Tou. Ed., Academic Press, New York, 1971, pp. 226-239.

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

38. Shank, H. S.,"Graph property recognition machine", Mathematical Systems Theory, 5, 1971, pp. 45-49.

39. Smith, D. H., et. al.,"Applications of artificial intelligence for chemical inference VIII", Journal of the American Chemical Society, 94, 1972, p. 5962.

40. Tate, A.,"Interacting goals and their use", 4th IJCAI, 1975,

pp. 296-303.

41. Torii, K., Sugito, Y., Mano, Y.,"GMS/I:An interactive graph manipulation system", 1st USA-Japan Comp. Conf.,Proc., 1972, p. 593.

42. Undewood, W. E., Kanal, L. N.,"Structural descriptions, transformational rules, and pattern analysis", 1st IJCPR, 1973.

43. Vere, S. A.,"Relational production systems", Artificial Intelligence, 8, 1, 1977, pp. 47-68.

44. Winston, P. H. (Ed),"The psychology of computer vision", McGraw-Hill, New York, 1975.

45. Winston,P. H."Artificial intelligence", Addison-Weslay, Massachusetts, 1977.

46. Yakimovsky, Y., Feldman, J.,"A semantic-based decision theoretic region analyzer", 3rd IJCAI, 1973, pp. 580-588.

47. Yakimovsky, Y.,"Onthe recognition of complex structures: computer software using AI applied to pattern recognition", IJCPR, 1974, pp. 345-353.