DATA STRUCTURES WITH ADDRESSING FUNCTIONS

TATSUO TSUJI

FEBRUARY 1978

GRADUATE SCHOOL OF ENGINEERING SCIENCE

OSAKA UNIVERSITY

DATA STRUCTURES WITH ADDRESSING FUNCTIONS

by

TATSUO TSUJI

Submitted in partial fulfillment of
the requirement for the degree of

DOCTOR OF ENGINEERING
(Electrical Engineering)

at

OSAKA UNIVERSITY
TOYONAKA, OSAKA, JAPAN

February 1978

ACKNOWLEDGEMENT

The author wishes to express his sincerest gratitude to Professor K. Tanaka who supervised this thesis and provided constant encouragements and invaluable guidance during the entire course of this work.

Special thanks are due to Associated Professors J. Toyoda, S. Tamura, and T. Kitahashi for their substantial advice and continuous encouragements.

He wishes to thank Dr. M. Mizumoto, Dr. M. Tanaka, and the colleagues of Prof. Tanaka's laboratory, for their useful and enlightening discussions and criticisms.

He is also grateful to Dr. Y. Ezawa (presently is with Kansai University) and K. Fujii for their helpful discussions and kind cooperations.

PREFACE

One of the most widely employed techniques to implement a data structure on a computer memory is the one that allocates data items by an addressing function, as is typical in the realization of arrays. This method is decidedly superior to "chaining method" in some practical points such as accessing time or memory utilization. However, there have been no satisfactory theories that discuss generally the class of data structures which may admit efficient addressing functions.

"Data Graph Theory" developed by A.L.Rosenberg is very enlightening in that it proposed a data structure model simple enough to be treated mathematically and gives an keen insight into that class of data structures, by presenting many investigations into the structural uniformities of data graphs.

In this thesis, motivated by his instructive works, we discuss the problems of data structures with addressing functions.

In Chapter 2, a relational data graph $\Gamma = (C, R)$ is newly defined to describe and investigate a more general structure represented by a generalized directed graph in which more than two equi-labeled edges emanate from a node. We formulate and discuss the uniformities needed to acquire an "index set".

In chapter 3, the class of relational data graphs with strong uniformity is specified and characterized in detail, which is advantageous to devising an efficient addressing function.

In chapter 4, a new class of data structures called "TA-structures" is introduced, each of which has a composite structure of trees and arrays.

For TA-structures, their indexing methods are described, and a few types of addressing functions are constructed and evaluated.

In Chapter 5, the labeling scheme for TA-structures is presented and for a labeled TA-structure, its addressing scheme is explained.

February, 1978

Tatsuo Tsuji

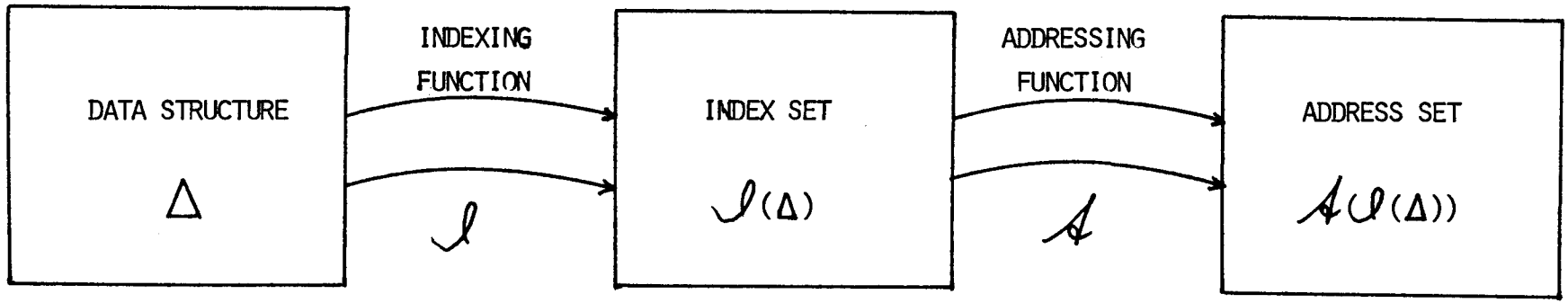
TABLE OF CONTENTS

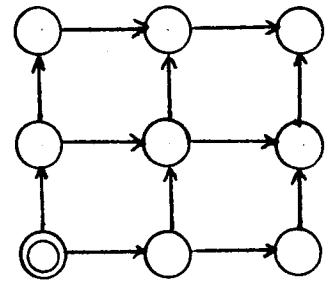
CHAPTER 1.

INTRODUCTION

One of the most widely employed techniques to implement a data structure on a computer memory is the one of allocating data items by an addressing function. This method is decidedly superior to "chaining method" from the following practical points of view. First, the mechanism for traversing the structure so realized tends to be simple and to require little overhead for "bookkeeping". Second, the "cost" of effecting transitions in the structure is often more uniform and, in many practical situations, this cost is uniformly low. Finally, with "full" graphs such as nonsparse arrays and complete trees, this technique tends to be more conservative of storage. However, this technique suffers at least two basic drawbacks. It tends to be wasteful of storage when applied to structures which are not "full" and it tends to be inflexible -minor changes in the structure may necessitate a totally different scheme for calculating addresses.

For a given data structure Δ (a logical structure represented by a directed graph), its addressing function $\mathcal{A}$ can be constructed by taking the next two steps (See Fig.1). First, by exploiting the structural uniformity of Δ, one should give an index to each cell in Δ which specifies the "relative position" from some "base cell"; of course, the indices of arrays are integer tupples, and those of trees are finite strings. Followingly, for the index set $\mathcal{I}(\Delta)$ obtained, one should design a function $\mathcal{A}$ which allocates $\mathcal{I}(\Delta)$ to a set of addresses, A, on a computer memory. $\mathcal{I}(\Delta)$ must be fully simple, however, to be able to construct





DIMENSION (M, N)
ARRAY

$$\begin{vmatrix} i + M(j-1) \\ j + N(i-1) \end{vmatrix}$$

$$\{\langle i, j\rangle \mid 1 \leqslant i \leqslant M,\ 1 \leqslant j \leqslant N\} \qquad \{1, 2, \ldots, MN\}$$

Figure 1. Two steps for constructing addressing functions

an efficient $\mathcal{A}$ in the simplicity of computation and memory utilization. The fully simple $\mathcal{J}(\Delta)$, however, is guaranteed only when strong uniformity exists in Δ. But, even if Δ is fairly uniformly structured and, consequently, fully simple $\mathcal{J}(\Delta)$ can be obtained, the construction of efficient $\mathcal{A}$ which maps $\mathcal{J}(\Delta)$ to address set A is often much difficult. The reason for this difficulty is that the structure of A is too simple to capture faithfully the "structural information" of $\mathcal{J}(\Delta)$, and thus Δ, according to some simple $\mathcal{A}$. Arrays are familiar data structures usually stored by addressing functions and only trees are occasionaly so stored except arrays.

The effort to discuss generally and theoretically such a process of designing an addressing function is a much laborious and difficult task, since the organization of the function is inherently dependent on the "shape" of individual data structures and little can be said generally. More concretely speaking, to clarify the answers theoretically to the following questions is substantially difficult:(1) What classes of data structures are implementable using a family of "simple" functions? (2) What class of functions suffices to implement some prespecified data structures?

Owing to these difficulties, there have been no satisfactory theories that deal with such data structures with addressing functions (or, we say directly accessible data structures). "Data Graph Theory" developed by A.L.Rosenberg, however, is very enlightening in that it proposes a data structure model simple enough to be treated mathematically, and gives a keen insight into that class of data structures, by presenting many investigations into the structural uniformities of data graphs.

In this thesis, motivated by his instructive and stimulating works, we discuss the problems of data structures with addressing functions.

In Chapter 2, a relational data graph $\Gamma = (C, R)$ is newly defined. It is possible that Γ captures more general and natural structure of data, which is represented by a generalized directed graph where more than two equi-labeled edges emanate from a node. Each element of R is a relation rather than a function on the set C of data cells. Owing to the relationality of $r \in R$, in relational data graphs, a set of data items can be obtained by the application of one retrieval procedure. We formulate and discuss the class of relational data graphs with faborable uniformities needed to acquire a simple index set. Furthermore, we make the detailed study of such uniformities by investigating the algebraic and graph-theoretical properties of R.

In Chapter 3, we specify the class of relational data graphs with strong uniformities, which shows the potential usefulness for devising a "good" function with respect to memory availability or computational simplicity, and so on. We give structural characterizations of relational data graphs with such strong uniformities.

In Chapter 4, with the theoretical basis developed in the previous chapters, we introduce a new class of data structures called "TA-structures". A TA-structure $\Gamma = (C, F)$ has a composite structure of trees and arrays, and it is allocated by an addressing function. Indexing methods for TA-structures are described, which reflect both the string type and integer tuple type indices. Moreover, a few types of addressing functions for the index set specified are constructed.

Chapter 5 is devoted to the labeling schemes for TA-structures,

an labeled TA-structures by some specified labeling scheme are presented. For an labeled TA-structure, its addressing mechanism is explained.

CHAPTER 2

RELATIONAL DATA GRAPHS

2.1 Introduction

Given a problem and its associated data, by analyzing the semantic of the problem, then imposing the problem-oriented structure on the data, we can construct an efficient procedure for solving the problem. Though even in a case of a simple problem, because of the many criteria conflicting with each other, it is generally difficult to obtain the optimum structuring of the data.

However once a data structure is established according to some criteria and implemented in a computer, many important properties of the data structure, on the computational stage, become independent of the contents of data items.

Instead of above mentioned problem-oriented approach to the data structuring, studying properties of data structures, whose analysis depends only on their forms themselves, would be an effective approach. Investigating algebraic and graph-theoretical properties of the various structures underlying data structures, we can expect to exploit the structures on which many fundamental manipulations can be applied effectively.

Much research has been done in concern with such morphological formulation of data structures. For example, Child[1], Rosenberg[17-20], Fleck[5], Turski[42,43]. Among such excellent works, "Data Graph Theory" developed by A.L.Rosenberg is very enlightening one where he proposed a model simple enough to be treated mathematically. A data graph

is obtained from a data structure by masking out the specific data items at the nodes of the structure and concentrating only on the linkages in the structure. Linkages denoting various "relations" among data items are partial functions λ's on C (the set of data cells). Data graph is defined in terms of these functions. Two notions arising in data graph realization have been isolated, namely relative addressing and relocatability, which can be studied in terms of the structures of the data graphs involved. In [17], these two notions are precisely formulated and those data graphs are characterized to which these two notions are applicable. In [18-20], the properties of those data graphs are investigated in detail.

In his formulation of data graph however, owing to the functionality of λ's, at most one item is related to some item by each of λ's. This makes it inevitable that only one data item is obtained by the application of one retrieval procedure. This would be a vital limitation when the size of data structures become large and their fast processings are demanded.

In this chapter, relational data graph $\Gamma = (C, R)$ is newly defined to describe and investigate more general structure as is represented in Fig.2.1 in which more than two equi-labeled edges emanate from a node. Each element of R is a relation rather than a function on the set C of data cells. We introduce "block partitionable relational data graph" which is mainly discussed in this chapter. In block partitionable relational data graph, owing to the relationality of $r \in R$, a set of data items called "block" can be obtained by one retrieval procedure. So a higher rank data such as a set of blocks can be also successfully

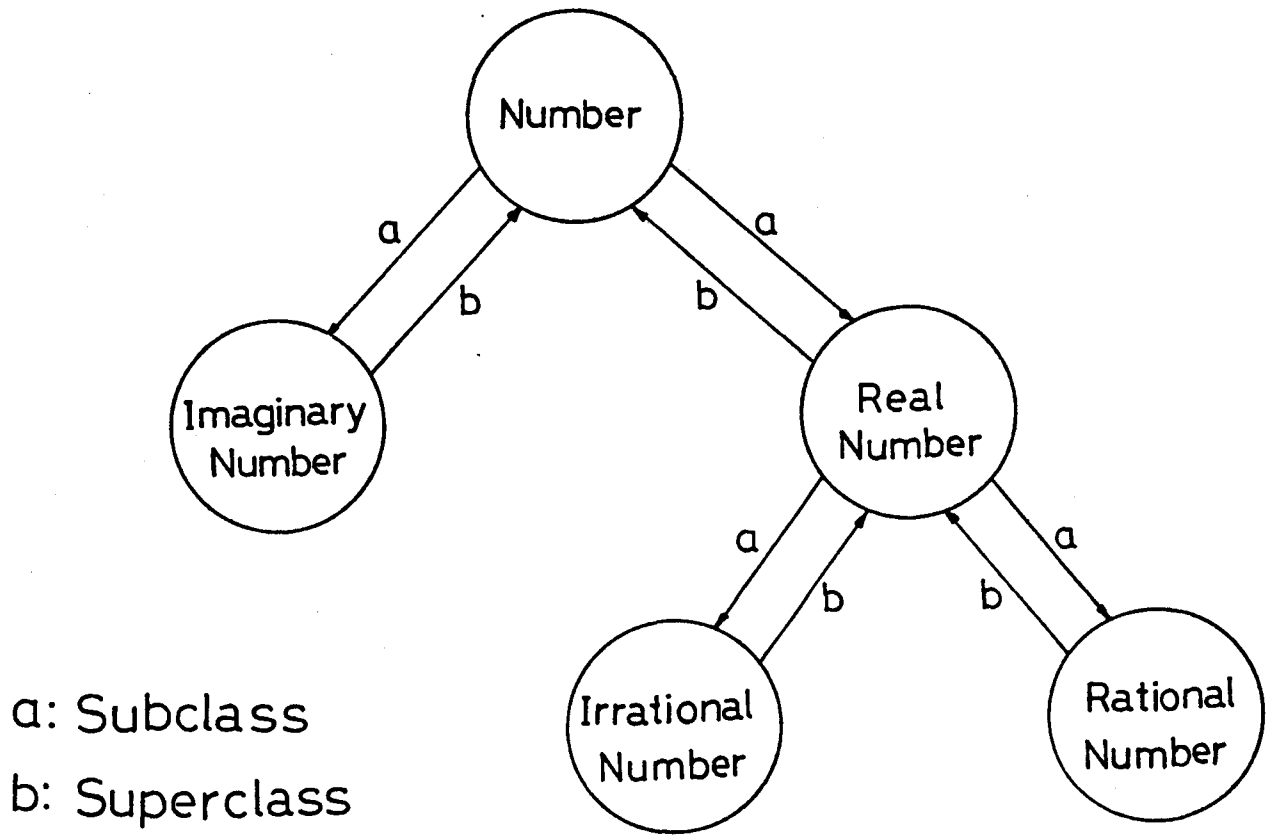


Figure 2.1. A representation of a data structure by a generalized directed graph

treated in our new model. By the notion of "root block", we characterize those class of block partitionable relational data graphs, each of whose blocks can be indexed by the link sequence from the root block cell. The detailed properties characteristic to the class of relational data graphs with root blocks are studied. In addition, we make various formulations such as "skeleton mapping" which would offer effective schemes for manipulating relational data graphs. Some results in [17-20] are naturally extended in our new model.

2.2 Relational Data Graphs and Block Partitionability

In this section, first a relational data graph is specified. Then, a block partitionable relational data graph is introduced and its several properties are studied. On the block partitionable relational data graphs, we formulate a class of relational data graphs which admit a "root block", and investigate various properties of them. For a relational data graph in this class, we can have an indexing method of it; each block in it can be indexed uniquely by the link sequence from the root block cell. It is stated in the next section that relational data graphs with root blocks can be allocated on a computer memory by "relative block addressing".

First we establish the notational conventions including the ones employed in the following chapters.

NOTATION. Let r be a binary relation on a set C, namely $r \in C \times C$. Each relation on C can be viewed as a function $r: C \longrightarrow 2^C$ (the power set of C), and the cell set $\{c' \in C \mid \langle c, c' \rangle \in r\}$ is denoted

by cr or often denoted by $r(c)$. For two relations r_1, r_2, we write r_1r_2 for the composite relation defined by, $cr_1r_2 = \{c' \in C \mid \langle d, c' \rangle \in r_2, d \in cr_1\}$ for all $c \in C$. We denote by R^τ the monoid of relations generated from R under relational compositions with identity function 1_C. We write r^k for k-fold composition of r with itself, whence $r^0 = 1_C$. $\nabla_R(c)$ is a set of relations defined on $c \in C$, that is $\{\xi \in R^\tau \mid \exists c' \in C, \langle c, c' \rangle \in \xi\}$. "a relation $\xi \in R^\tau$ is total on C" implies that $\xi \in \nabla_R(c)$ for every cell $c \in C$; that is, when viewed as a function, ξ is total on C. For $\xi \in R^\tau$ and $C' \subseteq C$, ξ/C' implies the restriction of ξ to C', that is $\xi/C' = \xi \subseteq C' \times C'$. For $\xi = r_1r_2 \ldots r_n \in R^\tau$ $(r_i \in R)$, the i-prefix ξ_i is the string $\xi_i = r_1r_2 \ldots r_i$ $(1 \leqslant i \leqslant m$, ξ_0 is the empty string).

Definition 2.1. A <u>relational data graph</u> (rdg for short) is specified as an ordered pair $\Gamma = (C, R)$, where

(i) C is a countable set of data cells.

(ii) R is a finite set of relations defined on C.

(iii) For all $c, d \in C$, there exists a relation $\xi \in R^\tau$ such that $d \in c\xi$, namely Γ is represented by a strongly connected directed graph.

Rosenberg's definition of a data graph is exactly equal to the definition of an rdg if we restrict relations to functions in (ii)

Definition 2.2. Let $\Gamma = (C, R)$ be an rdg. For any $\xi, \eta \in \nabla_R(b_0)$, if the following condition is obtained, $\{b_0\}$ is called a <u>base</u>

block of Γ and Γ is called a block partitionable rdg (bprdg for short).

$$b_0\xi \cap b_0\eta \neq \emptyset \quad \Longrightarrow \quad b_0\xi = b_0\eta.$$

It should be noted that baseblocks are singleton sets.

The set of base blocks of Γ is denoted by $\mathbb{B}_\Gamma$.

Example 2.1. Fig.2.2 is an example of a bprdg where $\mathbb{B}_\Gamma = \{\{1\}, \{9\}\}$.

Definition 2.3. Let $\Gamma = (C, R)$ be a bprdg and $\{b_0\} \in \mathbb{B}_\Gamma$. Then the relation $\simeq_{b_0}$ on C is defined as follows.

For all $c_1, c_2 \in C$,

$$c_1 \simeq_{b_0} c_2 \iff {}^{\exists}\xi \in \nabla_R(b_0), \quad c_1, c_2 \in b_0\xi.$$

Proposition 2.1. Let $\Gamma = (C, R)$ be a bprdg and $\{b_0\} \in \mathbb{B}_\Gamma$. Then the relation $\simeq_{b_0}$ is an equivalence relation on C.

Proof. That $\simeq_{b_0}$ is reflexive and symmetric is obvious from the definition. We show the transitivity of $\simeq_{b_0}$. For each $c_1, c_2, c_3 \in C$, let $c_1 \simeq_{b_0} c_2$ and $c_2 \simeq_{b_0} c_3$. Then, from the definition of $\simeq_{b_0}$, there exist $\xi, \eta \in \nabla_R(b_0)$ such that $c_1, c_2 \in b_0\xi$ and $c_2, c_3 \in b_0\eta$. Hence, $b_0\xi \cap b_0\eta \neq \emptyset$. Since, $\{b_0\}$ is a base block of Γ, $b_0\xi = b_0\eta$. Therefore $c_1, c_3 \in b_0\xi$, so that $c_1 \simeq_{b_0} c_3$. That $\simeq_{b_0}$ is an equiva-

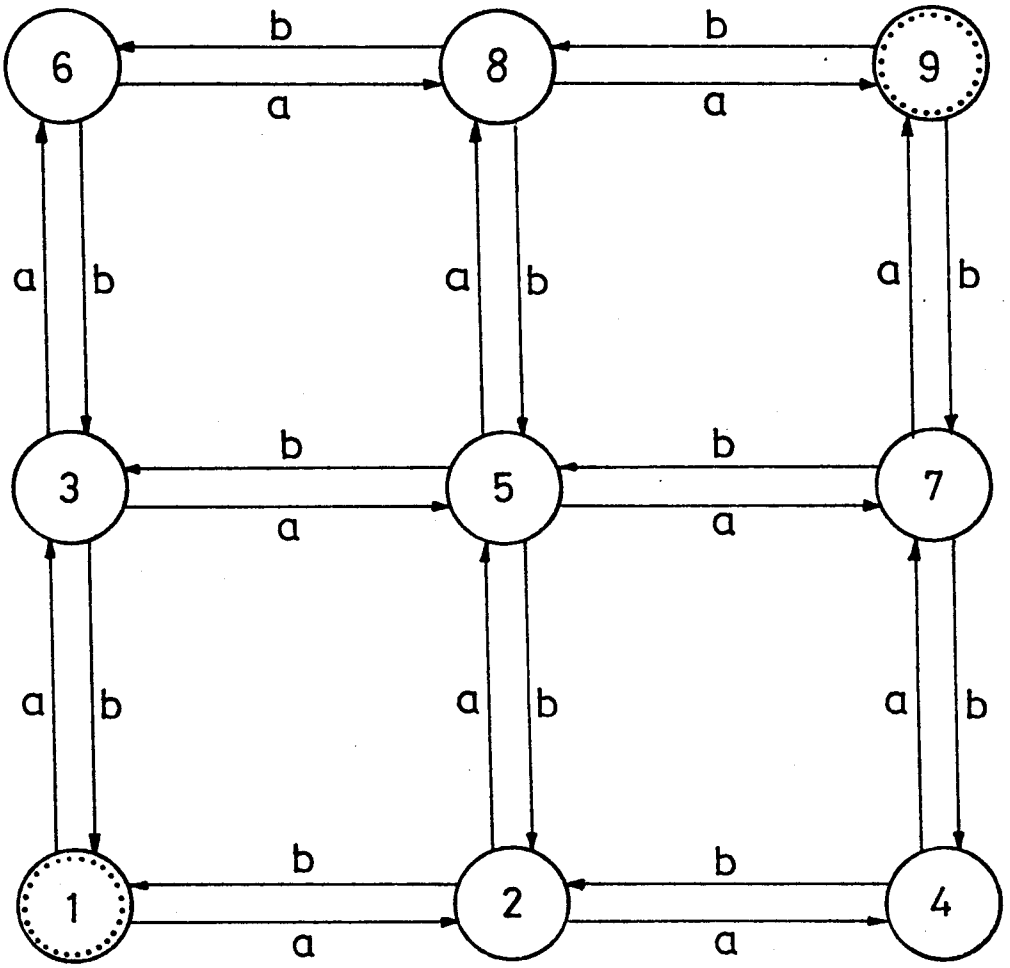


Figure 2.2. A block partitionable rdg

lence relation on C is now shown. Q.E.D.

From the above proposition, we can see that C is partitioned by the equivalence relation $\simeq_{b_0}$. Each equivalence class is called a block of Γ induced by a base block $\{b_0\}$.

Let $\mathcal{B}_\Gamma[b_0]$ denote the set of blocks of Γ induced by $\{b_0\}$, then $\mathcal{B}_\Gamma[b_0] = \{B \mid B = b_0\xi,\ \xi \in \nabla_R(b_0)\}$.

Example 2.2. For the bprdg Γ in Fig.2.2, $\mathcal{B}_\Gamma[1] = \mathcal{B}_\Gamma[9] = \{\{1\}, \{2, 3\}, \{4, 5, 6\}, \{7, 8\}, \{9\}\}$.

Base block $\{b_0\}$ is an entry block of Γ. Accessing to each block of Γ can be successfully accomplished by starting from the base block cell b_0. Some of the properties of bprdg's are developed, which result in Theorem 2.3.

Lemma 2.2. Let $\Gamma = (C, R)$ be a bprdg. For any cell $c \in C$, any base block $\{b_0\} \in \mathbb{B}_\Gamma$, and $\xi \in \nabla_R(c)$,

$$b_0 \in c\xi \implies c\xi = \{b_0\}.$$

Proof. From the strong connectivity of Γ, there exists $\eta \in \nabla_R(b_0)$ such that $c \in b_0\eta$. Say, there exists $d \in c\xi$ such that $d \neq b_0$. Then $b_0\eta\xi \supseteq \{d, b_0\}$, but since $1_C \in R^\tau$ is defined at every cell in C, $b_0 1_C = \{b_0\}$, so that $b_0\eta\xi \cap b_0 1_C \neq \emptyset$ and $b_0\eta\xi \neq b_0 1_C$. This contradicts that $\{b_0\}$ is a base block of Γ, that is, there are no $d \in c\xi$ such

that $d \neq b_0$. Hence, $c\xi = \{b_0\}$. Q.E.D.

Theorem 2.3. Let $\Gamma = (C, R)$ be a bprdg. For any two base blocks $\{b_1\}, \{b_2\} \in \mathbb{B}_\Gamma$,

$$\mathcal{B}_\Gamma[b_1] = \mathcal{B}_\Gamma[b_2].$$

Proof. Let $B \in \mathcal{B}_\Gamma[b_1]$ and $b_1\xi = B$ $(\xi \in \nabla_R(b_1))$. For $\zeta \in \nabla_R(b_1)$, we assume that $b_2 \in b_1\zeta$: Then from Lemma 2.2, $b_1\zeta = b_2$. For any $c \in B$, there is an $\eta \in \nabla_R(b_2)$ such that $c \in b_2\eta$ from the strong connectedness of Γ. Hence, $b_1\xi \cap b_1\zeta\eta \neq \emptyset$. Since $\{b_1\}$ is a base block of Γ, $b_1\xi = b_1\zeta\eta$ from Definition 2.2, namely $b_1\xi = b_2\eta$. Since $b_2\eta \in \mathcal{B}_\Gamma[b_2]$, $b_1\xi = B \in \mathcal{B}_\Gamma[b_2]$. Conversely, for any block $B \in \mathcal{B}_\Gamma[b_2]$, $B \in \mathcal{B}_\Gamma[b_1]$ is shown in the same way. Thus $\mathcal{B}_\Gamma[b_1] = \mathcal{B}_\Gamma[b_2]$ follows. Q.E.D.

The above theorem implies that the partitioning of C by the equivalence relation $\simeq_{b_0}$ gives the same set of blocks, which is independent of our choice of a base block $\{b_0\} \in \mathbb{B}_\Gamma$. This independence permits us from now on to denote the set of blocks of a bprdg simply as $\mathcal{B}_\Gamma$ without b_0.

Definition 2.4. Let $\Gamma = (C, R)$ be a bprdg. The <u>blocking mapping</u> $\gamma: C \longrightarrow \mathcal{B}_\Gamma$ is defined as follows.
For each $B \in \mathcal{B}_\Gamma$,

$$c\gamma = B \iff c \in B.$$

By the blocking mapping of Γ, each cell $c \in C$ is alloted to the block which contains it.

Definition 2.5. Let $\Gamma = (C, R)$ be a bprdg. If there exists $\{c_0\} \in \mathcal{B}_\Gamma$ such that for all $\xi, \eta \in \nabla_R(c_0)$,

$$c_0\xi = c_0\eta \implies \xi = \eta \qquad (2.1)$$

is satisfied, $\{c_0\}$ is called a <u>root block</u> of Γ.

Example 2.3. Fig.2.3 (A) is an example of an rdg which admits a root block $\{1\}$ and Fig.2.3 (B) is an rdg which admits two root blocks $\{1\}$ and $\{2\}$.

Sketch of proofs of Example 2.3.

Fig.2.3(A)

To show $\{1\}$ is a root block, first we can see the equalities $ab = 1_C$ and $a^2c = a$ hold. The set of blocks $\mathcal{B}_\Gamma$ is $\mathcal{B}_\Gamma = \{B \mid B = 1a^n, n = 0,1,2,\ldots\}$ $(a^0 = 1_C)$. Let ξ be an arbitrary relation to the block $1a^n$ $(n = 0,1,2,\ldots)$ and let $[\xi]_r$ be the number of occurence of $r \in R = \{a, b, c\}$ in ξ. Then, $[\xi]_a = [\xi]_b + [\xi]_c + n$ must holds. Repeated application of the above equalities into ξ reduces ξ to a^n due to $[\xi]_a = [\xi]_b + [\xi]_c + n$.
For example, consider the block $1a = \{3, 4\}$. Let $\xi = a^3bc$, so $1a = 1\xi$. Then,

$$a^3bc = a^2(ab)c$$

$= a^2c$ $(\because ab = 1_c)$

$= a$. $(\because a^2c = a)$

In this way, every relation to the block $1a^n$ $(n = 0,1,2,...)$ can be shown to be equal (as a set of binary relations) to a^n, so all are equal. Since, $1a^n$ is an arbitrary block, $\{1\}$ is a root block.

Q.E.D.

Fig.2.3 (B)

First, to show $\{1\}$ is a root block, note that the equalities $ab = c^2 = 1_c$, $ac = a$ hold. The set of blocks is $\mathcal{B}_\Gamma = \{B \mid B = 1a^n, n = 0,1,2,...\} \cup \{B \mid B = 1ca^n, n = 0,1,2,...\}$. Let ξ be an arbitrary relation to the block $1a^n$. For such ξ, $[\xi]_a = [\xi]_b + n$ must hold. Again, repeated application of the above equalities reduces ξ to a^n, due to $[\xi]_a = [\xi]_b + n$.

For example, consider the block $1a = \{3, 4\}$. Let $\xi = a^3bcb$, so $1a = 1\xi$. Then,

$a^3bcb = a^2(ab)cb$

$= a^2cb$ $(\because ab = 1_c)$

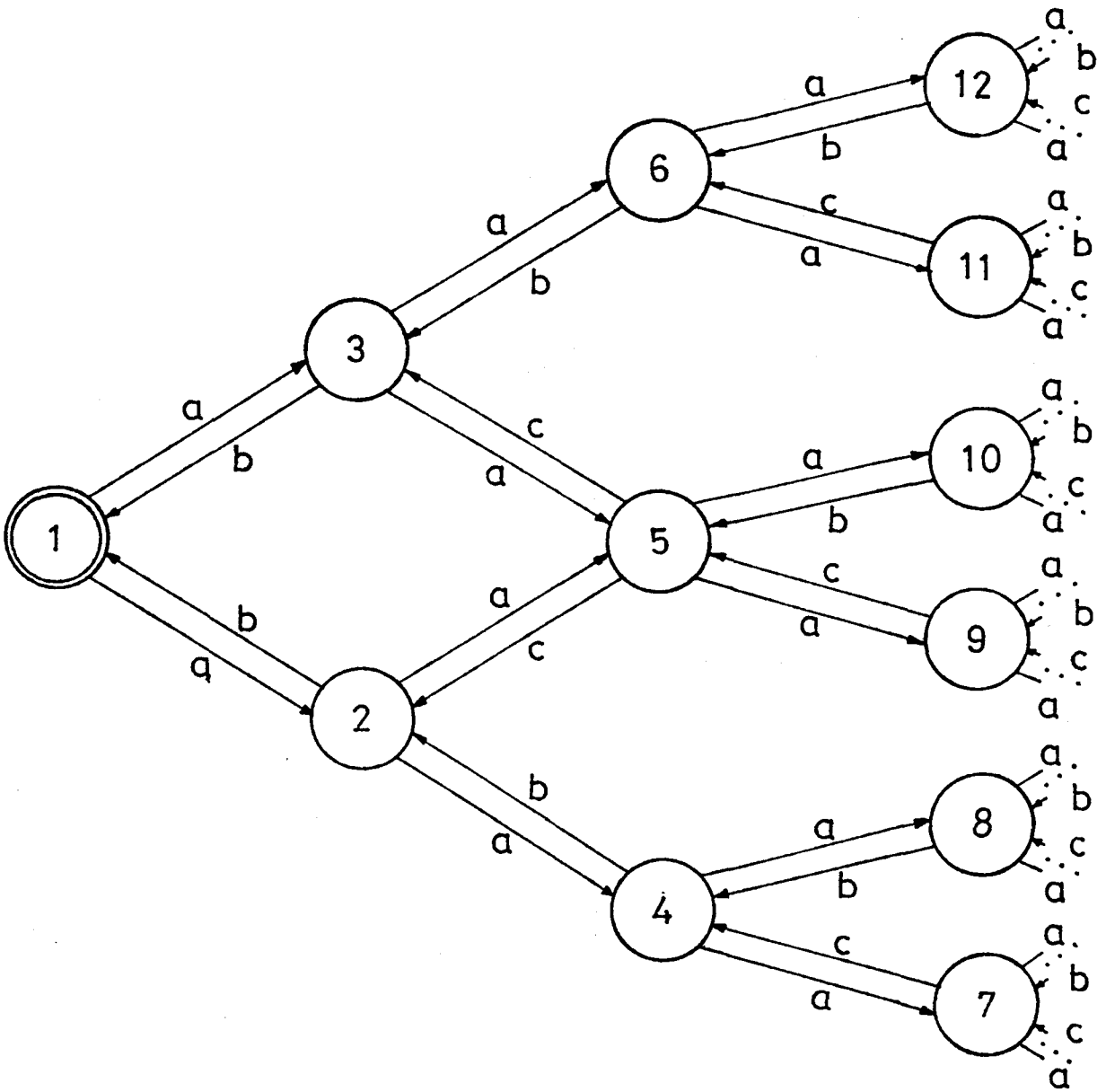
$= a(ac)b$

$= a^2b$ $(\because ac = a)$

$= a(ab)$

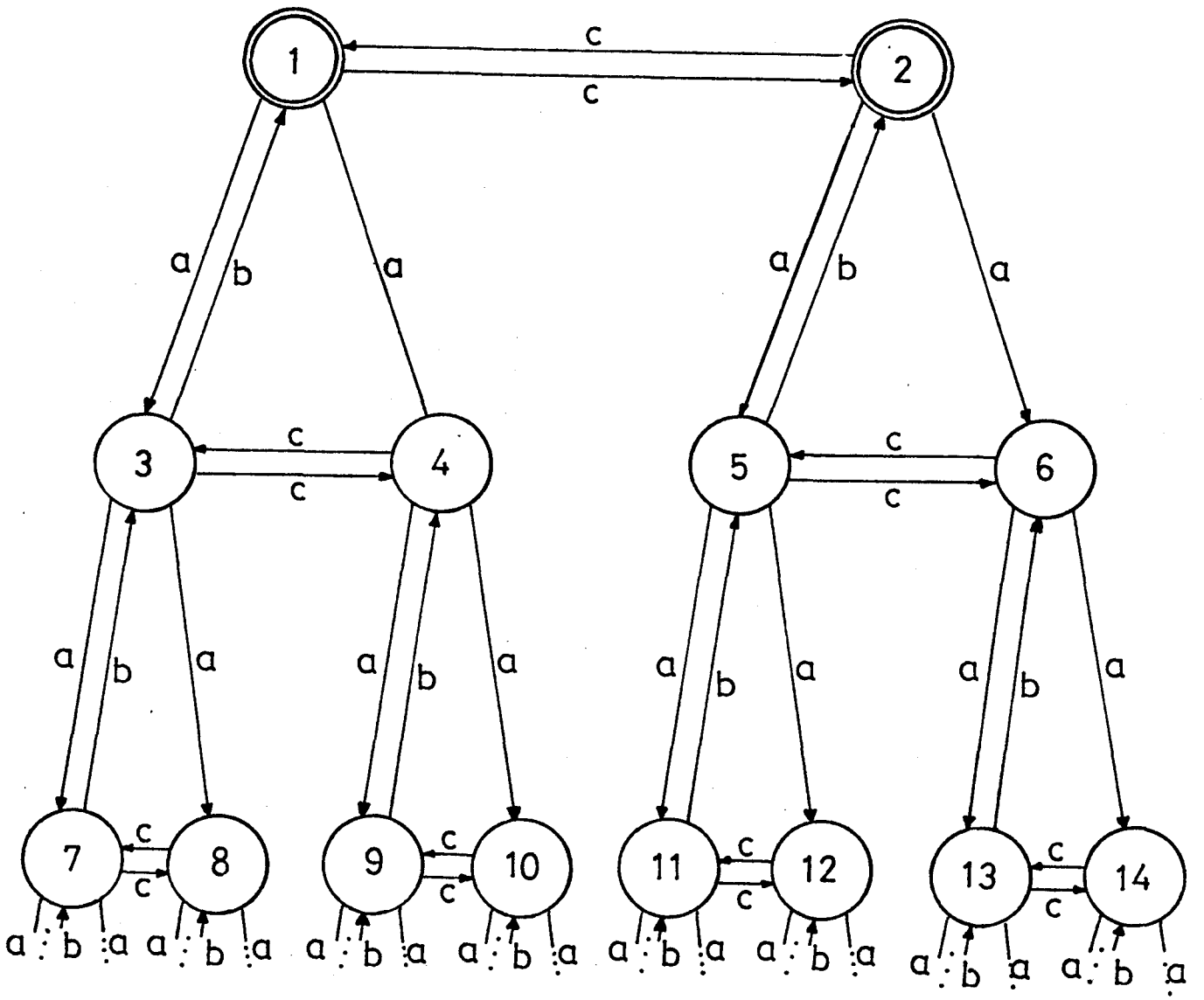
$= a$ $(\because ab = 1_c)$

Similarly every relation to the block $1ca^n$ can be shown to be equal to ca^n. That $\{1\}$ is a root block now follows. By a similar method, $\{2\}$ is shown to be also a root block. Q.E.D.



(A)

Figure 2.3(A) Two examples of rdg with root blocks



(B)

Figure 2.3(B) Two examples of rdg with root blocks

Let $\Gamma = (C, R)$ be an rdg which admits a root block $\{c_0\}$. For an arbitrary block B of Γ, there exists a relation $\xi \in \nabla_R(c_0)$ such that $B = c_0\xi$. From (2.1) we can see that all of the relations $\eta \in \nabla_R(c_0)$ such that $B = c_0\eta$ are exactly equal (as binary relations) to ξ. Hence a unique relation from the root block cell c_0 can be assigned to each block of Γ. This unique relation is designated as the "index" of the block. This notion of a "root block" is a general formulation of "root" in [17]. Several properties and features of an rdg with root blocks are provided by investigating the structures of relations in R^τ.

Lemma 2.4. Let $\Gamma = (C, R)$ be an rdg with a root block $\{c_0\}$. If $c_0 \in c_0\xi$[1] for any $\xi \in \nabla_R(c_0)$, $\xi = 1_C$.

Proof. Since $c_0 1_C = \{c_0\}$, $c_0 1_C \cap c_0\xi \neq \emptyset$. By the block partitionability of Γ, $c_0 1_C = c_0\xi$. Since $\{c_0\}$ is a root block of Γ, $\xi = 1_C$ follows from (2.1). Q.E.D.

Lemma 2.5. Let $\Gamma = (C, R)$ be an rdg with a root block $\{c_0\}$. Each $\xi \in \nabla_R(c_0)$ is total.

Proof. For each $c \in c_0\xi$, there exists an $\eta \in \nabla_R(c)$ such as $c_0 \in c\eta$ because of the strong connectedness of Γ. Therefore $c_0 \in c_0\xi\eta$ holds. From the above lemma, $\xi\eta = 1_C$ is obtained. The totality

---

[1] From Lemma 2.2, in fact, $c_0\xi = \{c_0\}$.

of 1_C ensures that ξ is total on C. Q.E.D.

Now, let $\mathcal{R}_\Gamma$ denote the set of root blocks of an rdg Γ. Then, it is verified that an rdg with root blocks has such a characteristic property as is exhibited in the following theorem.

Theorem 2.6. Let $\Gamma = (C, R)$ be an rdg with root blocks. For any $\{c_1\}$, $\{c_2\} \in \mathcal{R}_\Gamma$, every $\xi \in \nabla_R(c_1)$ such that $c_2 \in c_1\xi$ (from Lemma 2.2, in fact $c_2 = c_1\xi$) is a function.

Proof. From the strong connectedness of Γ, there exists $\eta \in \nabla_R(c_2)$ such that $c_1 \in c_2\eta$. Then, $c_1 \in c_1\xi\eta$ is obtained. Since $\{c_1\}$ is a root block of Γ, $\xi\eta = 1_C$ from Lemma 2.4. Similarly, $\eta\xi = 1_C$ since $\{c_2\}$ is also a root block of Γ. Now assume the existence of c, c_3, $c_4 \in C$ such that $c_3 \in c\xi$ and $c_4 \in c\xi$ but $c_3 \neq c_4$. Then there exists $c' \in c\xi$ such that $c'\eta = c$ invoking $\xi\eta = 1_C$. Hence, $c_3 \in c'\eta\xi$ and $c_4 \in c'\eta\xi$. While $c_3 \neq c_4$, and this contradicts $\eta\xi = 1_C$. Therefore, for each $c \in C$, there is at most one element in $c\xi$, that is to say ξ is a function on C. Q.E.D.

For example, in Fig.2.3(B), the relation $c \in R$ between the two root blocks $\{1\}$ and $\{2\}$ is a function.

It should be noted that from Theorem 2.6 or from Lemma 2.2, the

---

[2] From Lemma 2.2, in fact $c_2 = c_1\xi$.

next corollary is obtained.

Corollary 2.7. If there is a relation $r \in R$ which is not a function, there exists no rdg $\Gamma = (C, R)$ such that every cell is a root block cell.

Proof. Immediately follows from Theorem 2.6. or from Lemma 2.2.

Q.E.D.

2.3. Realization of Relational Data Graphs

This section considers the realization problem of a relational data graph on a computer memory (a random access memory device such as cores or a disk memories). Numerous techniques for realizing data structures have been developed, each having unique advantages and disadvantages. While most methods of realization can be used with arbitrary relational data graphs, our concentrating method requires high uniformities in the structure of the graph. First a general definition of a realization of a relational data graph is given. Then, a realization method by "relative block addressing" is formulated, and shows that the class of relational data graphs which can be realized by this method is exactly equal to the class of relational data graphs with root blocks.

Definition 2.6. Let Γ be an arbitrary rdg, and let A be a set of addresses such that $\#C \leqslant \#A$. Then a <u>realization</u> of Γ on A is

a pair of mapping,

$$\langle \sigma, \rho \rangle$$

where $\sigma : C \longrightarrow A$ is one-to-one total, and

$\rho : R \longrightarrow \{ r_A \mid r_A \subseteq A \times A \}^*$ is a one-to-one monoid homomorphism mapping. Thus, $1_C = 1_A$, and for $\xi, \eta \in R^\tau$,

$$(\xi\eta)\rho = (\xi\rho)(\eta\rho).$$

The pair $\langle \sigma, \rho \rangle$ satisfies the following conditions for all $c \in C$ and $r \in R$,

(i) $\quad \emptyset \neq (c\sigma)(r\rho) \subseteq C\sigma \implies cr \neq \emptyset,$

(ii) $\quad cr \neq \emptyset \implies (cr)\sigma = (c\sigma)(r\rho).$

According to this definition, if $\langle \sigma, \rho \rangle$ realizes $\Gamma = (C, R)$ on A, then $(C\sigma, R\rho)$ is isomorphic to Γ.

One of the most familiar technique for implementing data structures is the method of "relative addressing". Informally this technique is described as specifying a base address and representing the addresses of the various cells in the structure as displacements from this base address. Here for the class of bprdg's, a realization by "relative block addressing" is given, which specifies displacements for the blocks rather than the cells.

Definition 2.7. For a bprdg $\Gamma = (C, R)$, $\langle \sigma, \rho \rangle$ is called a realization of Γ by relative block addressing, if

(i) There exists a base address $a_0 \in C\sigma$,

(ii) Bijective (= one-to-one onto) displacement function $\delta : \mathcal{B}_\Gamma \longrightarrow \{\omega \in R^{\tau\rho} \mid a_0\omega \in \mathcal{B}_\Gamma\sigma\} = \Omega$ exists and for every block $B \in \mathcal{B}_\Gamma$,

$$B\sigma = a_0(B\delta).$$

Accessing to the block $B\sigma$ on A can be accomplished by knowing the displacement $B\delta$ of the block B and the base address a_0. The following result can be obtained, which implies that this realization scheme is an equivalent notion to the existence of root blocks.

Theorem 2.8. A bprdg $\Gamma = (C, R)$ is realizable by relative block addressing if and only if it admits root blocks.

Proof. Let A (a set of addresses) exist such that $\#C \leqslant \#A$.

(1) Say that Γ has a root block $\{c_0\}$. Let σ be an arbitrary total one-to-one mapping of C into A. For such σ, define the map $\rho : R^\tau \longrightarrow \{r_A \mid r_A \subseteq A \times A\}^*$ as follows:
For each $\xi \in R^\tau$, $\xi\rho = \sigma^{-1}\xi\sigma$.
Then, $1_C\rho = \sigma^{-1}1_C\sigma = 1_A$, and for each $\xi_1, \xi_2 \in R^\tau$, $(\xi_1\rho)(\xi_2\rho) = (\sigma^{-1}\xi_1\sigma)(\sigma^{-1}\xi_2\sigma) = \sigma^{-1}\xi_1\xi_2\sigma = (\xi_1\xi_2)\rho$. So the ρ specified as above is a monoid homomorphism mapping. First, we show that $\langle \sigma, \rho \rangle$ is a realization

of Γ in A.

(i) Let $\phi \neq (c\sigma)(r\rho) \subseteq C$, then $(c\sigma)(r\rho) = (c\sigma)(\sigma^{-1}r\sigma) = (cr)\sigma \subseteq C\sigma$.

So $(cr)\sigma \neq \phi$, and $cr \neq \phi$.

(ii) For any $c \in C$ and $r \in R$, let $cr \neq \phi$ hence $r \in \nabla_R(c)$. Then, $(cr)\sigma = c\sigma\sigma^{-1}r\sigma = (c\sigma)(\sigma^{-1}r\sigma) = (c\sigma)(r\rho)$.

Thus, $\langle \sigma, \rho \rangle$ realizes Γ in A.

Here we define a total one-to-one function $\beta: \mathcal{B}_\Gamma \longrightarrow R^\tau$ as follows.

For all $B \in \mathcal{B}_\Gamma$, if $c_0\xi = B$, $B\beta = \xi$.

Since β is a function, such ξ is uniquely determined for $B \in \mathcal{B}_\Gamma$. It is easily seen that a bprdg Γ with root blocks admits such β.

Now we show that $\langle \sigma, \rho \rangle$ is a realization by relative block addressing. Let β as defined above. Let $a_0 = c_0\sigma$ and $\delta = \beta\rho$. By definition of Ω, $\Omega = (\mathcal{B}_\Gamma\beta)\rho$. For each $B \in \mathcal{B}_\Gamma$, $B\sigma = c_0(B\beta)\sigma = c_0\sigma\sigma^{-1}(B\beta)\sigma = a_0(B\beta)\rho = a_0(B\beta\rho) = a_0(B\delta)$. Thus, $\langle \sigma, \rho \rangle$ is a realization by relative block addressing.

(2) Conversely, let $\langle \sigma, \rho \rangle$ be an realization of Γ by relative block addressing with base address a_0 and displacement function δ.

Consider the cell $c_0 = a_0\sigma^{-1} \in C$. Let ξ, η be arbitrary elements of $\nabla_R(c_0)$ such that $c_0\xi = c_0\eta$. Now, $(c_0\xi)\sigma = (c_0\sigma)(\xi\rho) = a_0(\xi\rho)$, and $(c_0\eta)\sigma = a_0(\eta\rho)$, therefore

(i) both $a_0(\xi\rho)$ and $a_0(\eta\rho)$ are included in $C\sigma$,

(ii) $a_0(\xi\rho) = a_0(\eta\rho)$.

Since δ is onto function, from (i),

(iii) $\exists B_1, B_2 \in \mathcal{B}_\Gamma$, $B_1\delta = \xi\rho$, $B_2\delta = \eta\rho$.

From our choice of c_0 and the definition of δ, it follows that

(iv) $(c_0\xi)\sigma = a_0(\xi\rho) = a_0(B_1\delta) = B_1\sigma$.

Since σ is one-to-one,

(v) $c_0\xi = B_1$, or equivalently, $(c_0\xi)\delta = \xi\rho$.

Similarly,

(vi) $c_0\eta = B_2$, or equivalently, $(c_0\eta)\delta = \eta\rho$.

Since $c_0\xi = c_0\eta$, and since δ is a function, $\xi\rho = \eta\rho$. Therefore, $\xi = \eta$ since ρ is one-to-one. We have thus shown $c_0 = a_0\sigma^{-1}$ to be a root block cell of Γ and the theorem is proved. Q.E.D.

2.4. Skeleton Structures of Relational Data Graphs

In a block partitionable relational data graph $\Gamma = (C, R)$ defined in section 2.2, not only the structure among the cells (C-structure), but also the structure among the blocks ($\mathcal{B}_\Gamma$-structure) can be described. One of our next concerns is to extract the $\mathcal{B}_\Gamma$-structure from Γ. To specify the $\mathcal{B}_\Gamma$-structure seperately from the C-structure would contribute to make the processing of each block itself easier.

In this section, we provide the skeleton mapping $h = \langle \varepsilon, \kappa \rangle$ of a block partitionable relational data graph $\Gamma = (C, R)$. The mapping h reveals the skeleton structure of Γ, that is $h(\Gamma) = (S, R')$. This skeleton structure $h(\Gamma)$ serves itself as $\mathcal{B}_\Gamma$-structure of Γ. ε maps each cell $c \in C$ to a single cell $s \in S$ which denotes the block containing the cell, and κ maps each relation $r \in R$ to a function r' on S. It is proved that the existence of a root block in Γ is preserved under the mapping h. In addition, for a block partitionable relational data graph Γ the condition is provided which

ensures the existence of roots in $h(\Gamma)$.

Definition 2.8. Let $\Gamma = (C, R)$ be a bprdg. The skeleton mapping of Γ is a pair of mappings defined as follows. S is an arbitrary set of cells such that $\#(\mathcal{B}_\Gamma) = \#(S)$.

$$h = \langle \varepsilon, \kappa \rangle.$$

where $\varepsilon = \gamma u$,

$\gamma: C \longrightarrow \mathcal{B}_\Gamma$ (blocking mapping),

$u: \mathcal{B}_\Gamma \longrightarrow S$ is an arbitrary one-to-one total function,

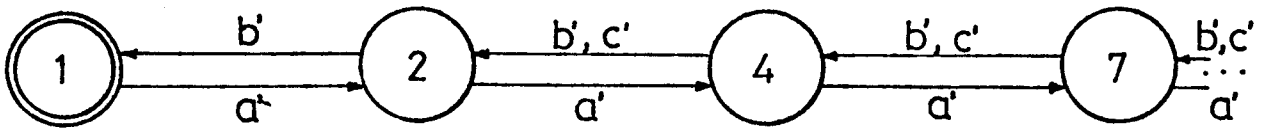
$\kappa: R \longrightarrow R' = \{ r' \mid r \in R \}$ is a total function and for any $r \in R$, $r' \in R'$ is specified according to the following rule,

$$r' = \{ \langle c_1\varepsilon, c_2\varepsilon \rangle \mid \langle c_1, c_2 \rangle \in r \}. \qquad (2.2)$$

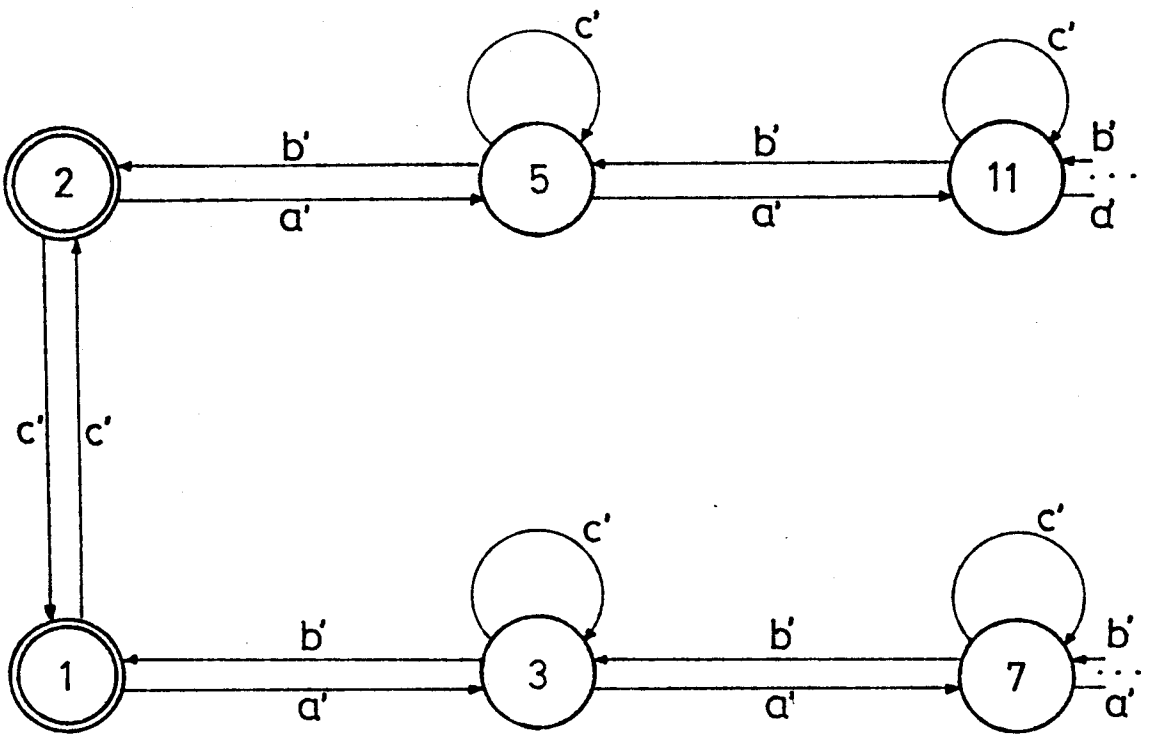
$h(\Gamma)$ is called the skeleton structure of Γ.

Example 2.4. The skeleton structures of Fig.2.3(A) and Fig.2.3(B) are afforded in Fig.2.4(A) and Fig.2.4(B), respectively.

Now let the domain of κ extend from R to R^τ. For each $\xi = r_1 r_2 \ldots r_n \in R^\tau$ (each r_i is contained in R), ξ' is defined as follows.



(A)



(B)

Figure 2.4. The skeleton structures of rdgs depicted in Fig.3

$$\xi' = \xi\kappa = r'_1 r'_2 \ldots r'_n \qquad (r_i \in R)$$

The mapping ε is generally a many-to-one function from C to S. From the definition of block partitionability, $c_1\gamma = c_2\gamma$ holds for each $c \in C$, $\xi \in \nabla_R(c)$ and $c_1, c_2 \in c\xi$. Then from the one-to-oneness of u, $c_1\varepsilon = c_2\varepsilon$ is obtained. Therefore the notation $(c\xi)\varepsilon$ is permitted and it in fact denotes $d\varepsilon$ for arbitrary cell $d \in c\xi$. Then, from Equation (2.2), for each $c \in C$ and $r \in \nabla_R(c)$,

$$(c\varepsilon)r' = (cr)\varepsilon. \qquad (2.3)$$

is obtained. Equation (2.3) insures that $r' \in R'$ is a function on C'. And the strong connectedness of $h(\Gamma)$, the fact that for each $s_1, s_2 \in S$, there exists $\xi' \in R'^{\tau}$ such that $s_1\xi' = s_2$, is guaranteed from the strong connectedness of Γ and Definition 2.8. Hence, $h(\Gamma) = (C\varepsilon, R\kappa)$ specifies an rdg $\Gamma' = (S, R')$ where each $r' \in R'$ is a function on S, and owing to the functionality of $r' \in R'$, it follows that Γ' is block partitionable. It should be noted that $h(\Gamma)$ is a data graph in the sense of [17].

Equation (2.3) is now extended in the following proposition.

Proposition 2.9. Let Γ be a bprdg. For each $c \in C$ and each $\xi \in \nabla_R(c)$,

$$(c\varepsilon)\xi' = (c\xi)\varepsilon. \qquad (2.4)$$

Proof. Let $\xi = r_1 r_2 \ldots r_m$ $(r_i \in R)$ and $\xi_j = r_1 r_2 \ldots r_j$

(the j-prefix of ξ, $1 \leqslant j \leqslant m-1$). There exists $d \in c\xi_j$ such that $r_{j+1} \in \nabla_R(d)$ for each $1 \leqslant j \leqslant m-1$. By Equation (2.3), $(d\varepsilon)r'_{j+1} = (dr_{j+1})\varepsilon$. And $d \in c\xi_j$ yields $d\varepsilon = (c\xi_j)\varepsilon$ and $dr_{j+1} \subseteq c\xi_j r_{j+1}$ yields $(dr_{j+1})\varepsilon = (c\xi_j r_{j+1})\varepsilon$, so that $(c\xi_j)\varepsilon r'_{j+1} = (c\xi_j r_{j+1})\varepsilon$. Hence, $(c\varepsilon)\xi' = (c\varepsilon)r'_1 r'_2 \ldots r'_m = (cr_1)\varepsilon r'_2 \ldots r'_m = (cr_1 r_2 \ldots r_m)\varepsilon$. $= (c\xi)\varepsilon$. This completes the proof. Q.E.D.

Here we will make sure that the extension of κ from R to R^τ is well defined. For all $r_1, \ldots, r_m, r_{m+1}, \ldots, r_n \in R$, let $r_1 r_2 \ldots r_m = r_{m+1} r_{m+2} \ldots r_n$. Then for arbitrary cell c in the domain of $r_1 \ldots r_m$ (or $r_{m+1} \ldots r_n$), $cr_1 \ldots r_m = cr_{m+1} \ldots r_n$, so $(cr_1 \ldots r_m)\varepsilon = (cr_{m+1} \ldots r_n)\varepsilon$. From Equation (2.4), $(c\varepsilon)r'_1 \ldots r'_m = (c\varepsilon)r'_{m+1} \ldots r'_n$. Since c is arbitrary, $r'_1 \ldots r'_m = r'_{m+1} \ldots r'_n$ is obtained. Therefore κ is a total function from R^τ to R'^τ. Thus the extension is well defined.

Proposition 2.10 Let $\Gamma = (C, R)$ be a bprdg and $\{b_0\} \in \mathbb{B}_\Gamma$. For $\xi, \eta \in \nabla_R(b_0)$, if $(b_0\varepsilon)\xi' = (b_0\varepsilon)\eta'$, then $b_0\xi = b_0\eta$.

Proof. If $(b_0\varepsilon)\xi' = (b_0\varepsilon)\eta'$, by Proposition 2.9 $(b_0\xi)\varepsilon = (b_0\eta)\varepsilon$, namely $(b_0\xi)\gamma u = (b_0\eta)\gamma u$ for $\varepsilon = \gamma u$. Since u is one-to-one, $(b_0\xi)\gamma = (b_0\eta)\gamma$. b_0 is a base block cell, so $b_0\xi, b_0\eta \in \mathcal{B}_\Gamma$, therefore $(b_0\xi)\gamma = b_0\xi$ and $(b_0\eta)\gamma = b_0\eta$. Hence, $b_0\xi = b_0\eta$. Q.E.D.

Proposition 2.11. Let $\Gamma = (C, R)$ be a bprdg and $h(\Gamma) = (S, R')$. For any $s \in S$ and $\xi' \in R'^\tau$, if $\xi' \in \nabla_{R'}(s)$, there exists $c \in su^{-1}$ such

that $\xi \in \nabla_R(c)$.

Proof. Immediate from Equation (2.2) Q.E.D.

Proposition 2.12. Let $\Gamma = (C, R)$ be a bprdg. For $\{b_0\} \in \mathbb{B}_\Gamma$ and $\xi' \in \nabla_{R'}(b_0')$, $\xi \in \nabla_R(b_0)$.

Proof. Immediate from Proposition 2.11, since the base block $\{b_0\}$ is a singleton set. Q.E.D.

If $h(\Gamma) = (S, R')$ has a root block $\{s_0\}$, s_0 is simply referred to as a root of $h(\Gamma)$. Such $h(\Gamma)$ is a rooted data graph in the sense of [17]. Hereafter, $c\varepsilon$ is often denoted as c'.

Theorem 2.13 If an rdg $\Gamma = (C, R)$ has a root block $\{c_0\}$, then $h(\Gamma)$ has a root c_0'.

Proof. For each $\xi', \eta' \in \nabla_R(c_0')$, let $c_0'\xi' = c_0'\eta'$. From Proposition 2.12, $\xi, \eta \in \nabla_R(c_0)$, so $c_0\xi = c_0\eta$ by Proposition 2.10. Then $\xi = \eta$, because $\{c_0\}$ is a root block of Γ. So $\xi\kappa = \eta\kappa$, i.e. $\xi' = \eta'$. Hence, c_0' is a root of $h(\Gamma)$. Q.E.D.

Example 2.5. Fig.2.3(A) has a root block $\{1\}$, while its skeleton structure afforded in Fig.2.4(A) has a root. Fig.3(B) has root blocks $\{1\}$, $\{2\}$, while its skeleton structure afforded in Fig.4(B) has roots 1 and 2.

It is made clear by the above theorem that the existence of a root block is preserved under the skeleton mapping h, but the existence of a root block in Γ is not a necessary condition to ensure that $h(\Gamma)$ has roots. The following theorem affords a necessary and sufficient condition to guarantee the existence of roots in $h(\Gamma)$.

Theorem 2.14. Let $\Gamma = (C, R)$ be a bprdg. A necessary and sufficient condition to ensure the existence of roots in $h(\Gamma)$ is that $B_0 \in \mathcal{B}_\Gamma$ exists such that for any $c_1, c_2 \in B_0$ and any $\xi \in \nabla_R(c_1)$, $\eta \in \nabla_R(c_2)$,

$$(c_1\xi)\gamma = (c_2\eta)\gamma \implies {}^{\forall}B \in \mathcal{B}_\Gamma, \ {}^{\exists}d_1, d_2 \in B, \ (d_1\xi)\gamma = (d_2\eta)\gamma. \tag{2.5}$$

Proof. (1) First, assume that $h(\Gamma)$ has a root s_0 and $B_0 = s_0u^{-1}$. For any $c_1, c_2 \in B_0$ and any $\xi \in \nabla_R(c_1)$, $\eta \in \nabla_R(c_2)$, let $(c_1\xi)\gamma = (c_2\eta)\gamma$. Then, $(c_1\xi)\varepsilon = (c_2\eta)\varepsilon$ applying the functionality of u and $\varepsilon = \gamma u$. Since $\xi \in \nabla_R(c_1)$ and $\eta \in \nabla_R(c_2)$, by Proposition 2.9 $(c_1\varepsilon)\xi' = (c_2\varepsilon)\eta'$. $c_1, c_2 \in B_0$ implies $c_1\varepsilon = c_2\varepsilon = s_0$ and since s_0 is a root of $h(\Gamma)$, $\xi' = \eta'$.

$\xi', \eta' \in \nabla_R'(s_0)$, therefore from Lemma 2.5, ξ' and η' are total on S, that is, for an arbitrary $s \in S$, $s\xi' = s\eta'$. Let $B = su^{-1}$, then the one-to-oneness of u ensures that B is also arbitrary on $\mathcal{B}_\Gamma$. Then from Proposition 2.11, $d_1, d_2 \in B$ exist such that $\xi \in \nabla_R(d_1)$, $\eta \in \nabla_R(d_2)$. Since for arbitrary $s \in S$, $s\xi' = s\eta'$, for such $d_1, d_2 \in B$

$(d_1\varepsilon)\xi' = (d_2\varepsilon)\eta'$. Then from Proposition 2.9, $(d_1\xi)\varepsilon = (d_2\eta)\varepsilon$. Since u is one-to-one, $(d_1\xi)\gamma = (d_2\eta)\gamma$ is obtained. Hence, (2.5) follows.

(2) Conversely, let $B_0 \in \mathcal{B}_{\Gamma}$ exists such that for any $c_1, c_2 \in B_0$, any $\xi \in \nabla_R(c_1)$, $\eta \in \nabla_R(c_2)$, (2.5) holds. Let $B_0 u = s_0$ and for any $\xi', \eta' \in \nabla_{R'}(s_0)$, $s_0\xi' = s_0\eta'$. Then from Proposition 2.11, $c_1, c_2 \in B_0$ exist such that $\xi \in \nabla_R(c_1)$, $\eta \in \nabla_R(c_2)$. Since $c_1\varepsilon = c_2\varepsilon = s_0$, $(c_1\varepsilon)\xi'$ $(c_2\varepsilon)\eta'$. Proposition 2.9 implies that $(c_1\xi)\varepsilon = (c_2\eta)\varepsilon$, so $(c_1\xi)\gamma = (c_2\eta)\gamma$. Then from (2.5) for arbitrary $B \in \mathcal{B}_{\Gamma}$, there exist $d_1, d_2 \in B$ such that $(d_1\xi)\gamma = (d_2\eta)\gamma$. By Proposition 2.9, $(d_1\varepsilon)\xi' = (d_2\varepsilon)\eta'$ and $d_1\varepsilon = d_2\varepsilon = Bu$. Since B is an arbitrary block in $\mathcal{B}_{\Gamma}$, Bu is an arbitrary cell in S. Therefore $\xi' = \eta'$, then $h(\Gamma)$ has a root s_0.

Thus the theorem is proved. Q.E.D.

(2.5) implies the following fact that if c_1 and c_2 are contained in the same block, then for an arbitrary block $B \in \mathcal{B}_{\Gamma}$, there exist d_1, d_2 in B such that both $d_1\xi$ and $d_2\eta$ are contained in the same block.

Example 2.6. There are no root blocks in Fig.2.5. This is assured as follows. In the figure, base block cell 1 is the only candidate for a root block. Assume that $\{1\}$ is a root block. Since $1ab = 1$, $ab = 1_C$ from Lemma 2.4. But in fact $ab = 1_C$, so $\{1\}$ is not a root block.

Although there is no root blocks in Fig.2.5, its skeleton structure given in Fig.2.6 has a root cell 1 obviously.

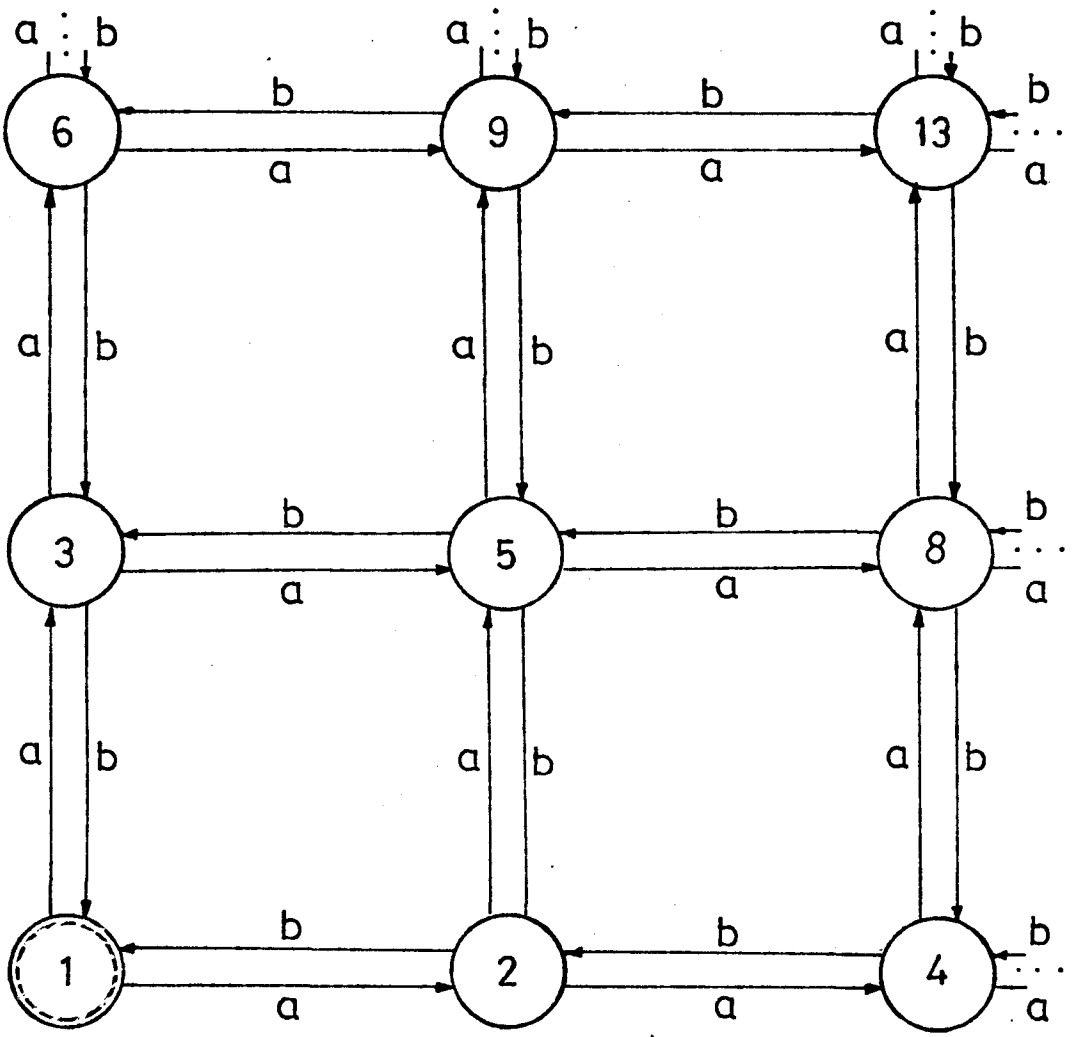


Figure 2.5. An rdg with no root blocks but its skeleton structure has a root

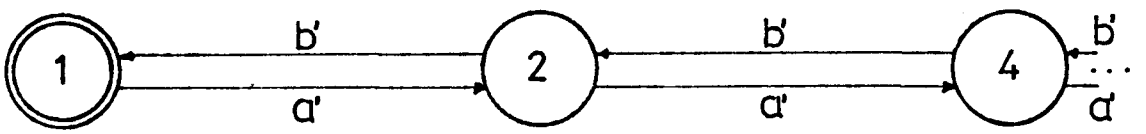


Figure 2.6. The skeleton structure of Fig.2.5

Remark. The bprdg depicted in Fig.2.7(A) has no root blocks The reason is as follows. In the figure, the set of base blocks is $\mathbb{B}_\Gamma = \{\{0\},\{-1\},\{-2\},\ldots\}$. For each $\{c\} \in \mathbb{B}_\Gamma$, assume that $\{c\}$ is a root block. From the figure, $cab = c$ so $ab = 1_C$ from Lemma 2.4. But, $1ab \neq 1$ so that $\{c\}$ is not not a root block. So Fig.2.7(A) has no root blocks.

In the skeleton structure of Fig.2.7 (A) depicted in Fig.2.7 (B), however, all the cells are obviously root cells.

Corollary 2.7 says that there exists no class of rdg's $\Gamma = (C, R)$ where every cell is a root block cell, if there exists a nonfunctional relation $r \in R$. But, we see from the above example that there is a class of rdg's $\Gamma = (C, R)$ such that every cell in the skeleton structure $h(\Gamma) = (S, R')$ is a root block cell, even if there exists a nonfunctional relation $r \in R$.

2.5. Conclusion

We have newly defined a relational data graph $\Gamma = (C, R)$ as a general description for a data structure. In a relational data graph, a set of data items can be obtained by the application of one retrieval procedure, owing to the relationality of $r \in R$.

The formulation of "block partitionability" is given, then the class of block partitionable relational data graphs with root blocks are characterized, each of whose blocks can be indexed uniquely by the link sequence from the root block cell. Several properties and features of relational data graphs with root blocks are studied; one of the striking properties is that every relation between any two root blocks is restricted

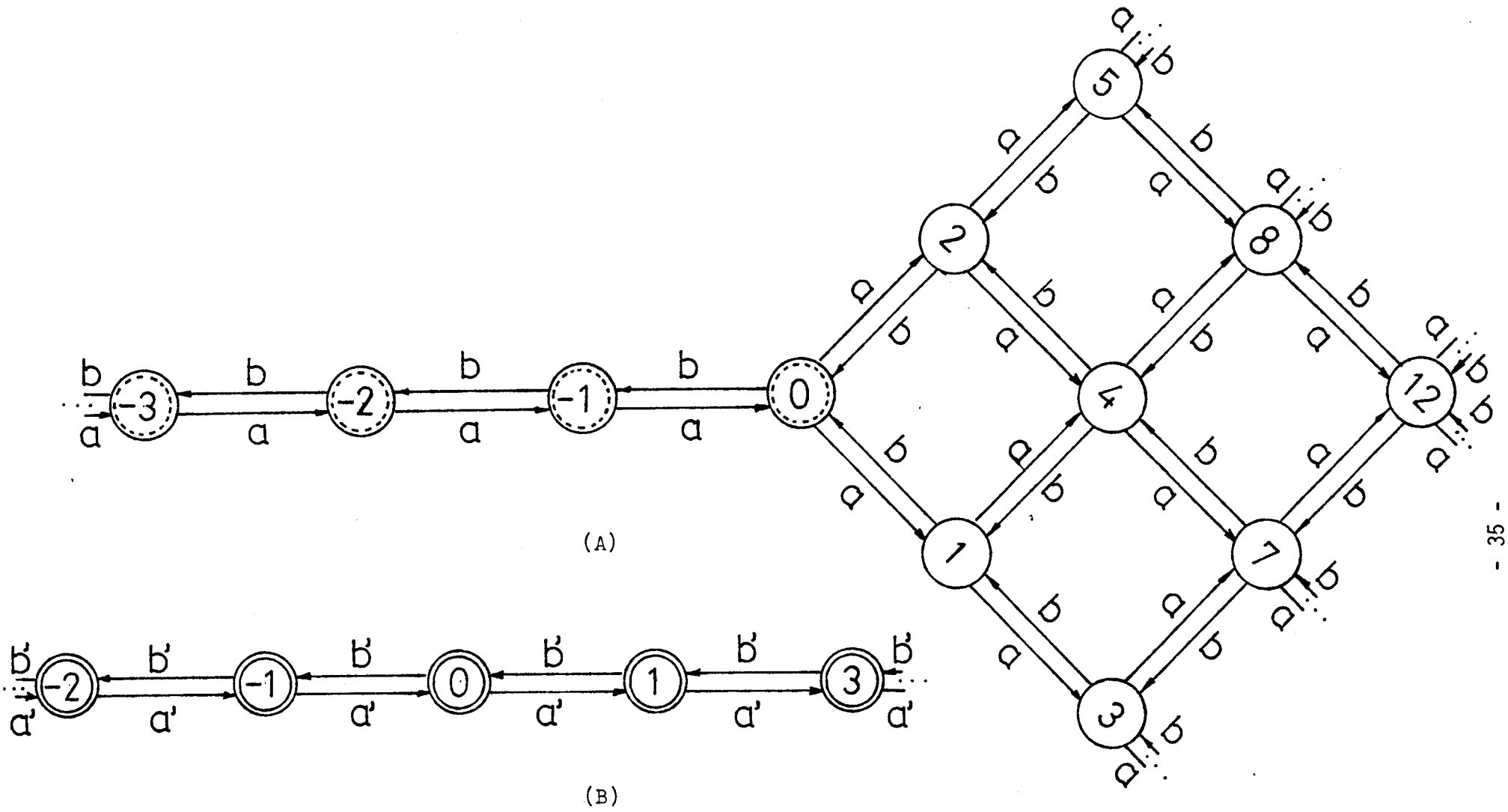


Figure 2.7. An rdg (A) such that every cell of its skeleton structure (B) is a root

to a functional relation.

Then the realization problem of a relational data graph is considered. It is shown that the class of relational data graphs which can be realized by "relative block addressing" is equal to the class of relational data graphs which admit root blocks.

Subsequently, the skeleton mapping h is introduced, which exposes the skeleton structure of a block partitionable relational data graph Γ. It is proved that if Γ has a root block, then $h(\Gamma)$ has a root. In addition, for a block partitionable relational data graph Γ the condition is provided which ensures the existence of roots in $h(\Gamma)$.

Even if Γ has a root block, so that each block in Γ can be indexed, the condition (2.1) cannot provide a sufficient uniformity for Γ to be addressed by some simple function. The next chapter is devoted to the investigation into the more stronger uniformity which would contribute to obtain an efficient addressing function.

CHAPTER 3

RELATIONAL DATA GRAPHS WITH STRONG UNIFORMITIES

3.1 Introduction

As is previousely explained, in a relational data graph Γ which admits a root block, each block of Γ can be uniquely indexed and allocated on a computer memory by relative block addressing method. But, if the function r $(r \in R)$ is not simple and thus the addressing function which maps the index set to an address set is prohibitely complex to compute, the advantages of the realization method mean little practically.

In Γ with a root block, the "connectivity relations" among cells of its any two substructures are identically the same with each other. Such uniformity, however, is not sufficient to obtain an efficient addressing function.

When we consider the structures such as complete trees or arrays which admit efficient and powerful addressing functions, we notice that these structures are constructed by repetitive patterns, and moreover each of the substructures are also complete trees or arrays.

Such strong uniformity that the shape of each substructure is same as that of the superstructure seems to be indispensable to devise a simple index set and an efficient addressing function.

In this chapter, we formulate relational data graphs with such strong uniformities. For the class of block partitionable relational data graphs Γ's, two kinds of self-embeddings (mappings from C into C)

are defined, which embed Γ itself into its substructure. So each of the substructures shares the same shape as well as connectivity.

We clarify the structural properties of block partitionable relational data graphs with such self-embeddings.

3.2 Self-embeddings of Relational Data Graphs

In this section, we provide some classes of block partitionable relational data graphs with self-embeddings θ. By θ, Γ itself is embedded into its substructure Γ_{sub}, so every operation on Γ is also applicable recursively to Γ_{sub}. Two kinds of self-embeddabilities are provided and some of their properties are studied. Here the property "θ_c-redundancy" is introduced, which is not discussed in the functional model (data graph).

Definition 3.1 Self-embedding of a bprdg $\Gamma = (C, R)$ is a total injection (one-to-one into) $\theta: C \longrightarrow C$, satisfying the condition that for an arbitrary $c \in C$ and $r \in R$,

$$cr \neq \phi \overset{3}{\Longrightarrow} (cr)\theta \subseteq (c\theta)r.$$

Γ is said to be uniformly self-embeddable if there is a $\{b_0\} \in \mathbb{B}_\Gamma$ such that for all $c \in C$ there is a self-embedding θ_c of Γ with $b_0\theta_c = c$. Such b_0 is called a base cell of Γ.

---

3 $cr \neq \phi \Longleftrightarrow r \in R$,

Example 3.1 Fig.2.5 given in Example 2.6 is a uniformly self-embeddable rdg. Fig.3.2 is also an example of a uniformly self-embeddable rdg.

Proof. (1) That Fig.2.5 is uniformly self-embeddable is proved as follows. Since Fig.2.5 is isomorphic to Fig.3.1, we do on Fig.3.1.

For cell (m,n) $(m,n \in N \cup \{0\})$, $\theta_{(m,n)}$ specified as follows is a self-embedding of Γ.

$$\theta_{(m,n)} = \{\langle (i,j), (i+m,j+n)\rangle \mid i,j \in N \cup \{0\}\}.$$

To show this, note that for $m,n \geqslant 0$, $(i,j)a \neq \emptyset$ and $((i,j)a)\theta = \{(i+1,j),(i,j+1)\}\theta = \{(i+m+1,j+n),(i+m,j+n+1)\}$ ($\theta_{(m,n)}$ is briefly written as θ), while $((i,j)\theta)a = (i+m,j+n)a = \{(i+m+1,j+n),(i+m,j+n+1)\}$, so $((i,j)a)\theta = ((i,j)\theta)a$.

For $i,j \geqslant 1$, $(i,j)b = \{(i-1,j),(i,j-1)\}$, so $((i,j)b)\theta = \{(i+m-1,j+n),(i+m,j+n-1)\}$ and $((i,j)\theta)b = (i+m,j+n)b = \{(i+m+1,j+n),(i+m,j+n-1)\}$, so $((i,j)b)\theta = ((i,j)\theta)b$.

Especially, $(0,j)b = \{(0,j-1)\}$ $(j \geqslant 1)$, so $((0,j)b)\theta = \{(m,j+n-1)\}$, while, $((0,j)\theta)b = (m,j+n)b = \{(m-1,j+n),(m,j+n-1)\}$ $(m \geqslant 1)$,

$\{(0,j+n-1)\}$ $(m = 0)$

So, $((0,j)b)\theta \subseteq ((0,j)\theta)b$. Similarly, $((i,0)b)\theta \subseteq ((i,0)\theta)b$ $(i \geqslant 0)$.

Thus for all cell (m,n) $(m,n \in N \cup \{0\})$, $\theta_{(m,n)}$ given as above is a self-embedding of Γ. Hence Γ is uniformly self-embeddable.

(2) Next, we prove the uniformly self-embeddability of Fig.3.2.

We prove on Fig.3.3 (where ε denotes a null string) which is isomorphic to Fig.3.2.

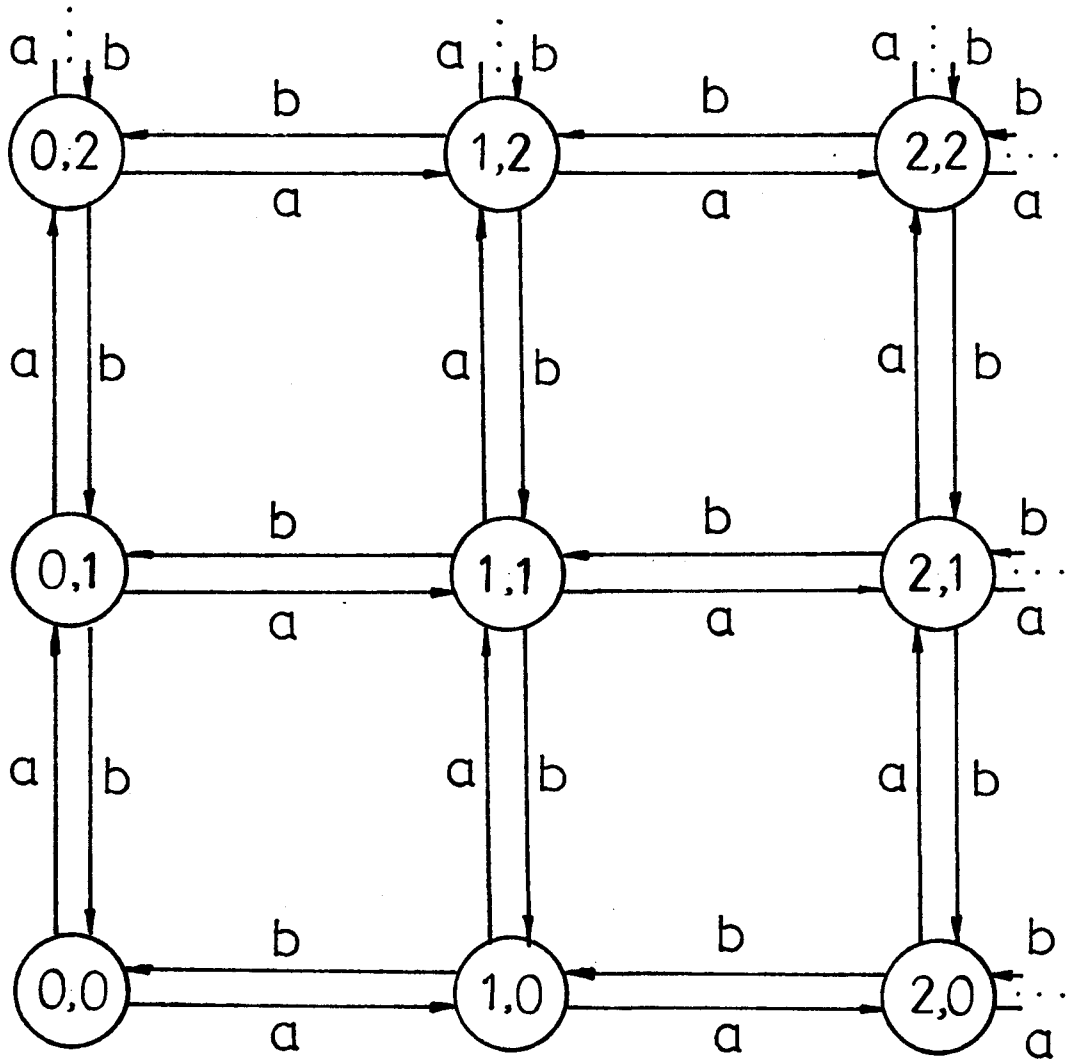


Figure 3.1. An rdg isomorphic to Fig.2.5

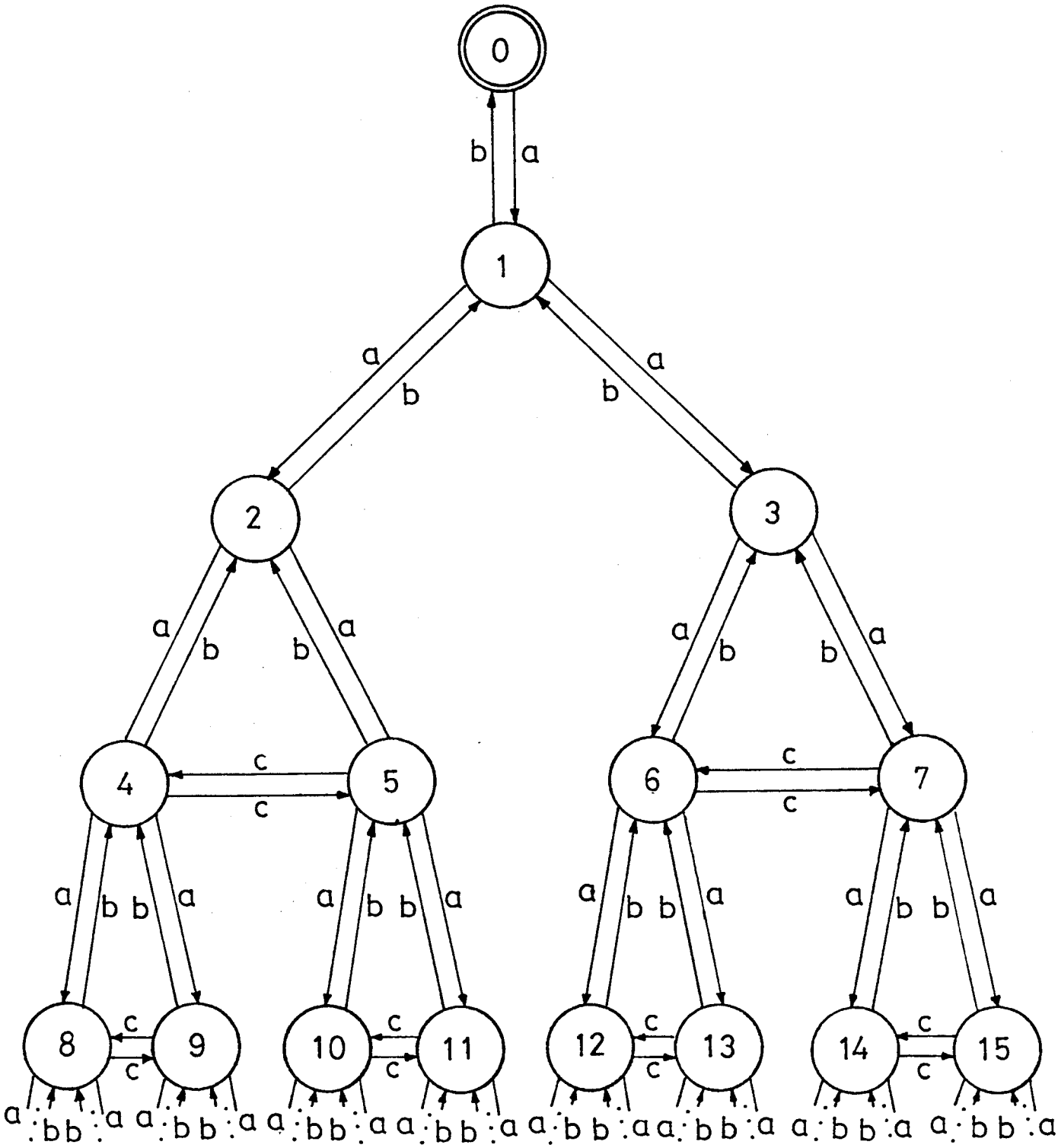


Figure 3.2. A uniformly self-embeddable rdg

First, for $\mathbb{k} \in C - \{*\}$, let

$$\theta^1_{\mathbb{k}} = \{\langle *, \mathbb{k}\rangle\} \mid \langle \mathbb{m}, \mathbb{k}0\mathbb{m}\rangle \mid \mathbb{m} \in C - \{*\}\},$$

here $\mathbb{k}0\mathbb{m}$ denotes the concatenation of strings $\mathbb{k}$, 0, $\mathbb{m}$.

$\theta^1_{\mathbb{k}}$ is a self-embedding of Fig.3.3. We prove this fact. First, $\theta^1_{\mathbb{k}}$ is obviously total injection, and

$(*a)\theta^1_{\mathbb{k}} = \{\varepsilon\}\theta^1_{\mathbb{k}} = \{\mathbb{k}0\}$, $(*\theta^1_{\mathbb{k}})a = \mathbb{k}a = \{\mathbb{k}0, \mathbb{k}1\}$, so $(*a)\theta^1_{\mathbb{k}} \subseteq (*\theta^1_{\mathbb{k}})a$.

For $\mathbb{m} \in C - \{*\}$,

$(\mathbb{m}a)\theta^1_{\mathbb{k}} = \{\mathbb{m}0, \mathbb{m}1\}\theta^1_{\mathbb{k}} = \{\mathbb{k}0\mathbb{m}0, \mathbb{k}0\mathbb{m}1\}$, $(\mathbb{m}\theta^1_{\mathbb{k}})a = (\mathbb{k}0\mathbb{m})a = \{\mathbb{k}0\mathbb{m}0, \mathbb{k}0\mathbb{m}1\}$, so $(\mathbb{m}a)\theta^1_{\mathbb{k}} = (\mathbb{m}\theta^1_{\mathbb{k}})a$.

And $(\varepsilon b)\theta^1_{\mathbb{k}} = \{*\}\theta^1_{\mathbb{k}} = \{\mathbb{k}\}$, $(\varepsilon\theta^1_{\mathbb{k}})b = \{\mathbb{k}0\}b = \{\mathbb{k}\}$, so $(\varepsilon b)\theta^1_{\mathbb{k}} = (\varepsilon\theta^1_{\mathbb{k}})b$.

For $\mathbb{m}0 \in C - \{*\}$,

$((\mathbb{m}0)b)\theta^1_{\mathbb{k}} = \{\mathbb{m}\}\theta^1_{\mathbb{k}} = \{\mathbb{k}0\mathbb{m}\}$, $((\mathbb{m}0)\theta^1_{\mathbb{k}})b = (\mathbb{k}0\mathbb{m}0)b = \{\mathbb{k}0\mathbb{m}\}$, hence $((\mathbb{m}0)b)\theta^1_{\mathbb{k}} = ((\mathbb{m}0)\theta^1_{\mathbb{k}})b$. Similarly for $\mathbb{m}1 \in C - \{*\}$, $((\mathbb{m}1)b)\theta^1_{\mathbb{k}} = ((\mathbb{m}1)\theta^1_{\mathbb{k}})b$.

Last, for $\mathbb{m}1 \in C - \{*, \varepsilon, 0, 1\}$,

$((\mathbb{m}0)c)\theta^1_{\mathbb{k}} = \{\mathbb{m}1\}\theta^1_{\mathbb{k}} = \{\mathbb{k}0\mathbb{m}1\}$, $((\mathbb{m}0)\theta^1_{\mathbb{k}})c = (\mathbb{k}0\mathbb{m}0)c = \{\mathbb{k}0\mathbb{m}1\}$, so that $((\mathbb{m}0)c)\theta^1_{\mathbb{k}} = ((\mathbb{m}0)\theta^1_{\mathbb{k}})c$. Similarly for $\mathbb{m}1 \in C - \{*, \varepsilon, 0, 1\}$, $((\mathbb{m}1)c)\theta^1_{\mathbb{k}} = ((\mathbb{m}1)\theta^1_{\mathbb{k}})c$.

In like manner, it is proved that

$$\theta^2_{\mathbb{k}} = \{\langle *, \mathbb{k}\rangle\} \cup \{\langle \mathbb{m}, \mathbb{k}\mathbb{l}\mathbb{m}\rangle \mid \mathbb{m} \in C - \{*\}\}$$

is also a self-embedding.

For *, let $\theta_* = 1_C$ (obviously a self-embedding).

Thus Fig.3.3 (Fig.3.2) is a uniformly self-embeddable rdg. Q.E.D.

In Fig.3.2, the next functions,

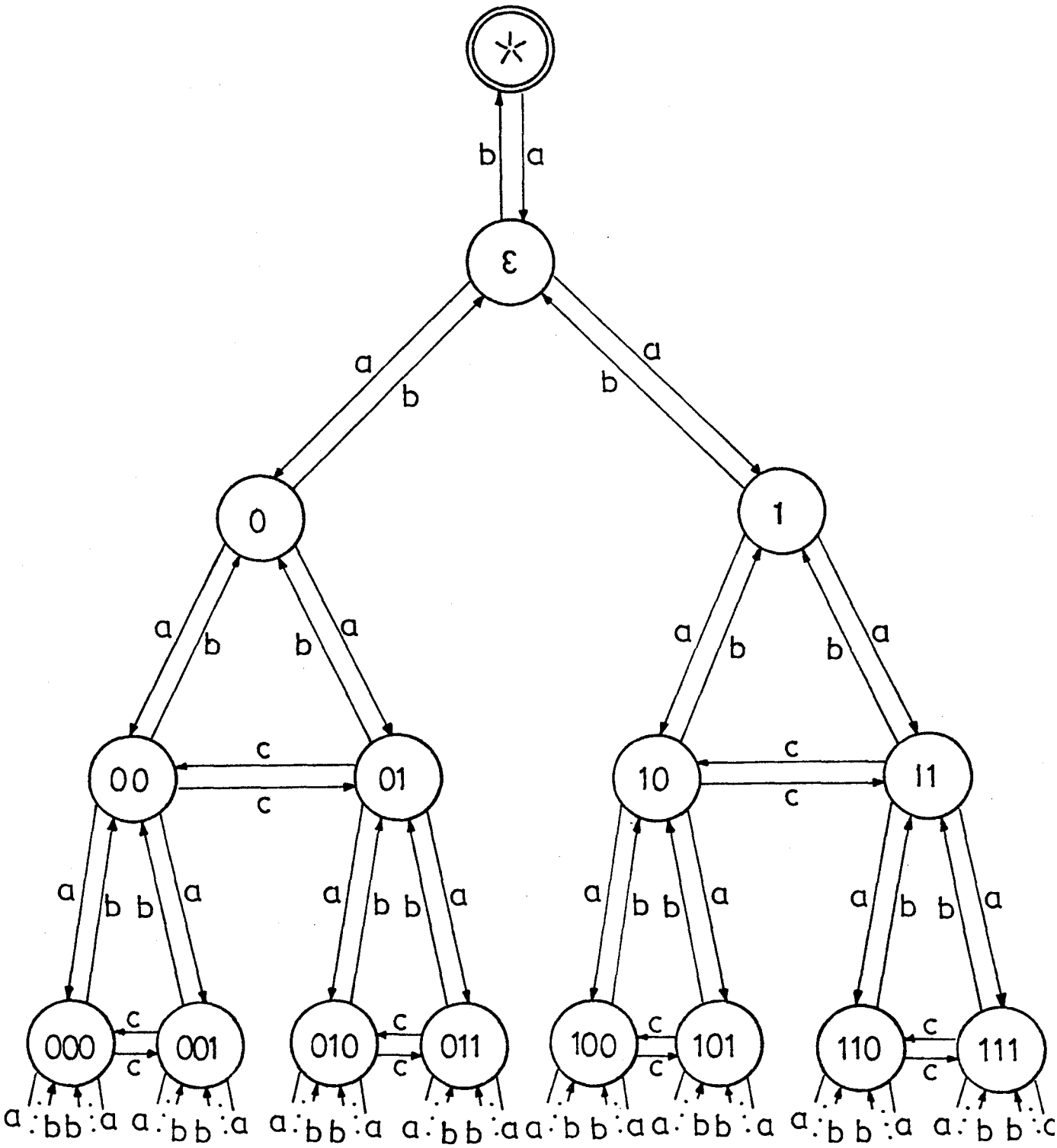


Figure 3.3. An rdg isomorphic to Fig.7

$$\theta_3^1 = \{\langle 0, 3\rangle, \langle 1, 6\rangle, \langle 2, 12\rangle, \langle 3, 13\rangle, \ldots\ldots\}$$

$$\theta_3^2 = \{\langle 0, 3\rangle, \langle 1, 7\rangle, \langle 2, 14\rangle, \langle 3, 15\rangle, \ldots\ldots\}$$

are both self-embeddings θ_3.

Same procedures as accessing, starting from base cell 0 can be also applied recursively to accessing cells $B\theta_3^1$ and $B\theta_3^2$, in fact the access starts from cell 3.

From the above example, when Γ is uniformly self-embeddable, for each $c \in C$, more than one θ_c may exist. But in the functional case, θ_c is uniquely determined for each $c \in C$.

Intuitively, one can imagine a self-embedding as taking a copy of Γ and laying it over a second copy so that every node and edge of the first copy covers a corresponding element of the second. This is reflected in the assertion that $cr \neq \phi$ implies $(cr)\theta \subseteq (c\theta)r$. For example in Fig.3.2, however, if θ_3^1 is chosen, many cells such as 7, 14, 15 and links such as $\langle 12, 13\rangle, \langle 3, 7\rangle$ fail to be covered by the first copy. Next, we will specify such uncovered cells and links.

Definition 3.2. Let Γ be uniformly self-embeddable and b_0 be a base cell of Γ. For the self-embedding θ_c specified, let $C_{\theta_c} = \bigcup_{\xi \in \nabla_R(b_0)} (b_0\theta_c)\xi$ and for each $r \in R$, let $r\theta_c = \{\langle c_1\theta_c, c_2\theta_c\rangle \mid \langle c_1, c_2\rangle \in r\}$. Each cell in $C_{\theta_c} - C\theta_c$ is called a θ_c-redundant cell. And each link

in $\bigcup_{r\in R}(r|C_{\theta_c} - r\theta_c)$ is called a θ_c-redundant link.

Now, by a straightforward induction, we can extend the condition in Definition 3.1 from $r \in R$ to $\xi \in R^*$. Then the following theorem can be obtained.

Theorem 3.1. If a bprdg $\Gamma = (C, R)$ is uniformly self-embeddable, the skeleton structure $h(\Gamma)$ has a root.

Proof. Since Γ is uniformly self-embeddable, for any $c \in C$, there exists $\{b_0\} \in B_\Gamma$ such that for all $c \in C$, there is a self-embedding θ_c of Γ with $c_0\theta_c = c$. On $h(\Gamma) = (S, R')$, for arbitrary ξ', $\eta' \in \nabla_{R'}(b')$, we assume $b'_0\xi' = b'_0\eta'$. Then, from Proposition 2.12, $\xi, \eta \in \nabla_R(b_0)$. From this and Proposition 2.10, $b_0\xi = b_0\eta$ holds, so $(b_0\xi)\gamma = (b_0\eta)\gamma$. Therefore from Theorem 2.14, it is sufficient to say that for any $c \in C$, $(c\xi)\gamma = (c\eta)\gamma$. First from the condition in Definition 3.1, for any $c \in C$, both $(b_0\xi)\theta_c \subseteq (b_0\theta_c)\xi = c\xi$, and $(b_0\eta)\theta_c \subseteq (b_0\theta_c)\eta = c\eta$ hold. $b_0\xi = b_0\eta$ and the functionality of θ_c results in that $(b_0\xi)\theta_c = (b_0\eta)\theta_c \neq \emptyset$. So $c\xi \cap c\eta \neq \emptyset$. Hence, $(c\xi)\gamma = (c\eta)\gamma$ and $b'_0 = b_0\varepsilon$ is a root of $h(\Gamma)$. Q.E.D.

Example 3.2. The skeleton structure of Fig.2.5 afforded in Fig.2.6 has a root 1. And the skeleton structure of Fig.3.2 has a root 0ε.

Next strengthening Definition 3.1, we give another self-embedding in which both θ_c-redundant cells and θ_c-redundant links are precluded.

Definition 3.3. Self-isomorphic-embedding of a bprdg $\Gamma = (C, R)$ is a total injection $\theta: C \longrightarrow C$, satisfying that for arbitrary $c \in C$ and $r \in R$,

(i) $\quad \emptyset \neq (c\theta)r \subseteq c\theta \implies cr \neq \emptyset.$

(ii) $\quad cr \neq \emptyset \implies (cr)\theta = (c\theta)r.$

Γ is said to be uniformly self-isomorphic-embeddable, if there is a $\{b_0\} \in \mathbb{B}_\Gamma$ such that for all $c \in C$ there is a self-isomorphic-embedding θ_c of Γ with $c_0\theta_c = c$. Such b_0 is called a base cell of Γ.

Example 3.3. Fig.2.3(B) is a uniformly self-isomorphic-embeddable rdg. This fact is easily verified by visual inspection of the graph.

For a uniformly self-isomorphic-embeddable rdg the next theorem follows immediately.

Theorem 3.2. A uniformly self-isomorphic-embeddable rdg $\Gamma = (C, R)$ has a root block.

Proof. Since Γ is uniformly self-isomorphic-embeddable, there exists a base cell b_0 defined in Definition 3.3. For this b_0 and arbitrary $\xi, \eta \in \nabla_R(b_0)$, let $b_0\xi = b_0\eta$. For an arbitrary $c \in C$, by the condition (ii) in Definition 3.3, $(b_0\xi)\theta_c = (b_0\theta_c)\xi = c\xi$, and $(b_0\eta)\theta_c = (b_0\theta_c)\eta = c\eta$. Since c is arbitrary, $\xi = \eta$ is obtained. This

completes the proof. Q.E.D.

From now on, let a bprdg $\Gamma = (C, R)$ be a uniformly self-embeddable rdg. By Definition 3.1., there exists a base cell b_0 such that, for all $c \in C$ there is a self-embedding θ_c of Γ with $c_0\theta_c = c$.

Now, we operate the skeleton mapping on Γ and then obtain its skeleton structure $h(\Gamma) = (S, R')$. Let θ_c be a self-embedding of Γ mentioned above, we construct from θ_c, the function θ'_c on S according to the following equation.

$$\theta'_c = \{\langle c_1\varepsilon, c_2\varepsilon\rangle \mid \langle c_1, c_2\rangle \in \theta_c\} \tag{3.1}$$

The totality of θ_c on C guarantees that θ'_c is a total function on S. Invoking Equation (3.1), for any $d \in C$,

$$(d\theta_c)\varepsilon = (d\varepsilon)\theta'_c \tag{3.2}$$

is obtained.

Fig.2.3 (B) is a uniformly self-isomorphic-embeddable rdg, but its skeleton structure, depicted in Fig.2.4 (B) is not uniformly self-isomorphic-embeddable nor uniformly self-embeddable.

Our next interest is to investigate the necessary and sufficient condition that the skeleton structure of Γ, $h(\Gamma) = (S, R')$ is uniformly self-embeddable and $b'_0 = b_0\varepsilon$ satisfies the condition of uniformly self-embeddability in Definition 3.1, namely b'_0 is a base cell of $h(\Gamma)$.

Lemma 3.3 For any $d \in C$, and any $c \in C$,

$$d\theta_c \in c\xi_d,$$

where $\xi_d \in \nabla_R(b_0)$ and $b_0\xi_d = d\gamma$.

Proof. $d \in d\gamma$ and $b_0\xi_d = d\gamma$ imply $d\theta_c \in (b_0\xi_d)\theta_c$. By the condition in Definition 3.1, $(b_0\xi_d)\theta_c \subseteq (b_0\theta_c)\xi_d$. $b_0\theta_c = c$, therefore, $d\theta_c \in c\xi_d$. Q.E.D.

Lemma 3.4. Let θ_s be a self-embedding of $h(\Gamma) = (S, R')$ obeying $b_0'\theta_s = s$. Then, for each $c \in su^{-1}$,

$$\theta_s = \theta_c'.$$

Here θ_c' comes from Equation (3.1).

Proof. For an arbitrary $s_1 \in S$ and an arbitrary $d \in s_1u^{-1}$, let $b_0\xi_d = d\gamma$. Since $d\varepsilon = (b_0\xi_d)\varepsilon$, $s_1 = (b_0\xi_d)\varepsilon = (b_0\varepsilon)\xi_d' = b_0'\xi_d'$, so that $s_1\theta_s = (b_0'\xi_d')\theta_s$. Moreover, by the condition of Definition 3.1, the functionality of ξ_d' and $b_0'\theta_s = s$, the next follows. $(b_0'\xi_d')\theta_s = (b_0'\theta_s)\xi_d' = s\xi_d'$. Therefore, $s_1\theta_s = s\xi_d'$ can be obtained. On the other hand, from Lemma 3.3, $d\theta_c \in c\xi_d$. And from Equation (3.1), $\langle d\varepsilon, (c\xi_d)\varepsilon\rangle \in \theta_c'$ holds. Hence, $s_1\theta_c' = (d\varepsilon)\theta_c' = (c\xi_d)\varepsilon = (c\varepsilon)\xi_d' = s\xi_d'$. Since, $s_1\theta_s = s\xi_d'$, $s_1\theta_s = s_1\theta_c'$ can be obtained. Here, s_1 is an arbitrary element of S, so $\theta_s = \theta_c'$. Q.E.D.

The above lemma insists that if the self-embeddings θ_s of $h(\Gamma)$ obeying $b_0'\theta_s = s$ exist, they are all obtainable from the self-embedding θ_c $(c \in su^{-1})$ according to Equation (3.1).

Next, we provide a necessary and sufficient condition to guarantee that the skeleton structure of Γ, namely $h(\Gamma)$ is uniformly self-embeddable and b_0' is a base cell of $h(\Gamma)$.

Theorem 3.5. $h(\Gamma) = (S, R')$ is a uniformly self-embeddable and $b_0' = b_0\varepsilon$ satisfies the condition of uniformly self-embeddability if and only if for an arbitrary $c \in C$, θ_c' is one-to-one.

Proof. First, let $h(\Gamma)$ be uniformly self-embeddable and for an arbitrary $s \in S$, there exists a self-embedding θ_s satisfying $b_0'\theta_s = s$. Then, by Lemma 3.4, $\theta_s = \theta_c'$. So the one-to-oneness of θ_s ensures that θ_c' is one-to-one. Since $c \in su^{-1}$ and s is arbitrary, c is also arbitrary on C.

Conversely, for an arbitrary $c \in C$, let θ_c' be one-to-one. The totality of θ_c' is assured. Now for any $s \in S$ and any $r' \in R'$, let $sr' \neq \phi$, then for an arbitrary $d \in su^{-1}$, $dr \neq \phi$. Since Γ is uniformly self-embeddable, from the condition in Definition 3.1, $(dr)\theta_c \subseteq (d\theta_c)r \subseteq ((d\theta_c)r)\gamma$. Hence, $((dr)\theta_c)\varepsilon = ((d\theta_c)r)\varepsilon$.

While, from Equation (3.2), $((dr)\theta_c)\varepsilon = ((dr)\varepsilon)\theta_c' = ((d\varepsilon)r')\theta_c' = (d'r')\theta_c' = (sr')\theta_c'$, and $((d\theta_c)r)\varepsilon = ((d\theta_c)\varepsilon)r' = ((d\varepsilon)\theta_c')r' = (s\theta_c')r'$.

Hence, $(sr')\theta_c' = (s\theta_c')r'$ is obtained. Now, it is shown that θ_c' satisfies the condition in Definition 3.1.

Therefore, $h(\Gamma)$ is uniformly self-embeddable and since $b_0\theta_c = c$,

Equation (3.2) assures $b'\theta'_c = s$. This completes the proof. Q.E.D.

On the preserving of the uniformly self-isomorphic-embeddability under the skeleton mapping h, we can immediately conclude that Theorem 3.5 also holds.

3.3 Further Characterization of Uniformly Self-Isomorphic-Embeddability

It is clear that a more simple addressing function can be employed to store a uniformly self-isomorphic-embeddable relational data graph than uniformly self-embeddable one, since it has no redundant links or redundant cells and is more homogeneously constructed.

In this section, taking notice of this usefulness underlying a uniformly self-isomorphic-embeddability, we give further characterizations of this strongest uniformity. The main result is obtained which states that a uniformly self-isomorphic-embeddable relational data graph is mainly composed of complete trees. The lemmas and the theorem obtained in this section will contribute the proof of Theorem 4.1 in the next chapter.

Throughout this section, we assume that a bprdg $\Gamma = (C, R)$ is uniformly self-isomorphic-embeddable with a base cell c_0.

Lemma 3.6. For each $a \in \nabla_R(c_0) \cap R$ and each $c \in C$, $\#(ca) = \#(c_0 a)$.

Proof. Since Γ is uniformly self-isomorphic-embeddable, there exists a self-isomorphic-embedding θ_c of Γ with $c_0\theta_c = c$, for each $c \in C$. $c_0 a \neq \phi$ and Definition 3.3-(ii) imply $(c_0 a)\theta_c = (c_0\theta_c)a = ca$.

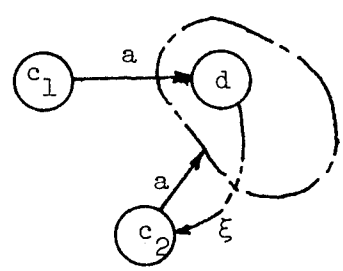
The totality and one-to-oneness of θ_c yield $\#(ca) = \#(c_0a)$. Q.E.D.

Lemma 3.7. (1) For arbitrary $a \in \nabla_R(c_0) \cap R$, Fig.3.4(A) is a forbidden subgraph of Γ.

(2) For arbitrary $a \in \nabla_R(c_0) \cap R$ and $b \in R$, Fig.3.4(B) is a forbidden subgraph of Γ.

Proof. Assuming that Γ contains Fig.3.4(A) or Fig.3.4(B) as a subgraph, we will show contradictions.

(1) Since Γ is uniformly self-isomorphic-embeddable, there exists self-isomorphic-embedding θ_{c_2} such that $c_0\theta_{c_2} = c_2$. $c_0a \neq \emptyset$ and Definition 3.3-(ii) yield $c_2a = (c_0\theta_{c_2})a = (c_0a)\theta_{c_2}$. $d \in c_2a$ and the one-to-oneness of θ_{c_2} guarantee the existence of $e \in c_0a$ such that $e\theta_{c_2} = d$. While, from the strong connectivity of Γ, there exists $\xi \in \nabla_R(e)$ such that $c_0 \in e\xi$. Moreover $(e\xi)\theta_{c_2} = (e\theta_{c_2})\xi = d\xi$ and $(e\xi)\theta_{c_2} \ni c_0\theta_{c_2} = c_2$ result in $d\xi \ni c_2$. From Theorem 3.2, $\{c_0\}$ is a root block, so by Lemma 2.4, $a\xi = 1_C$. On the other hand, since Fig.3.4(A) is supposed to be contained in Γ as a subgraph, $\langle c_1, c_2 \rangle \in a\xi$ and $c_1 \neq c_2$, so that $a\xi \neq 1_C$. This is contradiction.



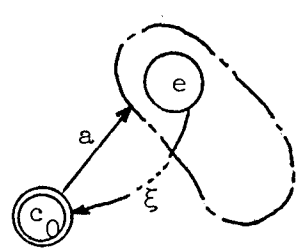


Figure 3.5(A) An explanatory figure of the proof of Lemma 3.7(1)

(2) For cell c_1, there exists a self-isomorphic-embedding θ_{c_1} $(c_0\theta_{c_1} = c_1)$.

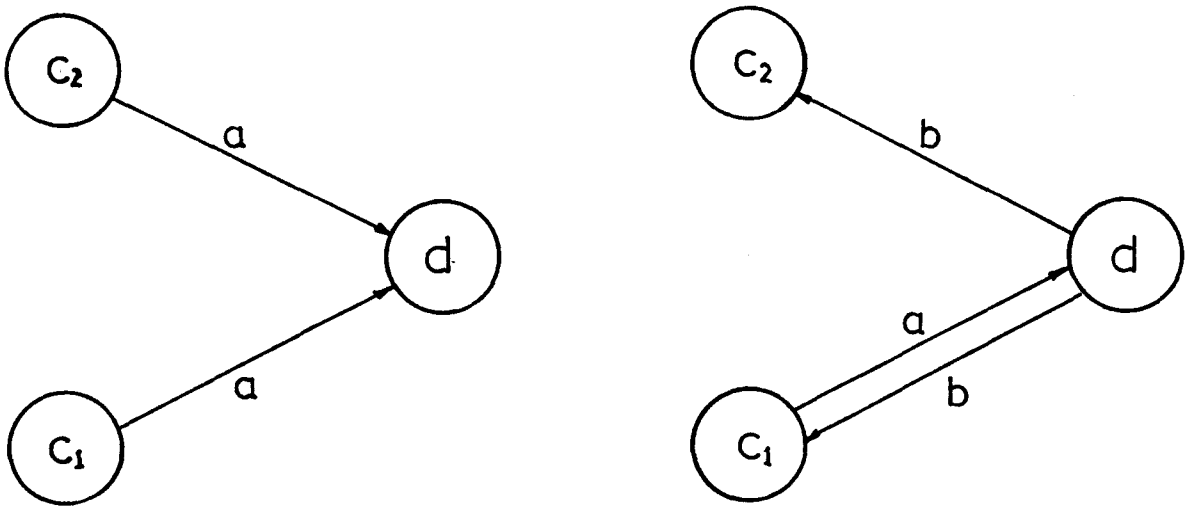


Figure 3.4. Forbidden subgraphs of a uniformly self-isomorphic-embeddable rdg

As in the proof of Part(1) the existence of $e \in c_0a$ such that $e\theta_{c_1} = d$ is ensured. Moreover, $(e\theta_{c_1})b \neq \emptyset$ since $db = (e\theta_{c_1})b$ and $db \neq \emptyset$. Hence $eb \neq \emptyset$ is obtained from Definition 3.3-(i). Therefore, applying Definition 3.3-(ii), we can obtain $(eb)\theta_{c_1} = (e\theta_{c_1})b = db \ni c_1$, so that $(eb)\theta_{c_1} \ni c_1 = c_0\theta_{c_1}$. The one-to-oneness of θ_{c_1} ensures $eb \ni c_0$, whence $c_0 \in c_0ab$. On the other hand, $\langle c_1, c_2 \rangle \in ab$ and $c_1 \neq c_2$, so that $ab \neq 1_C$. This is contradiction.

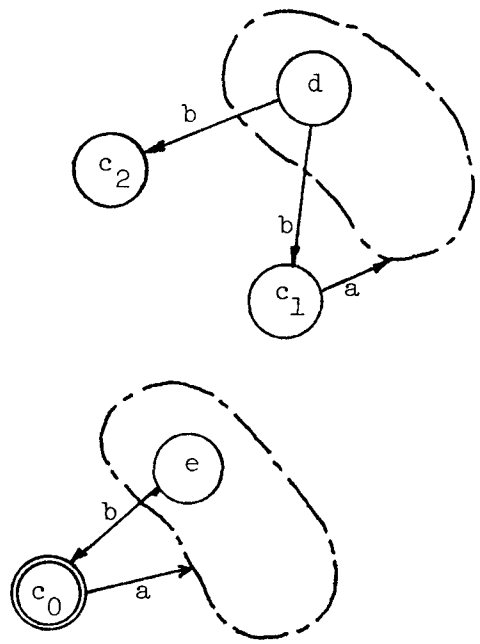


Figure 3.5(B) An explanatory figure of the proof of Lemma 3.7(2)

Part(1) and Part(2) complete the proof of the lemma. Q.E.D.

Theorem 3.8. For arbitrary $a \in \nabla_R(c_0) \cap R$, let $\#(c_0a) = k$ $(k \geqslant 2)$. If C_c^a is defined as $C_c^a = \bigcup_{i \geqslant 0} (ca^i)$, the subgraph of Γ specified by $(C_c^a, \{a/C_c^a\})$ is depicted in Fig.3.6.

Proof. By Lemma 3.6, $\#(da) = k$ is obtained for each $d \in C_c^a$. From this and Lemma 3.7-(1) (Fig.3.4(A) is a forbidden subgraph of Γ), the theorem immediately follows. Q.E.D.

Theorem 3.8 says that a uniformly self-isomorphic-embeddable rdg $\Gamma = (C, R)$ is mainly composed of tree-like structures (for example, see

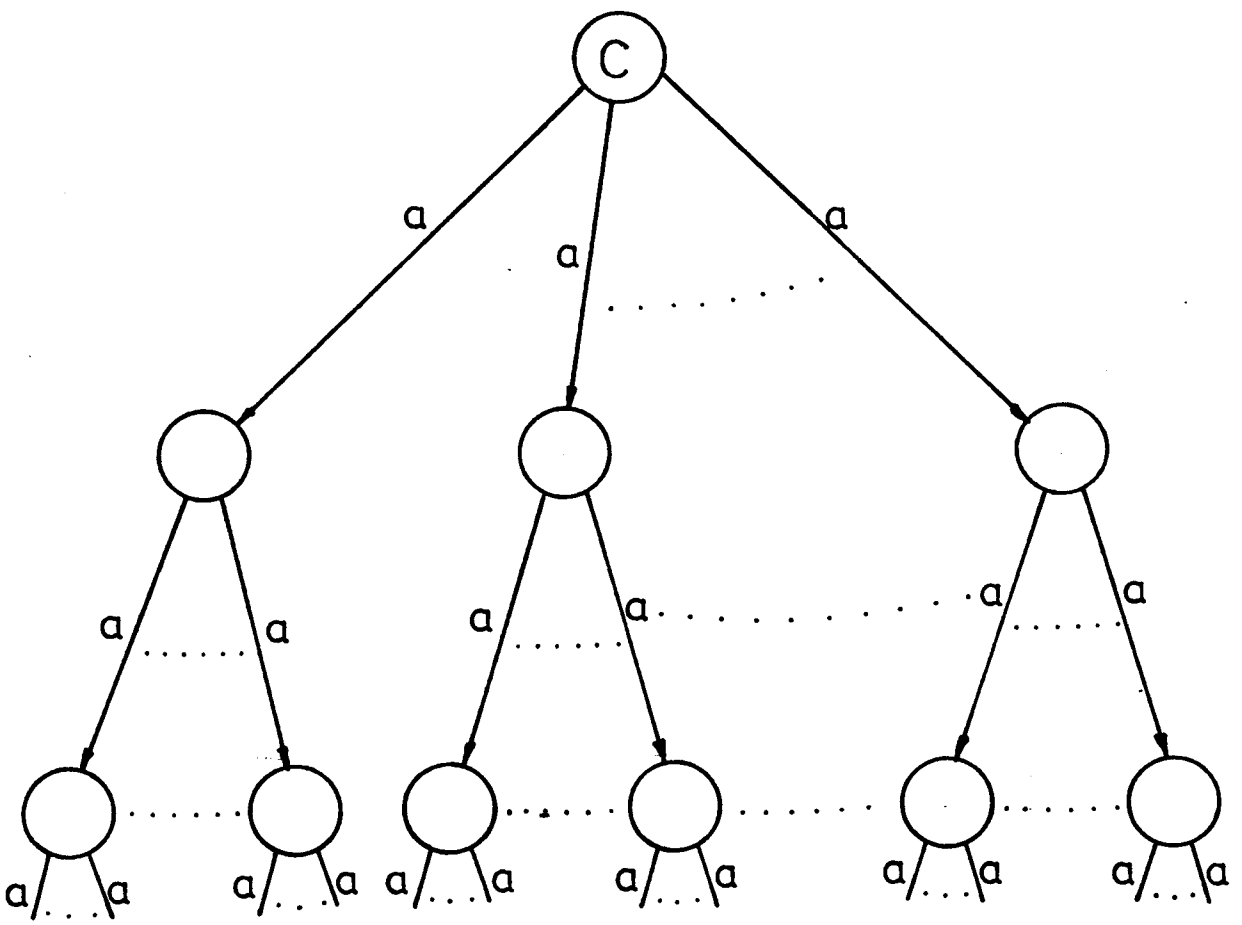


Figure 3.6. The subgraph specified by $(C_c^a, \{a/C_c^a\})$

Fig.2.3(B) or Fig.3.7) when there exists $a \in R$ such that $\#(c_0 a) \geqslant 2$. The reason why we say "tree-like" comes from the fact that the class of rdg's characterized by Theorem 3.8 contains such graphs that possess links which relate each cell to its brother cells (say, link τ in Fig.3.7). Moreover, Lemma 3.7-(2) implies that each link $\pi \in R$ which emanates from cell $c \in C$ and is incident into its father cell is a function and is never incident into the other cells (see link π in Fig.3.7). In Theorem 3.8 we have assumed $k \geqslant 2$, for in the case of $k = 1$, $(C_c^a, \{a/C_c^a\})$ may be a ring structure as is depicted in Fig.3.8(B).

3.4 Conclusion

In this chapter, for the class of block partitionable relational data graphs $\Gamma=(C, R)$, two kinds of self-embeddabilities are formulated. These uniformities, especially uniformly self-isomorphic-embeddability, might be essential in establishing an efficient addressing function.

For Γ with these uniformities, we clarify the existence of root blocks in Γ or $h(\Gamma)$. Subsequently, it is shown that the two kinds of self-embeddabilities are both preserved under the skeleton mapping h if every θ_c' on $C\varepsilon$ constructed from each kind of self-embedding θ_c of Γ respectively is one-to-one. Lastly, concentrating on the structural advantages of a uniformly self-isomorphic-embeddable relational data graph Γ, we study its properties in detail. These studies result in that a relational data graph with the uniformity is mainly composed of complete trees except a certain case.

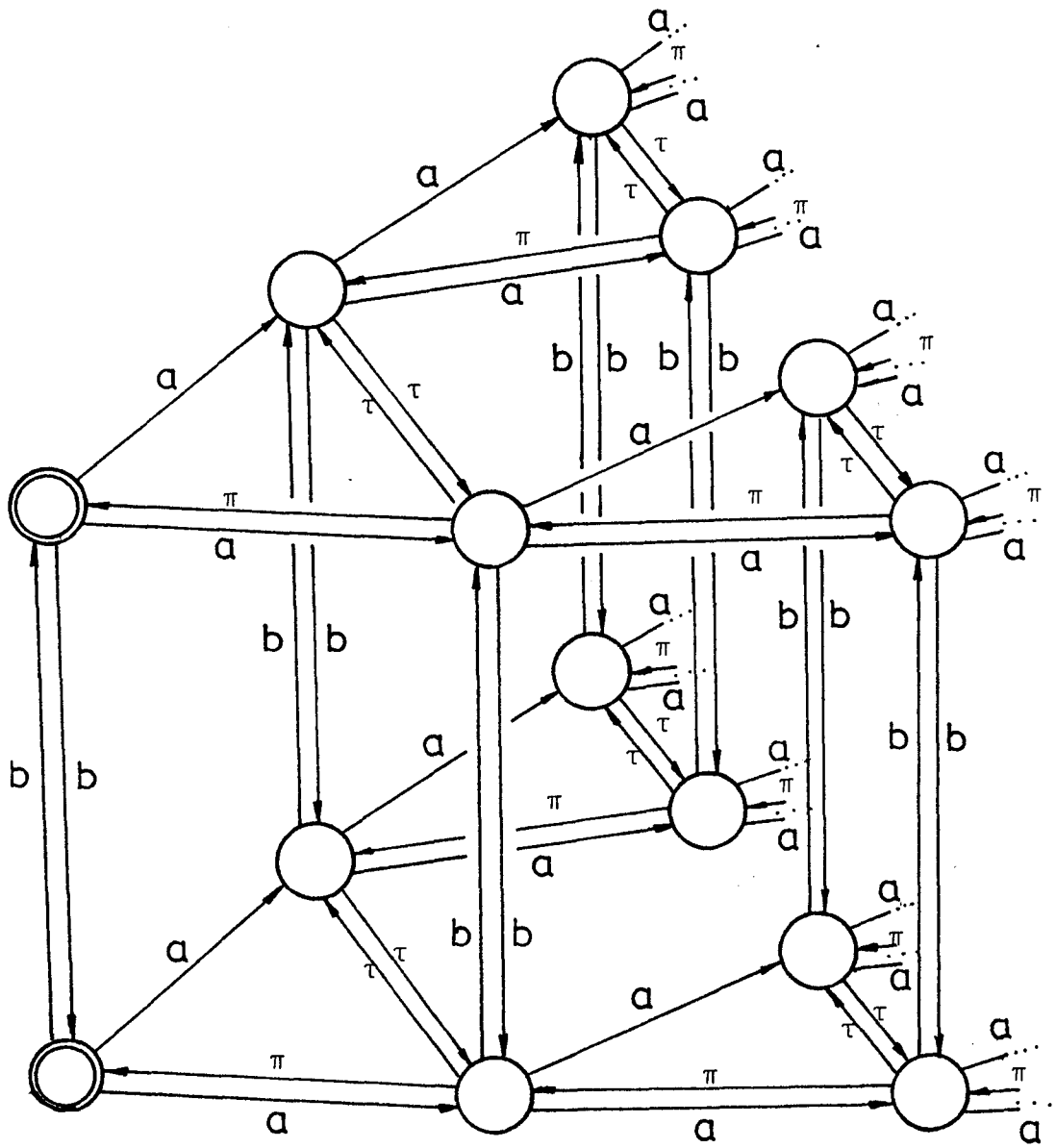
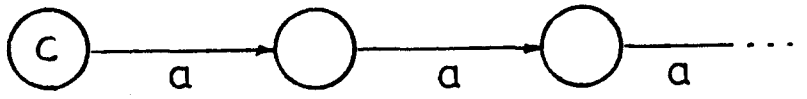
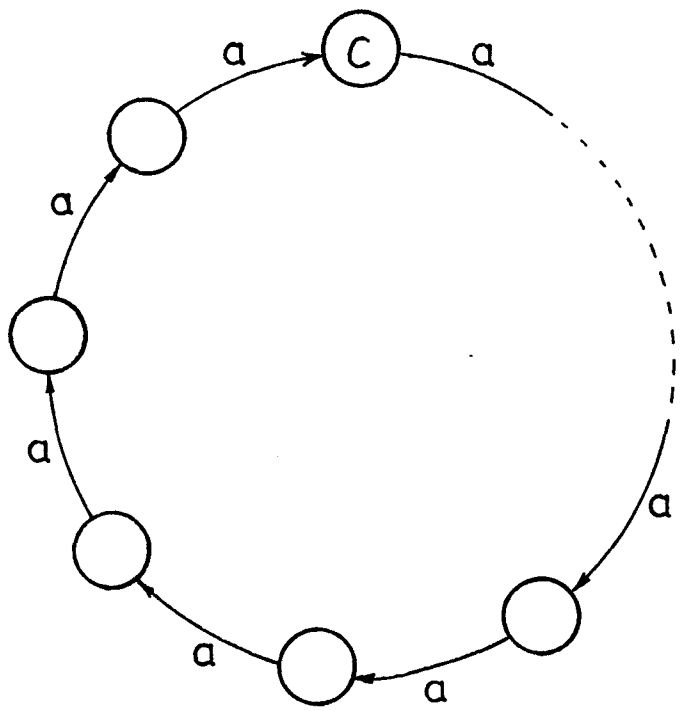


Figure 3.7. Another example of a uniformly self-isomorphic-embeddable rdg



(A)



(B)

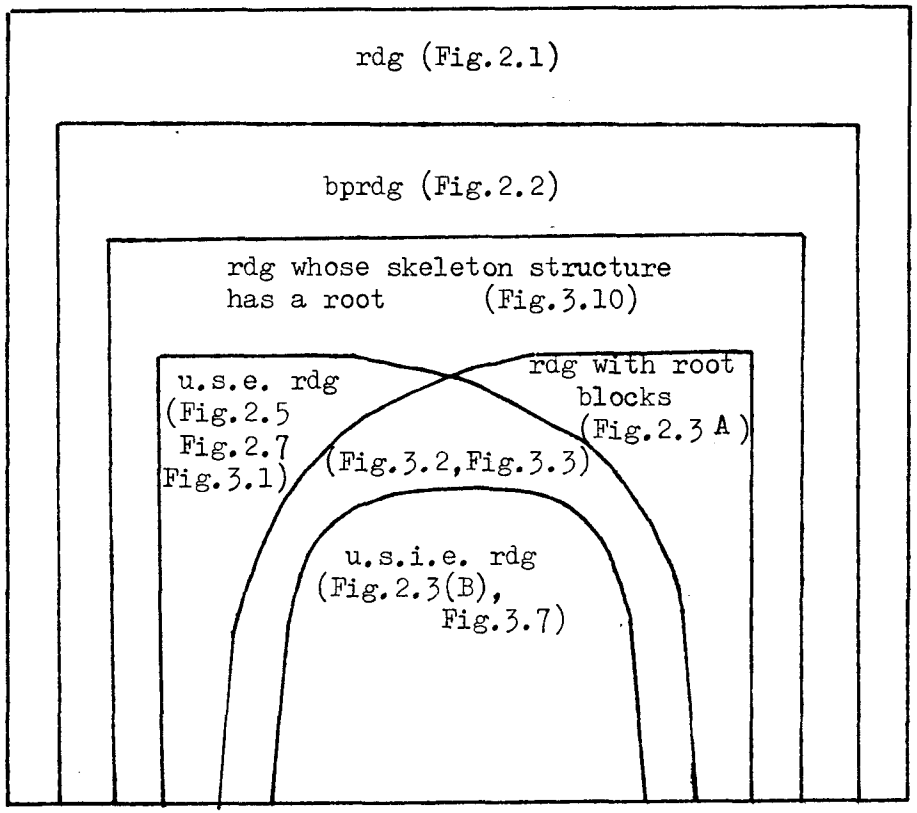
Figure 3.8. The subgraph specified by $(C_c^a, \{a/C_c^a\})$ in the case of $k = 1$

Now we summarize schematically in Fig.3.9(A), the hierarchy of the uniformities of relational data graphs hitherto developed. Fig.3.9(B) is the hierarchy restricted to the functional case (data graph). We supply in Fig.3.10, an example of Γ whose skeleton structure has a root, but Γ itself has no root blocks nor be uniformly self-embeddable.

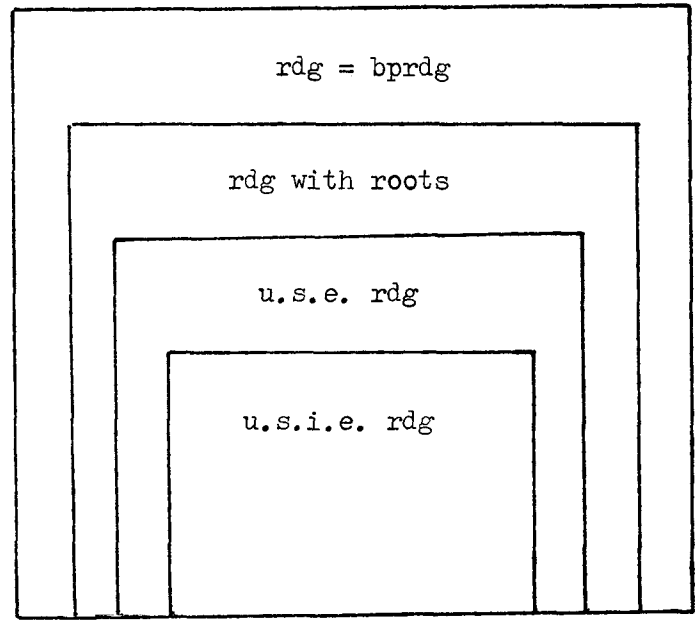
As is stated previously, a uniformly self-isomorphic-embeddable relational data graph does enjoy advantageous structure for constructing addressing functions. But the reason that we say "advantageous" is based on much intuitive ground, and in fact for Γ with the uniformity, we did not discuss generally its actual addressing functions.

It seems a much difficult task to treat addressing functions from a general point of view. In fact, the form of an addressing function is highly contextual; it is heavily dependent on the shape of the individual data structures.

In the next chapter, with the theoretical basis developed thus far, we define new data structures and construct their addressing functions from some criteria.



(A)



(B)

Figure 3.9. (A) The hierarchy of the uniformities of rdg's (B) The hierarchy restricted to the functional case

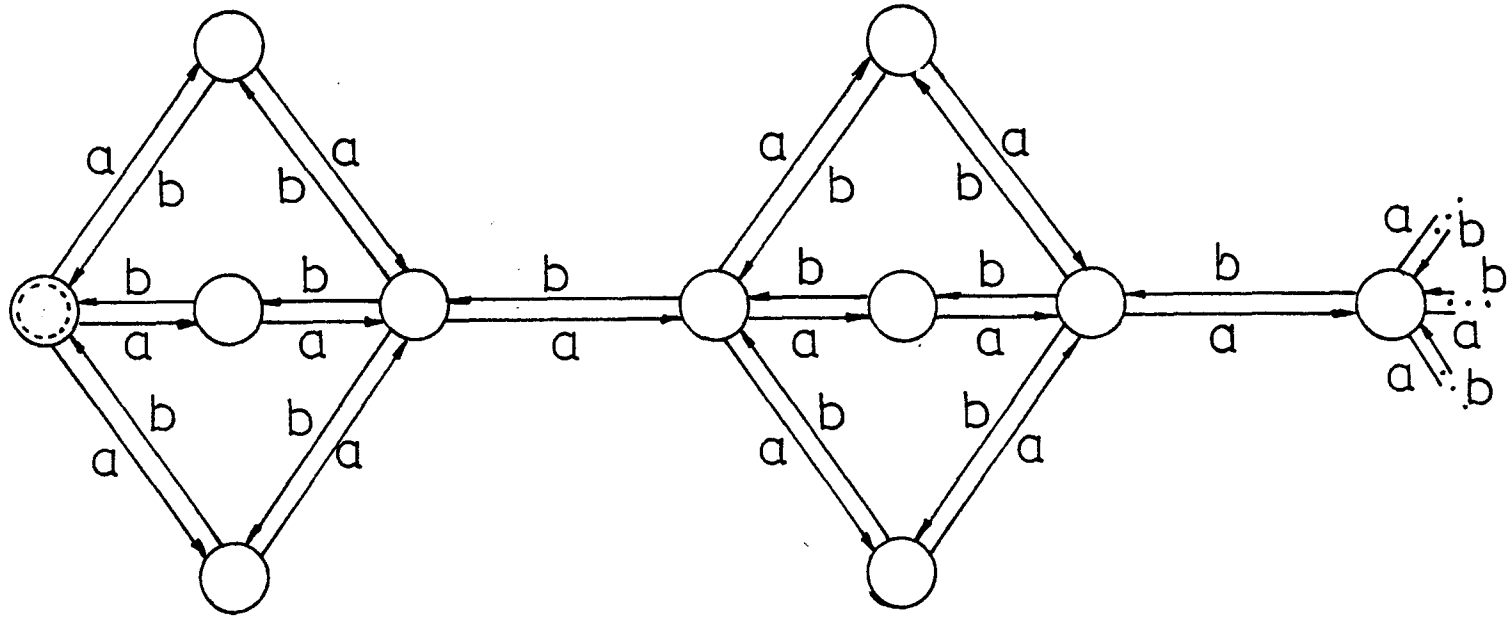


Figure 3.10. An example of Γ whose skeleton structure has a root, but Γ itself has no root blocks nor be uniformly self-embeddable

CHAPTER 4

TREE-ARRAY COMPOSITE STRUCTURES AND THEIR ADDRESSING FUNCTIONS

4.1 Introduction

Few classes of data structures are as well understood or as widely used as arrays and/or trees. The most high level programming languages offer some array processing facilities; indeed, certain languages such as Fortran and APL have been designed with arrays as the basic data structures. While, in some high level languages such as PL/I and COBOL we can use trees for representing the hierarchical structures, which are often introduced on data. These structures seems to be useful and powerful in the sense that they are often required for the efficient solutions of most problems with their associated data, and what is more, the specifications of data elements, i.e. indexing, are done in quite simple ways.

As for the realization method of these structures, arrays are the most familiar data structures usually implemented by addressing functions. We can find much research in concern with addressing functions of arrays (for example, [4] , [9], [32] , [22-28]). Trees except complete ones are (for a variety of efficiency-related reason) seldom stored by direct-access method.

As is mentioned previously, the construction of an addressing function is highly contextual, so it seems that practically significant results cannot be expected, if one treats the construction process from a general point of view.

In this chapter, rather than continue the abstract and general treatment of data structures which may admit addressing functions, we shall give investigations of specific family of directly accessible data structures, with the theoretical basis developed in the previous two chapters.

We present a new class $\mathbf{\Gamma}$ of data structures allocated by addressing functions. $\Gamma = (C, F) \in \mathbf{\Gamma}$, which has a composite structures of trees and arrays (we call it a TA-structure), can be naturally introduced and defined in terms of a d-dimensional array A_d and uniformly self-isomorphic -embeddability. We show that a TA-structure Γ is uniquely constructed from A_d for the specified "dimensionality" of Γ. Followingly we specify the various structures "sliced out" from TA-structure Γ, and for those structures we describe the several indexing methods which reflect both the string type (tree indexing) and integer tuple type (array indexing) indices, so that sharing their own characteristics and advantages. Moreover we establish a few kinds of addressing functions for the index set specified, according to the case that "tree-oriented" processing is mainly required and "array-oriented" processing is mainly required, and according to the case that access time is primary prefered and memory utilization is primary prefered.

4.2 TA-structures

In this section, firstly we define graphically the d-dimensional array A_d, and then define a TA-structure $\Gamma = (C, F)$ in terms of a uniformly self-isomorphic-embeddable relational data graph $\mathbf{\Gamma}$ whose

skeleton structure $h(\Gamma)$ is A_d. Secondly, we prove that a TA-structure Γ is uniquely constructed from A_d for the specified "dimensionality" of Γ. Subsequently, a TA-structure $\Gamma^{(m,n)}$ "sliced out" from Γ is specified, and then a tree Γ_t and an array Γ_a sliced out from $\Gamma^{(m,n)}$ are presented.

Definition 4.1 The d-dimensional array is the ordered pair $A_d = (S, M)$, where

(i) S is the set of cells specified as $S = N^d$, where N is the set of positive integers and N^d is the set of d-tuples of positive integers.

(ii) $M = G_s \cup G_p$, where both G_s and G_p are the following sets of d transformations of S. Let N_d denote the set $N_d = \{1,\ldots,d\}$.

$$G_s = \{b_i \mid i \in N_d\} \qquad \text{(successor links)}$$

$$G_p = \{\nu_i \mid i \in N_d\} \qquad \text{(predecessor links)}$$

For each $s = \langle s_1,\ldots,s_d\rangle \in S$,

$$(sb_i)_j = \begin{cases} s_j + 1 & (j = i) \\ s_j & (j \neq i), \end{cases} \qquad (s\nu_i)_j = \begin{cases} \begin{cases} s_j - 1 & (j = i) \\ s_j & (j \neq i) \end{cases} & (s_i = 1) \\ s & (s_i \neq 1) \end{cases}$$

Example 4.1 Fig.4.1 is the two dimensional array A_2, where $G_s = \{a, b\}$, and $G_p = \{\pi, \nu\}$.

In order to be able to compute the address assigned to each cell $s \in S$, it is sufficient that A_d has the structure determined by the successor links G_p. But, for example, in Fig.4.1, the transition from cell $\langle i, j\rangle$ to $\langle i - 1, j\rangle$ or $\langle i, j - 1\rangle$ cannot be accomplished if A_d has no other links than G_s. G_p is the set of links provided for the ease of transitions along arbitrary direction.

It should be also noted that A_d is infinite along each axis.

Definition 4.2 A <u>$\langle T, A\rangle$-structure</u> is a uniformly self-isomorphic -embeddable rdg $\Gamma = (C, R)$ subjecting to the following conditions. c_0 is a base cell of Γ.

(i) The skeleton structure $h(\Gamma)$ is the d-dimensional array $A_d = (S, M)$ $(M = G_s \cup G_p)$. Especially, for cell $\mathbb{1} = \langle 1,\ldots,1\rangle \in S$, $\mathbb{1}\varepsilon^{-1} = c_0$.

(ii) At most one element in $G_s\kappa^{-1}$ is a relation which is not a function. The elements in $G_p\kappa^{-1}$ are all functions.

Note that ε and κ come from Definition 2.8.

Clearly, a $\langle T, A\rangle$-structure Γ is the d-dimensional array if in (ii) all elements in $G_s\kappa^{-1}$ are functional relations.

If a relation $r \in G_s\kappa^{-1} \subseteq R$ exists which is not a function, we shall denote $r\kappa \in F$ by "a" on the sequel.

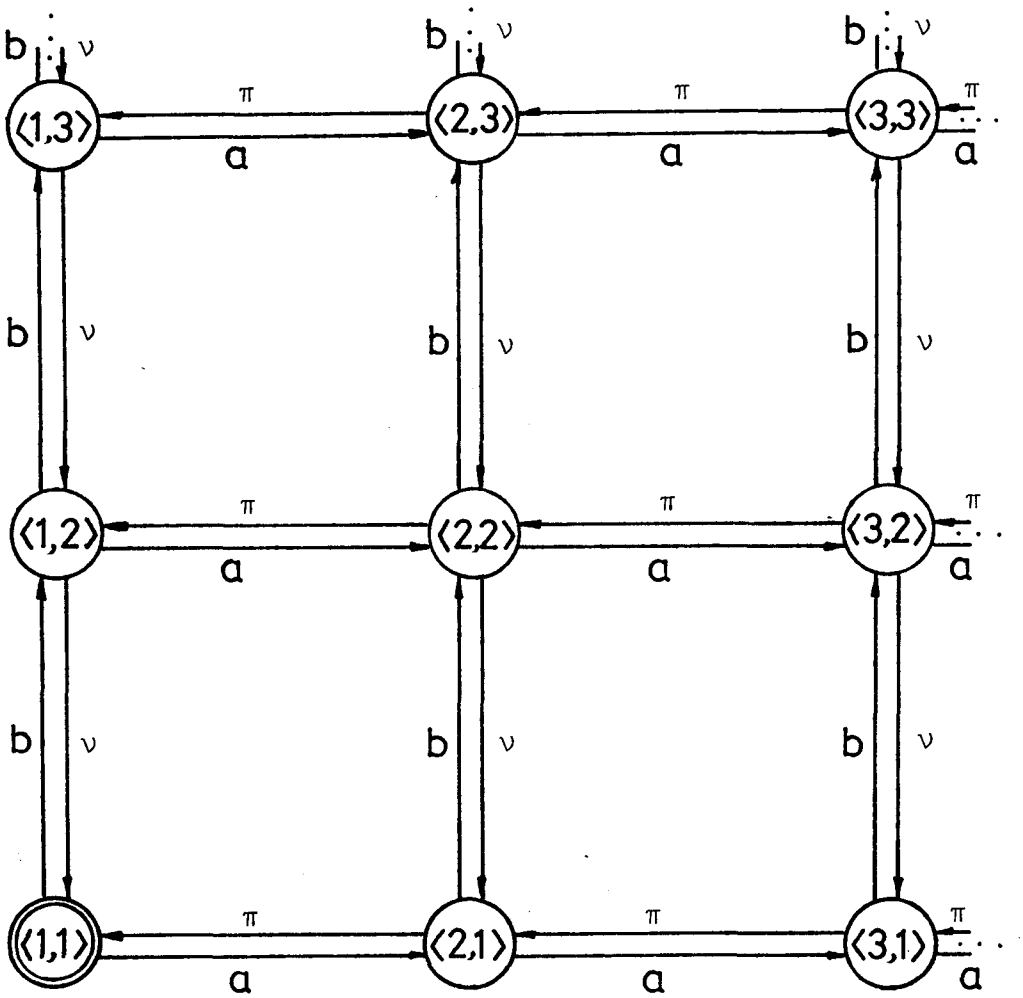


Figure 4.1. The two dimensional array

When $\#(c_0(a\kappa^{-1})) = k$, we call $\langle\langle d, k\rangle\rangle$ the dimensionality of Γ. According to this format, the dimensionality of the d-dimensional array A_d is represented as $\langle\langle d, 1\rangle\rangle$. It should be noted that $R = (G_s\kappa^{-1}) \cup (G_p\kappa^{-1})$.

Example 4.2. The $\langle T, A\rangle$-structure of dimensionality $\langle\langle 2, 2\rangle\rangle$ is shown in Fig.4.2.

Definition 4.3. Let $\Gamma = (C, R)$ be the $\langle T, A\rangle$-structure of dimensionality $\langle\langle d, k\rangle\rangle$ and c_0 be the base cell of Γ. For each $c \in C$, we have $\#(c(a\kappa^{-1})) = k$ from Lemma 3.6. Let us denote the cells in $c(a\kappa^{-1})$ as $c(a\kappa^{-1}) = \{d_0^c, d_1^c, \ldots, d_{k-1}^c\}$. By "functionizing" the relation $a\kappa^{-1}$ as $ca_i = d_i^c$ $(0 \leqslant i \leqslant k - 1)$, we can obtain a functional graph $\Gamma = (C, F)$ where each element in F is a function; the functionality of each element other than a_i $(0 \leqslant i \leqslant k - 1)$ is guaranteed by Definition 4.2-(ii). It is obvious that Γ is uniquely specified for the given Γ. Γ is called the TA-structure induced from Γ and c_0 is also called a base cell of Γ. $\langle\langle d, k\rangle\rangle$ is again called the dimensionality of Γ.

According to the above functionizing operation, the relation $a\kappa^{-1}$ is partitioned into the following k functions, namely, $a\kappa^{-1} = a_0 \cup a_1 \cup \cdots \cup a_{k-1}$. Let a be defined as $a = \{a_0, a_1, \ldots, a_{k-1}\}$, then $F = (R - \{a\kappa^{-1}\}) \cup a$.

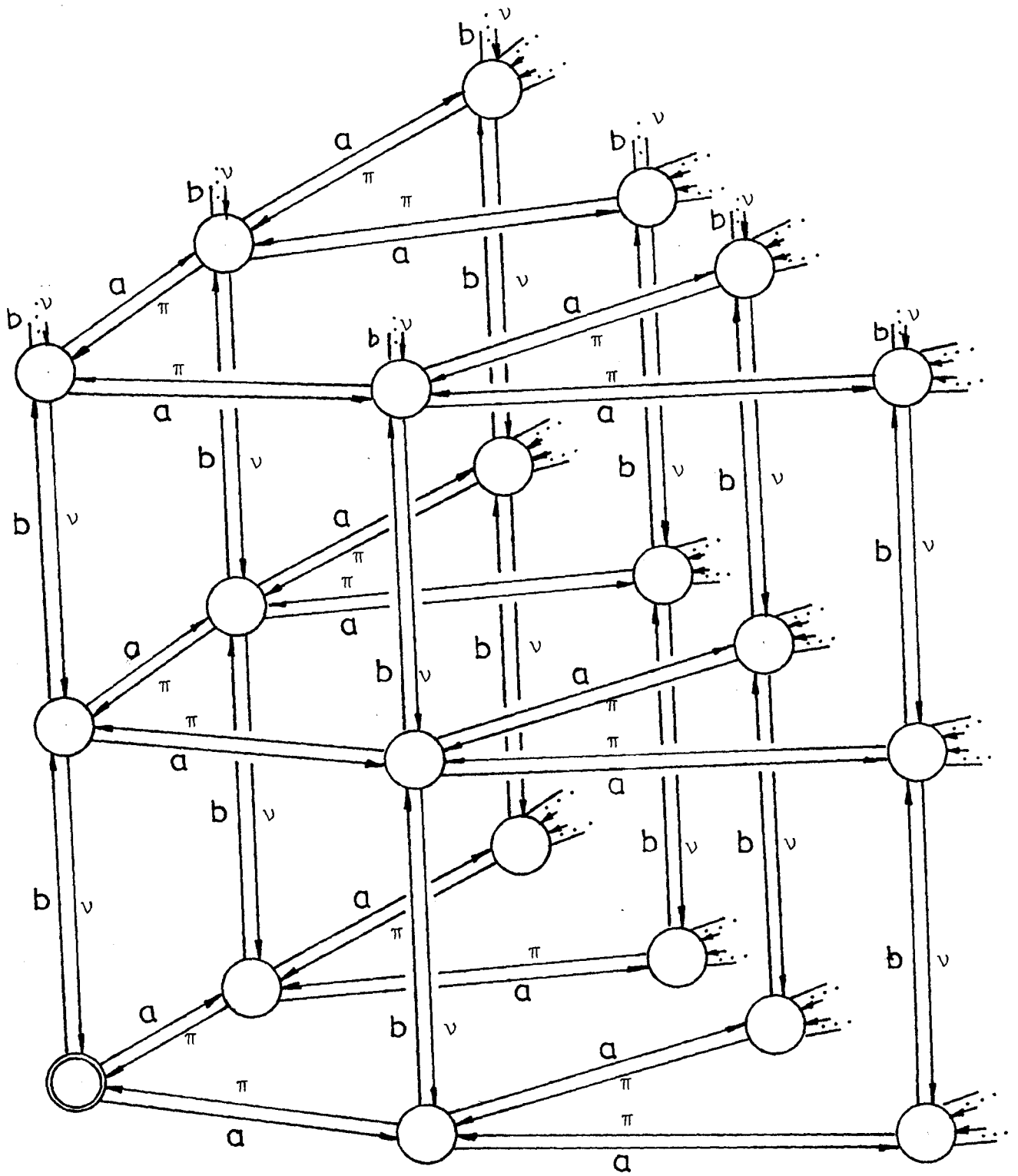


Figure 4.2. The ⟨T, A⟩-structure of dimensionality ⟪2, 2⟫

Example 4.3. We show in Fig.4.3 the TA-structure of dimensionality $\langle\langle 2, 3 \rangle\rangle$.

The uniformly self-isomorphic-embeddability of any $\langle T, A \rangle$-structure insures the next proposition.

Proposition 4.1. Any TA-structure is uniformly self-isomrphic -embeddable.

In the skeleton mapping $h = \langle \varepsilon, \kappa \rangle$ specified in Definition 2.8, ε is generally a many-to-one mapping, so for a given Γ' which is the skeleton structure of some Γ, the rdg which admits Γ' as its skeleton structure cannot be uniquely specified in general. We will prove however that a TA-structure Γ is uniquely constructed from A_d $(= h(\Gamma))$ for the given dimensionality of Γ, so that the TA-structure Γ induced from Γ is also determined uniquely.

In the following, let k be restricted to $k \geqslant 2$. In the d-dimensional array $A_d = (S, M)$ $(M = G_s \cup G_p)$, for $s = \langle s_1, s_2, \ldots, s_d \rangle \in S$, let $s' = (s_2, s_3, \ldots, s_d)$. Without loss of generality, we can let $b_1 \in G_s$ as a $(a\kappa^{-1}$ is the relation which is not a function as is previously mentioned). $\nu_1 \in G_p$ is especially denoted by τ.

Construction of a TA-structure $\Gamma = (C, F)$ from the d-dimensional array $A_d = (S, M)$

Let c_0 be a base cell of Γ and $(c_0(a\kappa^{-1})) = k$.

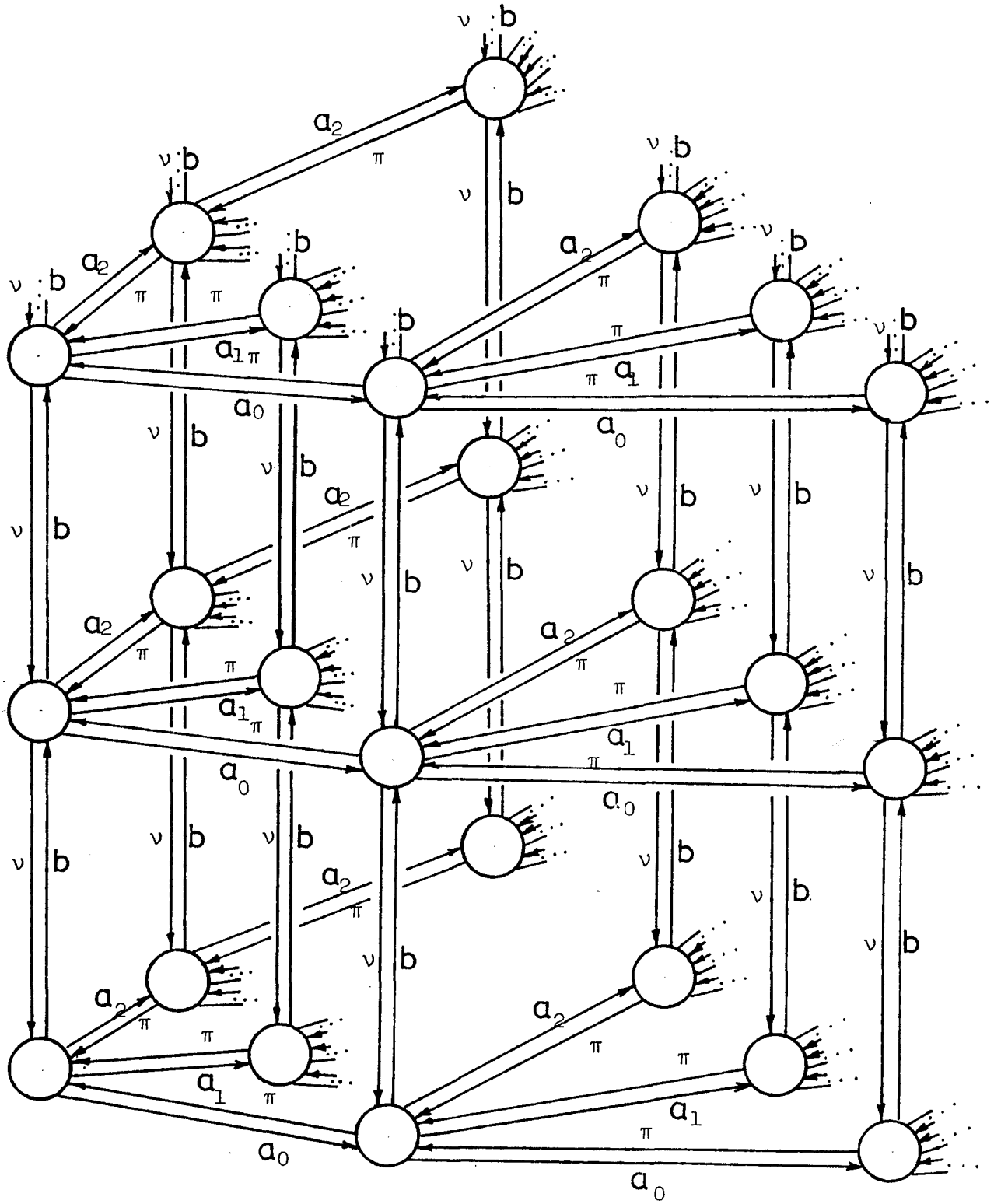


Figure 4.3. The TA-structure of dimensionality ⟪2, 3⟫

(1) Construction of C For each $s \in S$, let

$$s\varepsilon^{-1} = \{(\xi, s') \mid \xi \in a^{s_1 - 1}\} \tag{4.1}$$

$$C = S\varepsilon^{-1} = \{(\xi, s') \mid \xi \in a^*, s \in S\} \tag{4.2}$$

(2) Construction of a and $(G_s - \{a\})\kappa^{-1}$ For each $(\xi, s') \in C$ and each $a_i \in a$,

$$(\xi, s')a_i = (\xi a_i, s') \tag{4.3}$$

For each $(\xi, s') \in C$ and each $b_j \in G_s - \{a\}$,

$$(\xi, s')(b_j\kappa^{-1}) = (\xi, s'_+) \tag{4.4}$$

Here, $s'_+ = (s_2, \ldots, s_j+1, \ldots, s_d)$.

(3) Construction of $G_p\kappa^{-1}$ For each $(\xi, s') \in C - \{(e, s') \mid s \in S\}$ (e is the empty string),

$$(\xi, s')(\pi\kappa^{-1}) = (\xi', s') \tag{4.5}$$

Here, $\xi' = \xi_{m-1}$, when $|\xi| = m$ (ξ_{m-1} is the (m-1)-prefix of ξ). For each $\nu_j \in G_p - \{\pi\}$ and each $(\xi, s') \in C - \{(e, s') \mid s \in S\}$,

$$(\xi, s')(\nu_j\kappa^{-1}) = (\xi, s'_-) \tag{4.6}$$

In Equation (4.6), $s' = (s_2, \ldots, s_j - 1, \ldots, s_d)$

Theorem 4.2. TA-structure is uniquely determined for the specified dimensionality $\langle\langle d, k \rangle\rangle$ of Γ.

Proof. The uniqueness of C Let $\Gamma = (C, R)$ be a $\langle T, A \rangle$-structure of dimensionality $\langle\langle d, k \rangle\rangle$, then $h(\Gamma) = (S, M)$ is the d-dimensional array A_d. Let $S_0 = \{ s \in S \mid s_1 = 1 \}$ and $M_0 = \{ m_0 = m/S_0 \mid m \in M \}$, then the subgraph $A'_d = (S_0, M_0)$ of A_d is the $(d - 1)$-dimensional array such that $a/S_0 \notin M_0$. Therefore all the relations in $M_0\kappa^{-1} = \{ m\kappa^{-1}/S_0\varepsilon^{-1} \mid m \in M \}$ are restricted to functional relations, so that $h^{-1}(A'_d) = (S_0\varepsilon^{-1}, M_0\kappa^{-1})$ is isomorphic to A'_d. Hence, for each $s \in S_0$, $s\varepsilon^{-1}$ is a singleton set, so $s\varepsilon^{-1}$ can be denoted by s' ($s' = (s_2, \ldots, s_d)$ for $s = \langle 1, s_2, \ldots, s_d \rangle \in S$).

Now let $C_0 = S_0\varepsilon^{-1} = \{ s' \mid s \in S_0 \}$, $C^a_{s'} = \bigcup_{i \geq 0} (s'(a\kappa^{-1})^i)$ for each $s' \in C_0$, then from Theorem 3.8 the subgraph of Γ specified by $(C^a_{s'}, \{ a\kappa^{-1}/C^a_{s'} \})$ is isomorphic to Fig.3.6. First we show $C = \bigcup_{s' \in C_0} C^a_{s'}$. $C \supseteq \bigcup_{s' \in C_0} C^a_{s'}$ is obvious, and we prove $c \in \bigcup_{s' \in C_0} C^a_{s'}$ for each $c \in C$. Since $c\varepsilon \in S$ and $h(\Gamma)$ is an array, there exists $s \in S_0$ and $i \geq 0$ such that $c\varepsilon = sa^i$. $s = s'\varepsilon$ $(s' \in C_0)$ and (2.4) ensures $c\varepsilon = (s'\varepsilon)a^i = (s'(a\kappa^{-1})^i)\varepsilon$, then $c\gamma = (s'(a\kappa^{-1})^i)\gamma$ follows from the one-to-oneness

of u (mappings γ and u were defined in Definition 2.6). Invoking that singleton set $\{s'\}$ is a block of Γ, $(s'(a\kappa^{-1})^i)\gamma = s'(a\kappa^{-1})^i$ is obtained. Therefore $c \in s'(a\kappa^{-1})^i$ and $c \in \bigcup_{s' \in C_0} C^a_{s'}$ are concluded. Hence, $C = \bigcup_{s' \in C_0} C^a_{s'}$.

Now let $a\kappa^{-1}$ be functionized according to Definition 4.3 ($a = \{a_0, a_1, \ldots, a_{k-1}\}$). Since $(C^a_{s'}, \{a\kappa^{-1}/C^a_{s'}\})$ is a k-ary complete tree by Theorem 3.8, for arbitrary $c \in C^a_{s'}$ we can denote the path from s' to c by a string $\xi \in a^*$. Using this ξ and s', c can be denoted by (ξ, s'). Hence, $C = \{(\xi, s') \mid \xi \in a^*, s' \in C_0\}$. Then, $s' \in C_0 \subseteq C$ is newly denoted as (e, s'). Here we make sure that two distinct symbols $(\xi, s'_p), (\eta, s'_q) \in \bigcup_{s' \in C_0} C^a_{s'}$ never denote the same cell. For arbitrary (e, s'_p) and $(e, s'_q) \in C_0$ $(s'_p \neq s'_q)$, $C^a_{(e,s'_p)} \cap C^a_{(e,s'_q)} = \emptyset$ follows from Lemma 3.7-(1). Hence two cells $c, d \in C$ denoted by (ξ, s'_p) and (η, s'_q) $(s'_p \neq s'_q$ or $\xi \neq \eta)$ respectively are two distinct cells. Therefore cell set C is specified by (4.2).

<u>The uniqueness of a and $(G_s - \{a\})\kappa^{-1}$</u> That each $a_i \in a$ is uniquely determined and given by (4.3) is obvious from the uniqueness of $a\kappa^{-1}$ guaranteed by Theorem 3.8. We now prove the uniqueness of $b_j\kappa^{-1}$. For arbitrary $(\xi, s') \in C$, $(\xi, s')\varepsilon = \langle |\xi| + 1, s' \rangle$ from (4.1).

Here in general, when $s = \langle s_1, s_2, \ldots, s_d \rangle$, $\langle i, s' \rangle = \langle i, s_2, \ldots, s_d \rangle$. Since $h(\Gamma)$ is an array, $((\xi, s')\varepsilon)b_j = \langle |\xi| + 1, s'_+ \rangle$ from Definition 4.1. While Γ is uniformly self-isomorphic-embeddable by Proposition 4.1, Theorem 3.2 assures that base cell c_0 of Γ is a root of Γ. Therefore by Lemma 2.5 each $b_j\kappa^{-1} \in \nabla_R(c_0)$ is total, so that $b_j\kappa^{-1} \in \nabla_R((\xi, s'))$. Then $((\xi, s')\varepsilon)b_j = ((\xi, s')(b_j\kappa^{-1}))\varepsilon$ applying (2.3), hence $(\xi, s')(b_j\kappa^{-1}) \in \langle |\xi| + 1, s'_+ \rangle u^{-1} = \{(\eta, s'_+) \mid |\eta| = |\xi|\}$. Since Γ is uniformly self-isomorphic-embeddable, $(\xi, s')(b_j\kappa^{-1}) = (\xi, s'_+)$, so $b_j\kappa^{-1}$ is determined uniquely as (4.4).

The uniqueness of $G_p\kappa^{-1}$ Since Γ admits root c_0, for arbitrary $2 \leqslant j \leqslant d$, $(b_j\kappa^{-1})(\nu_j\kappa^{-1}) = 1_C$ ($\nu_j \in G_p$) is obtained from Lemma 2.4, so that $(\xi, s')(\nu_j\kappa^{-1}) = (\xi, s'_-)$ follows. Hence, $\nu_j\kappa^{-1}$ is determined uniquely as (4.6). Owing to $(a\kappa^{-1})(\pi\kappa^{-1}) = 1_C$, there exists, for each $c \in C$, $d \in c(a\kappa^{-1})$ such that $d(\pi\kappa^{-1}) = c$. In fact however, we can assert that for each $c' \in c(a\kappa^{-1})$, $c'\pi = c$ is obtained, invoking the uniformly self-embeddability of Γ and its strong connectedness. The detailed proof of this assertion is omitted. This assertion results in that for each $(\xi, s') \in C - \{(e, s') \mid s \in S\}$, $\pi\kappa^{-1}$ is determined as (4.5). Q.E.D.

Definition 4.4 Let $\Gamma = (C, F)$ be the TA-structure of dimensionality $\ll d, k \gg$. We define the TA-structure $\Gamma^{(m,\mathfrak{m})} = (C^{(m,\mathfrak{m})}, F^{(m,\mathfrak{m})})$ [4]

[4] $\mathfrak{m}$ implies $(n_2, \ldots, n_d)$

sliced out from Γ by $(a^m, b_2^{n_2}, \ldots, b_d^{n_d})$ as follows.[5]

$$C^{(m,n)} = \{c_0 \xi b_2^{j_2} \ldots b_d^{j_d} \mid \xi \in a^i,\ 0 \leqslant i \leqslant m,\ 0 \leqslant j_t \leqslant n_t\},$$

$$F^{(m,n)} = \{f/C^{(m,n)} \mid f \in F\}.$$

Each component of $(a^m, b_2^{n_2}, \ldots, b_d^{n_d})$ specifies cross-section.

$(m,n) = (m, n_2, \ldots, n_d)$ is called the dimension of $\Gamma^{(m,n)}$. $\ll d, k \gg$ is also the dimensionality of $\Gamma^{(m,n)}$.

Example 4.4 We show in Fig.4.4 the TA-structure sliced out from the TA-structure of dimensionality $\ll 2, 3 \gg$ (given in Fig.4.3) by (a^2, b_2^2).

Definition 4.5 Let $\Gamma^{(m,n)}$ be the TA-structure sliced out by $(a^m, b_2^{n_2}, \ldots, b_d^{n_d})$ from the TA-structure of dimensionality $\ll d, k \gg$. From $\Gamma^{(m,n)}$,

(1) The tree $\Gamma_t = (C_t, F_t)$ sliced out by $(\eta \cdot a^m, b_2^{n_2'}, \ldots, b_d^{n_d'})$ $(\eta \in a^*,\ |\eta| + m' \leqslant m,\ n_t' \leqslant n_t)$ is defined as follows.

$$C_t = \{c_0 \eta \xi b_2^{n_2'} \ldots b_d^{n_d'} \mid \xi \in a^i,\ 0 \leqslant i \leqslant m'\}$$

$$F_t = \{f/C_t \mid f \in F\}$$

In $(\eta \cdot a^{m'}, b_2^{n_2'}, \ldots, b_d^{n_d'})$, $(\eta, b_2^{n_2'}, \ldots, b_d^{n_d'})$ affords the root of Γ_t,

---

[5] For simplicity, we denote as $b_2^{n_2}$ in stead of $(b_2 \kappa^{-1})^{n_2}$

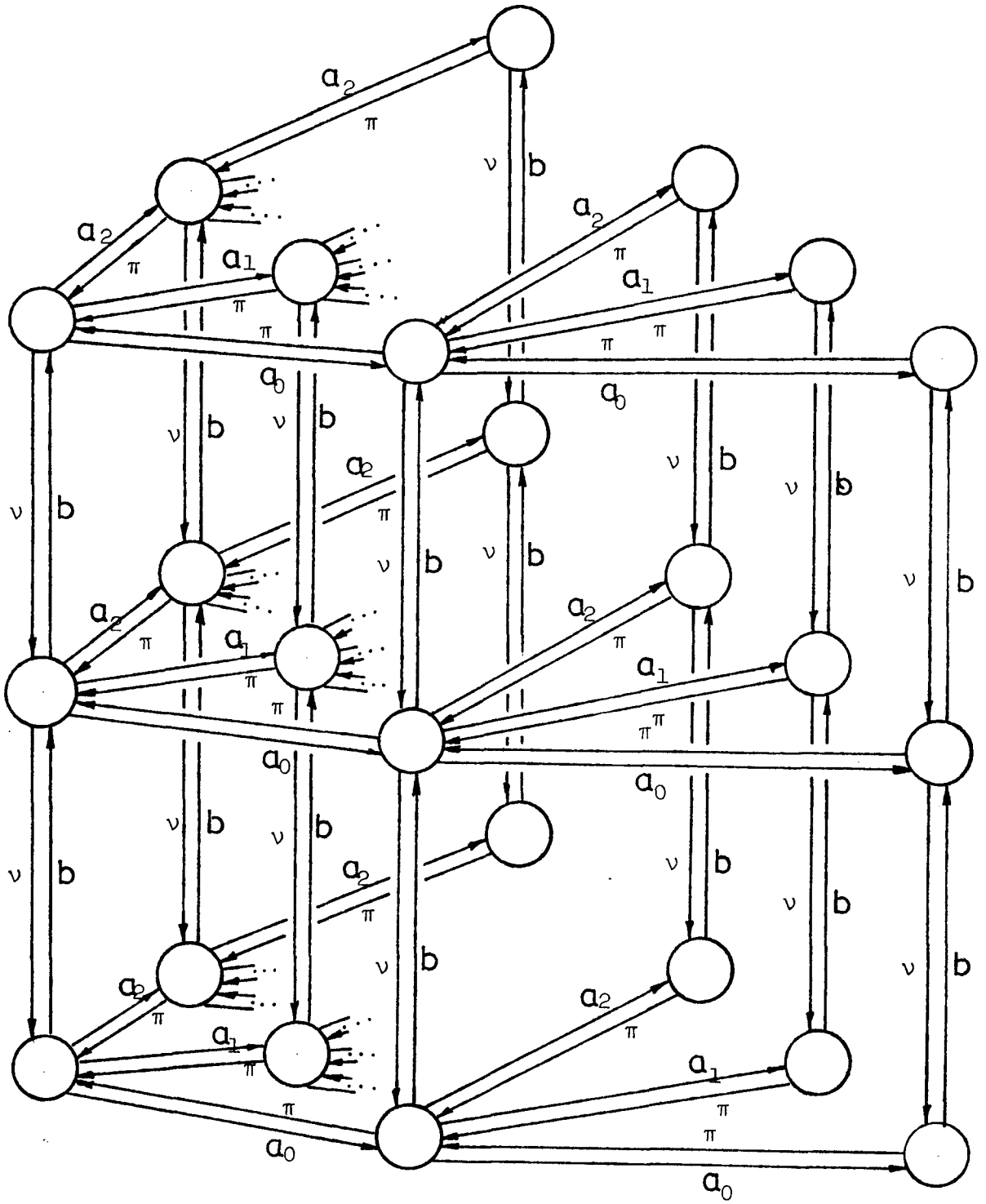


Figure 4.4. The TA-structure sliced out from Fig.4.3 by (a_2^2, b_2^2)

and $a^{m'}$ specifies the cross-section.

(2) The array $\Gamma_a = (C_a, F_a)$ sliced out by $(\eta\cdot\xi, b_2^{n_2'+p_2},\ldots, b_d^{n_d'+p_d})$ $(\eta, \xi \in a^*, |\eta|+|\xi| \leqslant m, n_t'+p_t \leqslant n_t)$ is defined as follows.

$$C_a = \{c_0\eta\xi_i b_2^{n_2'+j_2}\ldots\ldots b_d^{n_d'+j_d} \quad 0 \leqslant i \leqslant |\xi|, \ 0 \leqslant j_t \leqslant p_t\},$$

$$F_a = \{f/C_a \mid f \in F\}.$$

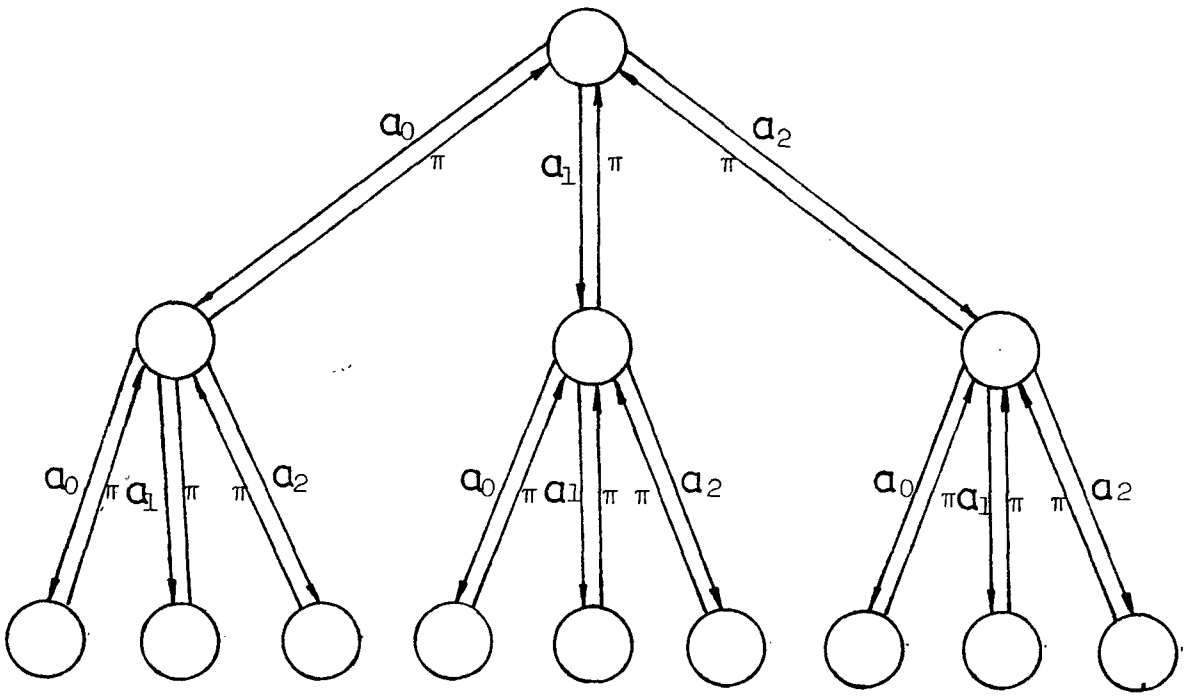
Example 4.5. From the TA-structure specified in Example 4.4,
(A) The tree Γ_t sliced out by $(1_C a_1 a_0, b_2)$
(B) The array Γ_a sliced out by $(1_C a_1 a_0, b_2^{0+2})$
are depicted in Fig.4.5.

4.3. Indexing of TA-structures

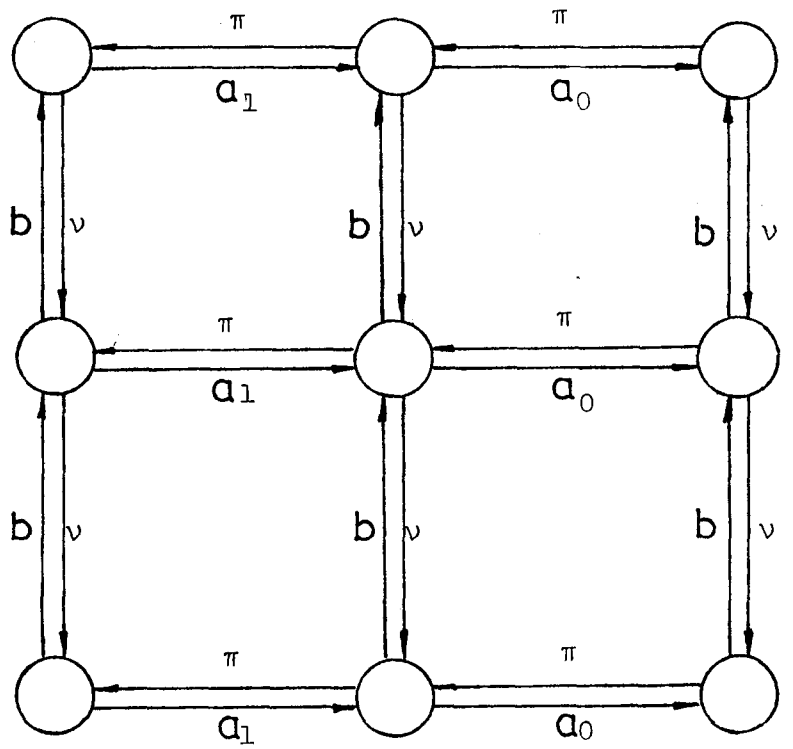
In fact, the index of a data cell is really an encoding of the very path from the base cell to the sought cell the realization scheme we are concentrating renders unnecessary to follow.

While, since a TA-structure $\Gamma = (C, F)$ is composed of both trees and arrays, structural aspects of trees and arrays coexist in Γ. So an index method of Γ can be naturally derived which reflects both string type (characteristic to trees) and integer tuple type (characteristic to arrays) indices.

This section is devoted to the indexing of a TA-structure Γ which assigns to each cell $c \in C$ its relative position from base cell



(A)



(B)

Figure 4.5. (A) The tree Γ_t sliced out from Fig.4.4 by $(1_C a_1 a_0, b_2)$

(B) The array Γ_a sliced out from Fig.4.4 by $(1_C a_1 a_0, b_2^{0+2})$

c_0. Moreover, for the several structures sliced out from TA-structure Γ or $\Gamma^{(m,n)}$, effective indexing methods are provided, which present programmers much convienience in processing such "local structures".

Let $\Gamma = (C, F)$ be an arbitrary TA-structure and c_0 be the base cell of Γ. Although each cell $c \in C$ can be indexed by an arbitrary sequence of atomic links $\zeta_c \in \nabla_R(c_0)$ such that $c_0\zeta_c = c$, it should be noted that the following structural property of Γ may permit us to specify the index of c uniquely.

Theorem 4.3. Let $\Gamma = (C, F)$ be an arbitrary TA-structure. Then for arbitrary b_i and b_j in $(G_s - \{a\})\kappa^{-1}$, $b_i b_j = b_j b_i$. For arbitrary $a_i \in a$ and arbitrary b_j in $(G_s - \{a\})\kappa^{-1}$, $a_i b_j = b_j a_i$.

In Fig.4.3 for example, $c_0a_0ba_2b^2a_1$ and $c_0a_0a_2b^2a_1b$ identify the same cell and both are reduced to $c_0a_0a_2a_1 \cdot b^3$ applying the above theorem. It is such decomposability of a path into the path in the tree involved and that of the array that makes it possible to denote the index by the followin fashion.

Definition 4.6. Let $\Gamma = (C, F)$ be the TA-structure of dimensionality $\ll d, k \gg$ and c_0 be the base cell of Γ. The <u>index set</u> I of Γ is,

$$I = \{\langle \xi, j_2, \ldots, j_d \rangle \mid \xi \in a^*, j_\ell \geq 1\}.$$

The indexing function $\mathcal{I} : C \longrightarrow I$ is given as follows.

For $c = c_0 \xi b_2^{j_2} 2 b_3^{j_3} 3 \ldots\ldots b_d^{j_d} d$ $(\xi \in a^*,\ j_\ell \geqslant 0)$,

$$\mathcal{I}(c) = \langle \xi,\ j_2+1, \ldots\ldots,\ j_d+1 \rangle \qquad (4.7)$$

The index of $c \in C$ given in (4.7) is an integer tuple type index whose first component is especially a string type index. Thus the conventional indices of a tree (string type) and an array (integer tuple type) are naturally extended and composed in the index. This reflects the structural property of a TA-structure $\Gamma = (C, F)$. That is, for the corresponding $\langle T, A \rangle$-structure $\mathbf{\Gamma} = (C, R)$, its skeleton structure $h(\mathbf{\Gamma})$ is an array A_d and one of whose successor links expands into trees in Γ.

Definition 4.7. Let $\Gamma^{(m,n)}$ be a TA-structure which is sliced out by $(a^m, b_2^n 2, \ldots\ldots, b_d^n d)$ from the TA-structure Γ of dimensionality $\ll d, k \gg$, and c_0 be the base cell of Γ. The index set $I^{(m,n)}$ of $\Gamma^{(m,n)}$ is,

$$I^{(m,n)} = \{ \langle \xi,\ j_2, \ldots\ldots,\ j_d \rangle \mid \xi \in a^m,\ 1 \leqslant j_\ell \leqslant n_d+1 \}$$

The indexing function $\mathcal{I} : C^{(m,n)} \longrightarrow I^{(m,n)}$ is the following one-to-one function.

For $c = c_0 \xi b_2^{j_2} 2 \ldots\ldots b_d^{j_d} d$ $(\xi \leqslant a^m,\ 0 \leqslant j_\ell \leqslant n)$

$\mathcal{I}(c) = \langle \xi,\ j_2+1, \ldots\ldots, j_d+1 \rangle$ (composite type of string and integer tuple)

Definition 4.8. Let $\Gamma^{(m,n)}$ be the TA-structure presented in Definition 4.7.

(1) For a tree $\Gamma_t = (C_t, F_t)$ sliced out from $\Gamma^{(m,n)}$ by $(\eta \cdot a^m, b_2^{n_2}, \ldots\ldots, b_d^{n_d})$, its index set is

$$I_t = \{\xi \mid \xi \in a^m\}.$$

The indexing function $\mathcal{I}_t : C_t \longrightarrow I_t$ is given as follows.

For $c = c_0 \eta \xi_j b_2^{j_2} \ldots\ldots b_d^{j_d}$ $(\eta, \xi \in a^m,\ |\eta| + |\xi| \leqslant m,\ 0 \leqslant j_\ell \leqslant n_\ell)$,

$$\mathcal{I}_t(c) = \xi \qquad \text{(string type)} \tag{4.9}$$

(2) For an array $\Gamma_a = (C_a, F_a)$ sliced out from $\Gamma^{(m,n)}$ by $(\eta \cdot \xi, b_2^{n_2'+p_2}, \ldots\ldots, n_d^{n_d'+p_d})$, its index set is

$$I_a = \{\langle i_1, i_2, \ldots\ldots, i_d\rangle \mid 1 \leqslant i_1 \leqslant |\xi| + 1,\ 1 \leqslant i_\ell \leqslant p_\ell + 1\ (2 \leqslant \ell \leqslant d)\}.$$

The indexing function $\mathcal{I}_a : C_a \longrightarrow I_a$ is given as follows.

For $c = c_0 \eta \xi_{i_1} b_2^{n_2'+i_2} \ldots\ldots b_d^{n_d'+i_d}$ $(\eta, \xi \in a^m,\ |\eta| + |\xi| \leqslant m,$

$0 \leqslant n_\ell' + i_\ell \leqslant n_\ell)$,

$$\mathcal{I}_a(c) = \langle i_1+1, \ldots\ldots, i_d+1\rangle \qquad \text{(integer tuple type)} \tag{4.10}$$

It is noteworthy that after slicing out a tree Γ_t or an array Γ_a from $\Gamma^{(m,n)}$, its indexing can be accomplished by string type (4.8) or integer tuple type (4.10) index respectively, being independent of the composite type index (4.8). Consequently, when a programmer refers to

a cell in Γ_t or Γ_a, he may concentrate his attention only on its position in Γ_t or Γ_a, without any care of its position in $\Gamma^{(m,n)}$.

4.4 Addressing Function of TA-structures

This section is devoted to the construction of addressing functions which map index sets of TA-structures and each of their local structures (i.e., trees or arrays sliced out) to an address set.

For an addressing function established, some of the criteria for assessing the quality of it must be considered. At least, the next three criteria should be markedly taken into account.

(a) Complexity of accessing to a data cell : the computational complexity of the addressing function $\mathcal{A}$.

(b) Efficiency of storage utilization : the extent to which $\mathcal{A}$ stores the set of data cells C in contiguous memory block, a measure of the size of "gaps".

(c) Complexity of traversal : For $f_i \in F$ and $\langle \xi, n \rangle \in I$, the difficulty of computing the sequence

$$(f_1(\xi, n)),\ (f_1 f_2(\xi, n)),\ (f_1 f_2 f_3(\xi, n)),\ \ldots\ldots\ldots .$$

First we construct two kinds of $\mathcal{A}$'s for a "finite" TA-structure $\Gamma^{(m,n)}$, taking account of criterion (a). One of them facilitates the traversals in each tree Γ_t in $\Gamma^{(m,n)}$. The other facilitates the traversals in each array Γ_a in $\Gamma^{(m,n)}$ except along the "tree direction".

But these addressing functions can be obtained at the cost of

criterion (b) : many "gaps" deteriorate the storage utilization.

Second we construct another two kinds of $\mathcal{A}$'s inserting modifying terms which compensate the gaps to the first two functions. Last we give $\mathcal{A}$'s for the local structures sliced out from $\Gamma^{(m,m)}$.

In the beginning, we give a general definition of an addressing function for an arbitrary data structure Δ which admits index set I_Δ.

Definition 4.9. Let Δ be an arbitrary data structure which admits index set I_Δ. An addressing function of Δ is the next one-to-one total function.

$$\mathcal{A} : I_\Delta \longrightarrow N,$$

where N is a set of natural numbers, each of which denotes an address of a computer memory (a random access memory).

We now discuss addressing functions of TA-structure $\Gamma^{(m,m)}$. For the sake of simplicity, the dimensionality $\langle\langle d, k \rangle\rangle$ of $\Gamma^{(m,m)}$ is restricted to $d = 2$. We shall have little trouble extending our results to the case of $d \geqslant 3$.

Many computational procedures would call for repeated traversal along the specific directions (e.g. "tree-direction" or "array-directions") of TA-structures. Since such traversal-oriented procedures are so common, we now construct for $\Gamma^{(m,n-1)}$ two kinds of addressing functions by which repeated traversal along the specific direction is facilitated.

Let $\Gamma^{(m,n)}$ be a TA-structure of dimensionality $\langle\!\langle 2, k \rangle\!\rangle$, and $\langle \xi, j \rangle$ be an index of an arbitrary cell in $\Gamma^{(m,n-1)}$.

$$\mathcal{A}_t(\langle \xi, j \rangle) = \overline{a_1 \xi} + \wp_k^m (j - 1) \tag{4.11}$$

$$\mathcal{A}_a(\langle \xi, j \rangle) = j + n(\overline{a_1 \xi} - 1) \tag{4.12}$$

Here, $\overline{a_{i_1} a_{i_2} \cdots a_{i_\ell}}$ $(a_{i_j} \in a)$ is the decimal representation of k-ary number $i_1 i_2 \cdots i_\ell$, and $\wp_k^m = \overline{a_1 a_{k-1}^m} = \overline{a_1 a_{k-1} \cdots a_{k-1}} = 2k^m - 1$. For instance, for $\Gamma^{(2,2)}$ of dimensionality $\langle\!\langle 2, 3 \rangle\!\rangle$,

$$\mathcal{A}_t(\langle a_0 a_2, 3 \rangle) = \overline{a_1 a_0 a_2} + \wp_3^2 (3 - 1) = 11 + 17 \times 2 = 45,$$

$$\mathcal{A}_a(\langle a_0 a_2, 3 \rangle) = 3 + 3(\overline{a_1 a_0 a_2} - 1) = 3 + 3 \times (11 - 1) = 33.$$

It should be noted that the address of the base cell c_0 is $\mathcal{A}_t(\langle e, 1 \rangle) = \mathcal{A}_a(\langle e, 1 \rangle) = 1$ regardless of the dimensionality and the dimension of $\Gamma^{(m,n-1)}$. Although $\Gamma^{(m,n-1)}$ is relocatable in the sense that one can allocate $\Gamma^{(m,n-1)}$ anywhere by fixing c_0 at an arbitrary address. This relocatability is easily guaranteed by adding $\mathcal{A}_t$ or $\mathcal{A}_a$ the displacement $\underline{\text{address}}(c_0) - 1$.

Example 4.6. Address of each cell in Fig.4.4 allocated by $\mathcal{A}_t$ or $\mathcal{A}_a$ is shown inside or outside the cell in Fig.4.6 respectively.

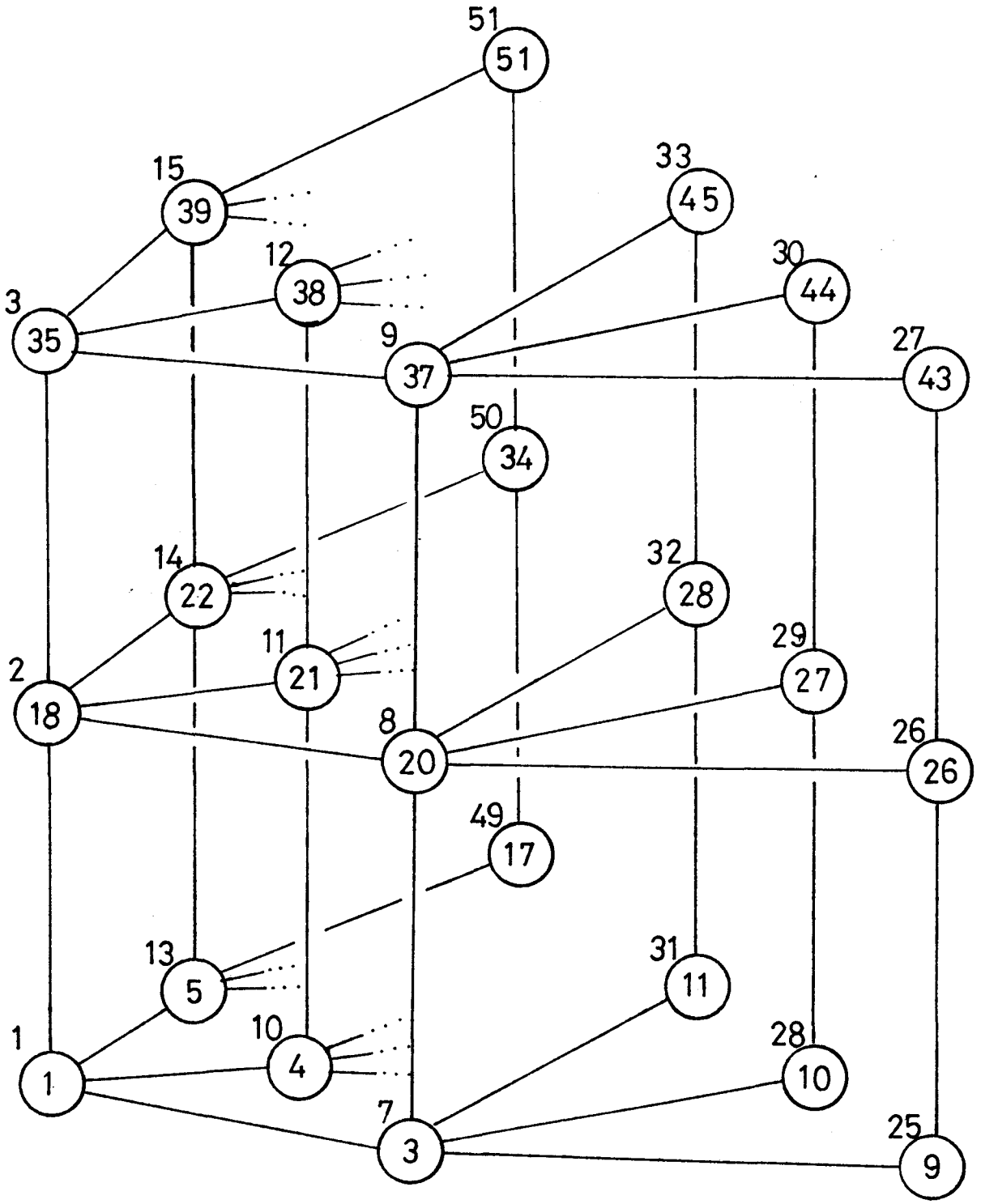


Figure 4.6. Allocation of Fig.4.4 by $\mathcal{A}_t$ (inside the cells) and $\mathcal{A}_a$ (outside the cells)

If $\mathcal{A}_t$ is chosen, a memory block is assigned to each tree T_i in $\Gamma^{(m,n-1)}$, and to each cell set on the same level in T_i, a consecutive storage area is afforded. If $\mathcal{A}_a$ is chosen, a consecutive storage area is afforded to each "b-axis".

It is noteworthy that $\mathcal{A}_t$ and $\mathcal{A}_a$ are based on storage mapping functions of an array of dimension (m,n) ; i.e.,

$$A(\langle i, j \rangle) = i + m(j - 1) \qquad \text{(by row-wise order)},$$

$$A(\langle i, j \rangle) = j + n(i - 1) \qquad \text{(by column-wise order)},$$

respectively. Only simple decimalization of $a_1\xi$ is additionaly required to compute the address of $\langle \xi, j \rangle$. It should be noted however that the storage utilization is not so good in the allocation by $\mathcal{A}_t$ or $\mathcal{A}_a$ because of a "gap" existing between i-level and (i+1)-level in each tree. This gap becomes wider when the level increases.

Now, we estimate the memory availability briefly when $\Gamma^{(m,n-1)} = (C^{(m,n-1)}, F^{(m,n-1)})$ is stored by $\mathcal{A}_t$ or $\mathcal{A}_a$. Here it is assumed that one storage location is occupied by a single cell. In the following, let $\mathcal{A}$ be $\mathcal{A}_t$ or $\mathcal{A}_a$. All the results that will be obtained are same on $\mathcal{A}_t$ and $\mathcal{A}_a$.

Let σ_k^m be the number of cells in each tree in $\Gamma^{(m,n-1)}$, hence $\sigma_k^m = \sum_{i=0}^{m} k^i$. Then, $C^{(m,n-1)} = n\sigma_k^m$. While, the size of storage area required to store $C^{(m,n-1)}$ is $\mathcal{A}(\langle (k-1)^m, n \rangle) = n\rho_k^m$ ($\rho_k^m = 2k^m - 1$).

Hence memory availability α_k^m is given as follows.

$$\alpha_k^m = \frac{n\sigma_k^m}{n\rho_k^m} = \frac{k^{m+1}-1}{(k-1)(2k^m-1)}.$$

For example, $\alpha_3^2 = 13/17$, $\alpha_2^m = 1$ for all m, and $\lim_{m\to\infty} \alpha_k^m = \lim_{k\to\infty} \alpha_k^m = 1/2$. In addition,

$$\frac{\alpha_k^{m+1}}{\alpha_k^m} \approx \frac{2k^{m+2}-k^2-2}{2k^{m+2}-3k} \ (\leqslant 1), \qquad \frac{\alpha_{k+1}^m}{\alpha_k^m} \approx \frac{k^2-1}{k^2} \ (<1) \quad \text{for } k \geqslant 3.$$

Now, let δ_i^k be the gap between i- and $(i+1)$-level in each tree, i.e., $\delta_i^k = \mathcal{A}(\langle a_0^{i+1}, j\rangle) - \mathcal{A}(\langle a_{k-1}^i, j\rangle)$. Then the gap ratio $\beta_i^k = \delta_{i+1}^k / \delta_i^k$ is

$$\beta_i^k = \frac{k^{i+2}-2k^{i+1}+1}{k^{i+1}-2k^i+1}.$$

In particular, $\delta_i^2 = 1$ for all i, which implies that there exists no gap when $k = 2$ (binary tree). When $k \geqslant 3$, $\beta_i^k \approx k$ is obtained.

In the allocation of $\Gamma^{(m,n-1)}$ by $\mathcal{A}_t$ or $\mathcal{A}_a$, it can be seen that α_k^m deteriorates monotonously with the lower bound $1/2$ as k or m increases. If one desires efficiency of storage utilization, by inserting only a simple term which cancels every gap in each tree, one can obtain new addressing functions of 100% storage utilization for arbitrary k and m.

$$\mathcal{A}_t'(\langle \xi, j \rangle) = (\overline{a_1 \xi} - \overline{a_{k-2}^{|\xi|}}) + \sigma_k^m(j-1)$$

$$\mathcal{A}_a'(\langle \xi, j \rangle) = j + n(\overline{a_1 \xi} - \overline{a_{k-2}^{|\xi|}} - 1)$$

Especially when $k = 2$, the inserted term $\overline{a_{k-2}^{|\xi|}}$ becomes 0, so that $\mathcal{A}_t'$ and $\mathcal{A}_a'$ coincide with $\mathcal{A}_t$, $\mathcal{A}_a$ respectively. This implies the fact that a binary tree can have efficient addressing functions in both memory availability and accessing time, i.e., computational simplicity.

Example 4.7. We show allocations of TA-structure in Fig.4.4 by $\mathcal{A}_t'$ and $\mathcal{A}_a'$ inside and outside cells in Fig.4.7.

Let Γ_t be a tree sliced out from $\Gamma^{(m,n-1)}$ by $(\eta \cdot a^{m'}, b^{n'-1})$ and Γ_a be an array sliced out from $\Gamma^{(m,\,n-1)}$ by $(\eta \cdot \xi, b^{n'+p-1})$. Lastly, we give addressing functions for both Γ_t and Γ_a. Four kinds of functions can be constructed according to $\mathcal{A}_t$, $\mathcal{A}_a$, $\mathcal{A}_t'$, $\mathcal{A}_a'$.

$$\mathcal{A}_T(\xi) = \mathcal{A}(\langle \eta\xi, n' \rangle) \qquad (|\xi| \leqslant m'),$$

$$\mathcal{A}_A(\langle i, j \rangle) = \mathcal{A}(\langle \eta\xi_{i-1}, n'+j \rangle) \qquad (i \leqslant |\xi|+1).$$

Here, $\mathcal{A} \in \{\mathcal{A}_t, \mathcal{A}_a, \mathcal{A}_t', \mathcal{A}_a'\}$.

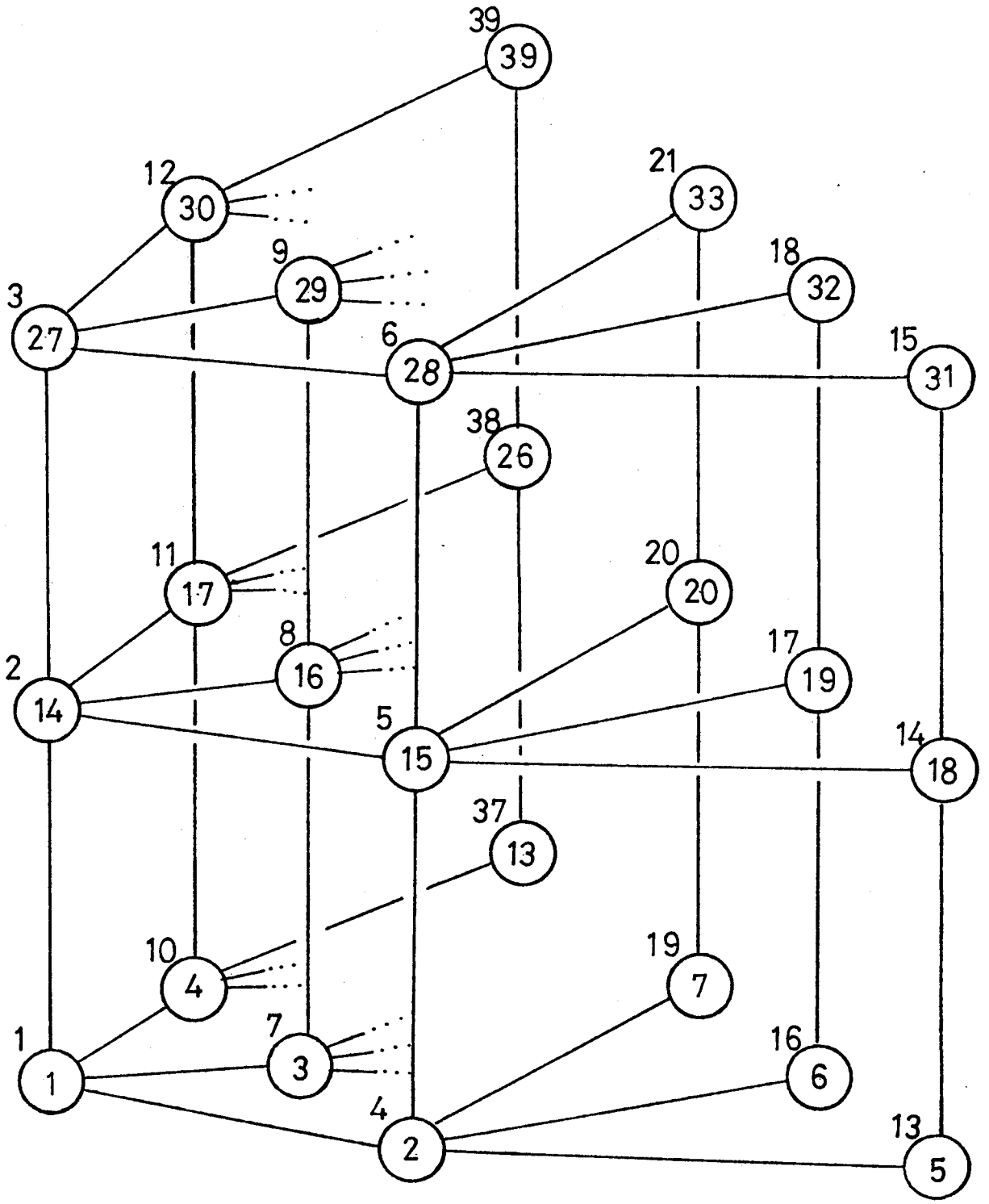


Figure 4.7 Allocation of Fig.4.4 by $\mathcal{A}_t'$ (inside the cells) and by $\mathcal{A}_a'$ (outside the cells)

4.5 Conclusion

With the theoretical basis developed in the previous chapters, we have newly defined a directly accessible data structure called a TA-structure, which is a composite structure of trees and arrays. We have described indexing methods of TA-structures and constructed their addressing functions from some criteria.

We have not discussed an allocation of an infinite TA-structure $\Gamma = (C, F)$, which is "extendible" in the sense of[23].

In the following, we give an example of an addressing function for the TA-structure of dimensionality $\ll 2, 2 \gg$.

For $\langle \xi, j \rangle \in I$,

$$\mathcal{A}(\langle \xi, j \rangle) = 2^{j-1}(2\overline{a_1 \xi} - 1)$$

It will be a difficult but interesting work to analysis such an addressing function of an infinite TA-structure under the three criteria listed in the introduction of this chapter.

CHAPTER 5

LABELED TA-STRUCTURES

5.1 Introduction

Of course, it is possible that each of the link labels in a TA-structure $\Gamma = (C, F)$ carries some meaning of the relation between the items related by the link. The link labels are in fact the names of the functions in F. Thus the variety of "relations" which one can impose on the links of Γ is fairly restricted: in fact, restricted to $\#F$ kinds of "relations", and this $\#F$ is usually small owing to the high uniformity of Γ. Moreover, each kind of successor links of both trees and arrays in Γ should emanate from every cell. Such restrictions arising from the strong uniformity of Γ may inevitably narrow the scope of the data structures that are representable naturally and efficiently by our TA-structures. Indeed we can say that the set of functions (links) F of a TA-structure is only able to bear some rough meanings. That is to say, F is not fully refined (or partitioned) to hold the meanings of various relations among the data items.

It is one of the urgent tasks that we establish a new scheme which would overcome such defect. Such a scheme is especially needed when we employ tree structures by which hierarchical structures (which is often introduced on a data set) can be described directly and naturally.

In this chapter, we provide the strategy of partitioning each function in $a \subset F$ into the set of sub-functions. For a TA-structure thus "divided" we explain its labeling scheme. Then we construct an addressing function for this labeled TA-structure.

5.2 Labeling Schemes of TA- structures

In this section, we provide the strategy of partitioning each function in $a \subset F$ into the set of sub-functions. By the application of this strategy to $\Gamma^{(m,n)}$, we are able to obtain the "refinement of $\Gamma^{(m,n)}$", namely $Re(\Gamma^{(m,n)})$. What is more, we provide a labeling scheme $(\mathcal{L}, \Sigma)$ which fixes an arbitrary label that represents the meaning of the connection between two items of a tree in $Re(\Gamma^{(m,n)})$.

In the following, let $\Gamma = (C, F)$ be a TA-structure of dimensionality $\langle\langle d, k \rangle\rangle$, and let $\Gamma^{(m,n)} = (C^{(m,n)}, F^{(m,n)})$ be a TA-structure sliced out from Γ.

Definition 5.1. Let $a_i \in a \subset F$ $(0 \leqslant i \leqslant k-1)$. The refinement of a_i on (m, n) is defined as

$$Re(a_i;(m,n)) = \{a_{il} \mid 1 \leqslant l \leqslant m\}.$$

a_{il} is provided according to the following rule.

$$a_{il} = \{\langle\langle \xi, j \rangle, \langle \xi a_i, j \rangle\rangle \mid |\xi| = l-1,\ j \leqslant n\} \qquad (5.1)$$

In (5.1), $\langle \xi, j \rangle = \langle \xi, j_2, j_3, \ldots, j_d \rangle$ and $j \leqslant n$ implies that $j_p \leqslant n_p$ for each $2 \leqslant p \leqslant d$.

The refinement of a on (m, n), $Re(a; (m, n))$ is defined as follows.

$$Re(a; (m, n)) = \bigcup_{a_i \in a} Re(a_i;(m, n))$$

Then, the refinement of $F^{(m,n)}$, $Re(F^{(m,n)})$ is defined as

$$Re(F^{(m,n)}) = (F^{(m,n)} - a/C^{(m,n)}) \cup Re(a;(m, n))$$

Last, the refinement of $\Gamma^{(m,n)}$, $Re(\Gamma^{(m,n)})$ is defined as follows.

$$Re(\Gamma^{(m,n)}) = (C^{(m,n)}, Re(F^{(m,n)})).$$

Example 5.1. For $\Gamma^{(2,2)}$ in Fig.4.4, $Re(\Gamma^{(2,2)})$ is given in Fig.5.1.

Definition 5.2. For $a_{i\ell}$ $(0 \leqslant i \leqslant k-1,\ 1 \leqslant \ell \leqslant m)$ specified in (5.1), let ℓ be fixed, and α_ℓ be defined as follows.

$$\alpha_\ell = \{a_{i\ell} \mid 0 \leqslant i \leqslant k-1\}.$$

Definition 5.3. Refine $\Gamma^{(m,n)}$ according to Definition 5.1 and obtain $Re(\Gamma^{(m,n)})$. Let $L_1, L_2, \ldots, L_m$ be an arbitrary set of labels such that $\# L_\ell = k$, and $\psi_1, \psi_2, \ldots, \psi_m$ be an arbitrary one-to-one function such that $\psi_\ell : \alpha_\ell \longrightarrow L_\ell$ $(1 \leqslant \ell \leqslant m)$. Here α_ℓ comes from Definition 5.2. $\mathcal{L} = \bigcup_{1 \leqslant \ell \leqslant m} L_\ell$ is called a label set of $a/C^{(m,n)}$, and $\Sigma = \bigcup_{1 \leqslant \ell \leqslant m} \psi_\ell$ is called a labeling function of $a/C^{(m,n)}$. Note that Σ is a function which maps $Re(a; (m, n))$ to $\mathcal{L}$ and generally this Σ is not one-to-one. $(\mathcal{L}, \Sigma)$ is called a labeling scheme of $\Gamma^{(m,n)}$. Note that each label $q \in \mathcal{L}$ specifies a (partial) function on $C^{(m,n)}$.

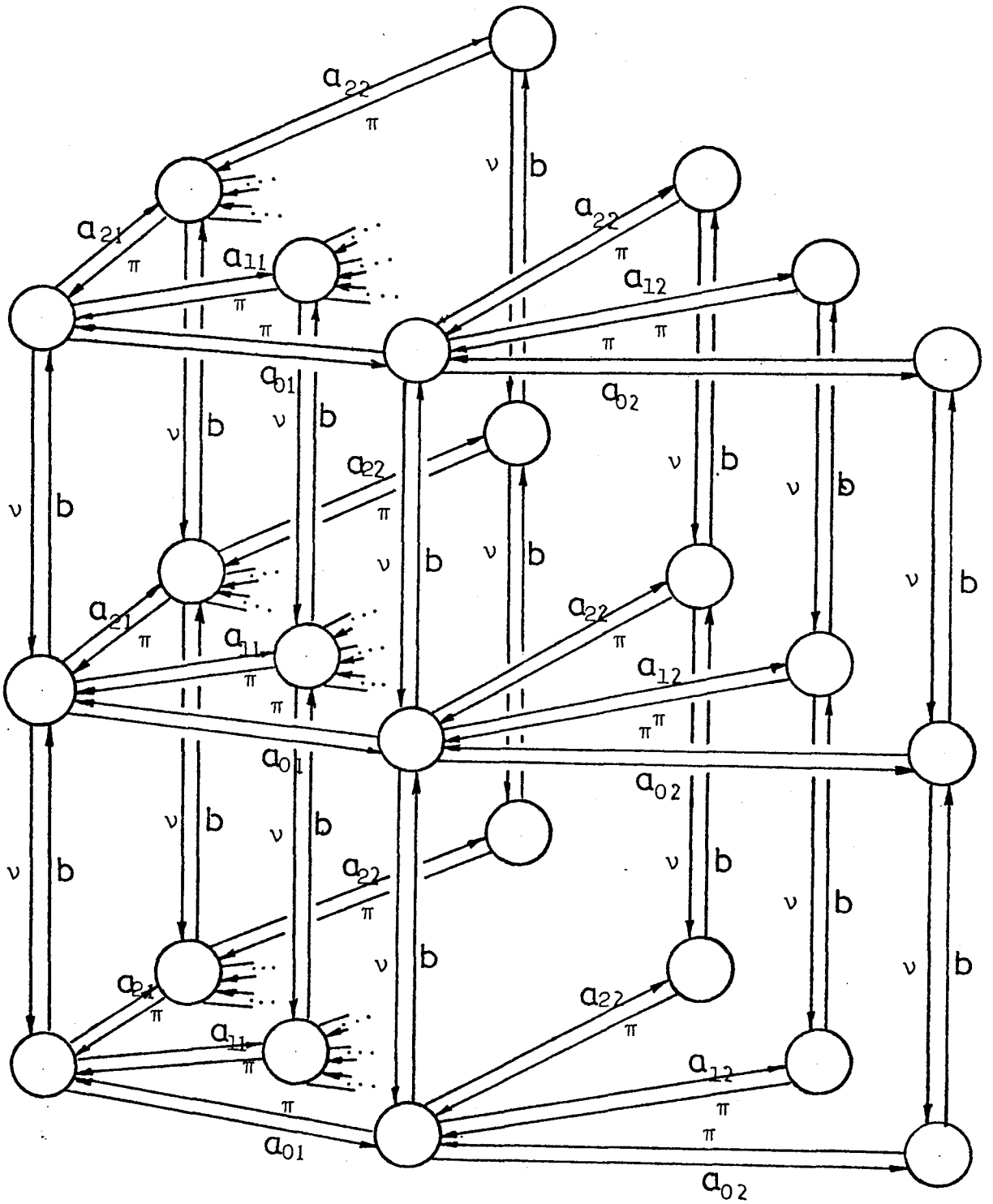


Figure 5.1. The refinement of the TA-structure in Fig.4.4

Definition 5.4. Refine $\Gamma^{(m,n)}$ and obtain $Re(\Gamma^{(m,n)})$. For the labeling scheme $(\mathcal{L}, \Sigma)$ of $\Gamma^{(m,n)}$ specified, let $F^{(m,n)} = (F^{(m,n)} - \{a/C^{(m,n)}\}) \cup \mathcal{L}$. Then,

$$\underbrace{Re(\Gamma^{(m,n)})} = (C^{(m,n)}, \underbrace{F^{(m,n)}})$$

is said as the labeled $\Gamma^{(m,n)}$ by $(\mathcal{L}, \Sigma)$.

Example 5.2. For $\Gamma^{(2,2)}$ of dimensionality $\langle\langle 2, 2 \rangle\rangle$ and the following labeling scheme of $\Gamma^{(2,2)}$, the labeled $\Gamma^{(2,2)}$ by $(\mathcal{L}, \Sigma)$, namely $\underbrace{Re(\Gamma^{(2,2)})}$, is described in Fig.5.2. $\mathcal{L} = \bigcup_{\ell=1,2} L_\ell$, $\Sigma = \bigcup_{\ell=1,2} \psi_\ell$ are specified as follows.

$$L_1 = \{\text{FANAME, MANAME}\}, \quad L_2 = \{\text{BIRDAY, AGE}\}$$

$$\psi_1 = \{\langle a_{01}, \text{FANAME}\rangle, \langle a_{11}, \text{MANAME}\rangle\}, \quad \psi_2 = \{\langle a_{02}, \text{BIRDAY}\rangle, \langle a_{12}, \text{AGE}\rangle\}$$

Definition 5.5. Let $(\mathcal{L}, \Sigma)$ be a labeling scheme of $\Gamma^{(m,n)}$. We say the following table representation of $(\mathcal{L}, \Sigma)$ as a labeling table of $\Gamma^{(m,n)}$.

	a_0	a_1	. . . . .	a_{k-1}
1	$q_{0,1}$	$q_{1,1}$		$q_{k-1,1}$
2	$q_{0,2}$	$q_{1,2}$		$q_{k-1,2}$
.				
.				
.				
m	$q_{0,m}$	$q_{1,m}$		$q_{k-1,m}$

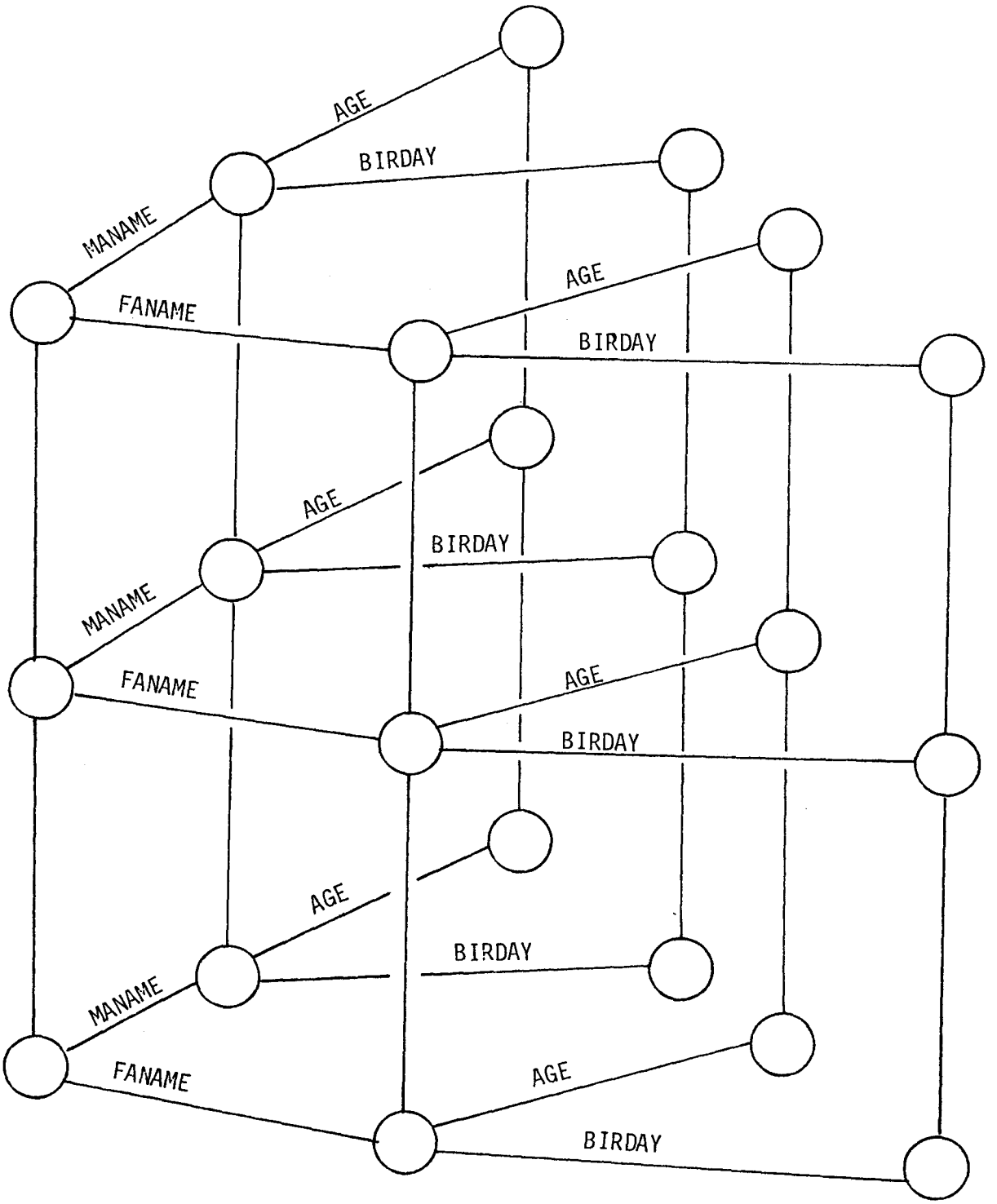


Figure 5.2. A labeled TA-structure

Here, $L_\ell = \{q_{0,\ell}, q_{1,\ell}, \cdots, q_{k-1,\ell}\}$ $(1 \leqslant \ell \leqslant m)$.

Example 5.3. Let $(\mathcal{L}, \Sigma)$ be as in example 5.2, then the labeling table of $\Gamma^{(2,2)}$ is given as follows.

	a_0	a_1
1	FANAME	MANAME
2	BIRDAY	AGE

5.3. Addressing Schemes of Labeled TA-structures

In this section, first for a TA-structure $\Gamma^{(m,n)}$ labeled by some labeling scheme $(\mathcal{L}, \Sigma)$ in the previously described manner, we explain its indexing method by introducing an index labeling function φ. Second, for the labeled index set $I^{(m,n)}$ of a labeled $\Gamma^{(m,n)}$, we explain an addressing mechanism which makes use of a specified labeling table of $\Gamma^{(m,n)}$.

Definition 5.6. Let $(\mathcal{L}, \Sigma)$ be a labeling scheme of $\Gamma^{(m,n)}$. Obtain the labeled $\Gamma^{(m,n)}$ by $(\mathcal{L}, \Sigma)$, namely $\underset{\sim}{\mathrm{Re}(\Gamma^{(m,n)})}$. Let $\underset{\sim}{I^{(m,n)}}$ be defined as ,

$$\underset{\sim}{I^{(m,n)}} = \{\langle \zeta , \jmath \rangle \mid \zeta = e, \zeta \in L_1 \times L_2 \times \cdots \times L_t, 1 \leqslant t \leqslant m, \jmath \leqslant n\}.$$

For the index set $I^{(m,n)}$ of $\Gamma^{(m,n)}$, we define an index labeling function $\varphi: I^{(m,n)} \longrightarrow \underset{\sim}{I^{(m,n)}}$ as follows.

For each $\langle a_{i_1} a_{i_2} \ldots a_{i_t}, \vec{j} \rangle \in I^{(m,n)}$ $(1 \leqslant t \leqslant m,\ a_{i_j} \in \mathfrak{a})$,

$$\varphi(\langle \xi, \vec{j} \rangle) = \langle (a_{i_1 1}\psi_1)(a_{i_2 2}\psi_2) \ldots (a_{i_t t}\psi_t), \vec{j} \rangle \quad (a_{i_j j} \in \alpha_j)$$

Let $\varphi(\langle \xi, \vec{j} \rangle)$ be denoted as $\langle \underset{\sim}{\xi}, \vec{j} \rangle$. Clearly, the labeled index set $\underset{\sim}{I}^{(m,n)}$ serves itself as an index set of $\underline{\mathrm{Re}(\Gamma^{(m,n)})}$.

Proposition 5.1. For a given $\Gamma^{(m,n)}$ and its labeling scheme $(\mathcal{L}, \Sigma)$, the index labeling function φ is one-to-one.

Proof. The one-to-oneness of each function ψ_ℓ $(1 \leqslant \ell \leqslant m)$ immediately assures that φ is a one-to-one function.

Now we explain an addressing scheme of a labeled $\Gamma^{(m,n)}$. Let $\Gamma^{(m,n)} = (C^{(m,n)}, F^{(m,n)})$ be a TA-structure of dimensionality $\langle\!\langle d, k \rangle\!\rangle$. Let $(\mathcal{L}, \Sigma)$ be a labeling scheme of $\mathrm{Re}(\Gamma^{(m,n)})$. For the labeled index set $\underset{\sim}{I}^{(m,n)}$ and an addressing function $\mathcal{A}$ of $\Gamma^{(m,n)}$, we define $\underset{\sim}{\mathcal{A}}: \underset{\sim}{I}^{(m,n)} \longrightarrow N$ as follows. For $\langle \zeta, \vec{j} \rangle \in I^{(m,n)}$,

$$\underset{\sim}{\mathcal{A}}(\langle \zeta, \vec{j} \rangle) = \mathcal{A}(\varphi^{-1}(\langle \zeta, \vec{j} \rangle)). \tag{5.2}$$

Note that since φ is one-to-one, $\varphi^{-1}(\langle \zeta, \vec{j} \rangle)$ is uniquely determined for $\langle \zeta, \vec{j} \rangle \in I^{(m,n)}$. That $\underset{\sim}{\mathcal{A}}$ is an addressing function of $\underline{\mathrm{Re}(\Gamma^{(m,n)})}$ is ensured by the following proposition.

Proposition 5.2. The function $\underset{\sim}{\mathcal{A}}: \underset{\sim}{I}^{(m,n)} \longrightarrow N$ defined by (5.2) is an addressing function of $\underline{\mathrm{Re}(\Gamma^{(m,n)})}$.

Proof. For arbitrary $\langle \zeta_1, \jmath \rangle, \langle \zeta_2, \jmath \rangle \in I^{(m,n)}$, let $\underset{\sim}{\mathcal{A}}(\langle \zeta_1, \jmath \rangle) = \underset{\sim}{\mathcal{A}}(\langle \zeta_2, \jmath \rangle)$. Then, $\mathcal{A}(\varphi^{-1}(\langle \zeta_1, \jmath \rangle)) = \mathcal{A}(\varphi^{-1}(\langle \zeta_2, \jmath \rangle))$ from (5.2). Since $\mathcal{A}$ is an addressing function, hence one-to-one, $\varphi^{-1}(\langle \zeta_1, \jmath \rangle) = \varphi^{-1}(\langle \zeta_2, \jmath \rangle)$. The proved one-to-oneness of φ implies $\langle \zeta_1, \jmath \rangle = \langle \zeta_2, \jmath \rangle$. Thus $\underset{\sim}{\mathcal{A}}$ is one-to-one and the proposition follows from Definition 4.9. Q.E.D.

Now, for a given $\Gamma^{(m,n)}$ and its labeling scheme $(\mathcal{L}, \Sigma)$ specified, we describe the addressing algorithm of $\underline{Re(\Gamma^{(m,n)})}$ using the labeling table. This is achieved by simple table looking up. Let $\mathcal{A}$ be an arbitrary addressing function of $\Gamma^{(m,n)}$, and $\langle q_1 q_2 \cdots q_t, \jmath \rangle \in \underset{\sim}{I}^{(m,n)}$ be an arbitrary labeled index ($q_i \in L_i$, $1 \leqslant t \leqslant m$). Let ξ be a variable whose value is in $\{a_0, a_1, \cdots, a_{k-1}\}^*$, and ζ be a variable whose value is in $\bigcup_{1 \leqslant t \leqslant m} L_1 \times L_2 \times \cdots \times L_t$. NEXT($\zeta$) is an operator whose value is the first label in ζ, and REST(ζ) is an operator whose value is ζ with NEXT(ζ) removed.

(1) Let $\xi = e$ and $\zeta = q_1 q_2 \cdots q_t$.

(2) Let $I = 0$.

(3) Let $I = I + 1$. Scan the I-th row of the labeling table and find out NEXT(ζ).

(4) Obtain the function a_{i_I} in $\mathcal{a}$ corresponding to the position of NEXT(ζ).

(5) Let $\xi = \xi a_{i_I}$ and $\zeta = \text{REST}(\zeta)$.

(6) If $I \geqslant t$, calculate $\mathcal{A}(\langle \xi, \jmath \rangle)$. This result is the desired address of $\langle q_1 q_2 \cdots q_t, \jmath \rangle$. Stop.

(7) Go to (3).

Of course the address of $\langle e, j \rangle \in \underset{\sim}{I}^{(m,n)}$ can be obtained by calculating $\mathcal{A}(\langle e, j \rangle)$.

5.4 Conclusion

In this chapter, a labeling scheme for TA-structures is presented, and for a labeled TA-structure its addressing scheme is explained.

Now we conclude the merits of TA-structures (labeled TA-structures)

(i) The use of tree structures enlarges the scope of problems and their associated data which can be dealt efficiently. That is, in the solutions of many problems, the attributes or relations introduced on the set of data items are "dominate-dominated" hierarchical relations. These hierarchical relations can be represented directly and efficienly by employing tree structures.

(ii) Every data cell in TA-structures can store a data item; in "structures" available in PL/I or trees available in COBOL, any data cells except leaf cells cannot store data items.

(iii) A set of trees can be treated and the correspondence between two cells in two trees Γ_{t_1} and Γ_{t_2} can be described explicitly. Furthermore, each of these correspondence relations serves themselves as transition paths between Γ_{t_1} and Γ_{t_2}.

(iv) Labeled TA-structures are traverse-oriented structures; the linkages can be labeled so that a desired cell is accessed from an arbitrary cell by specifying the string labels of linkages between the two cells.

(v) The "local structures" of data stored in TA-structure can be easily specified by slicing out trees or arrays. Moreover, when a programmer refers to a cell in a tree or an array sliced out, he may concentrate his

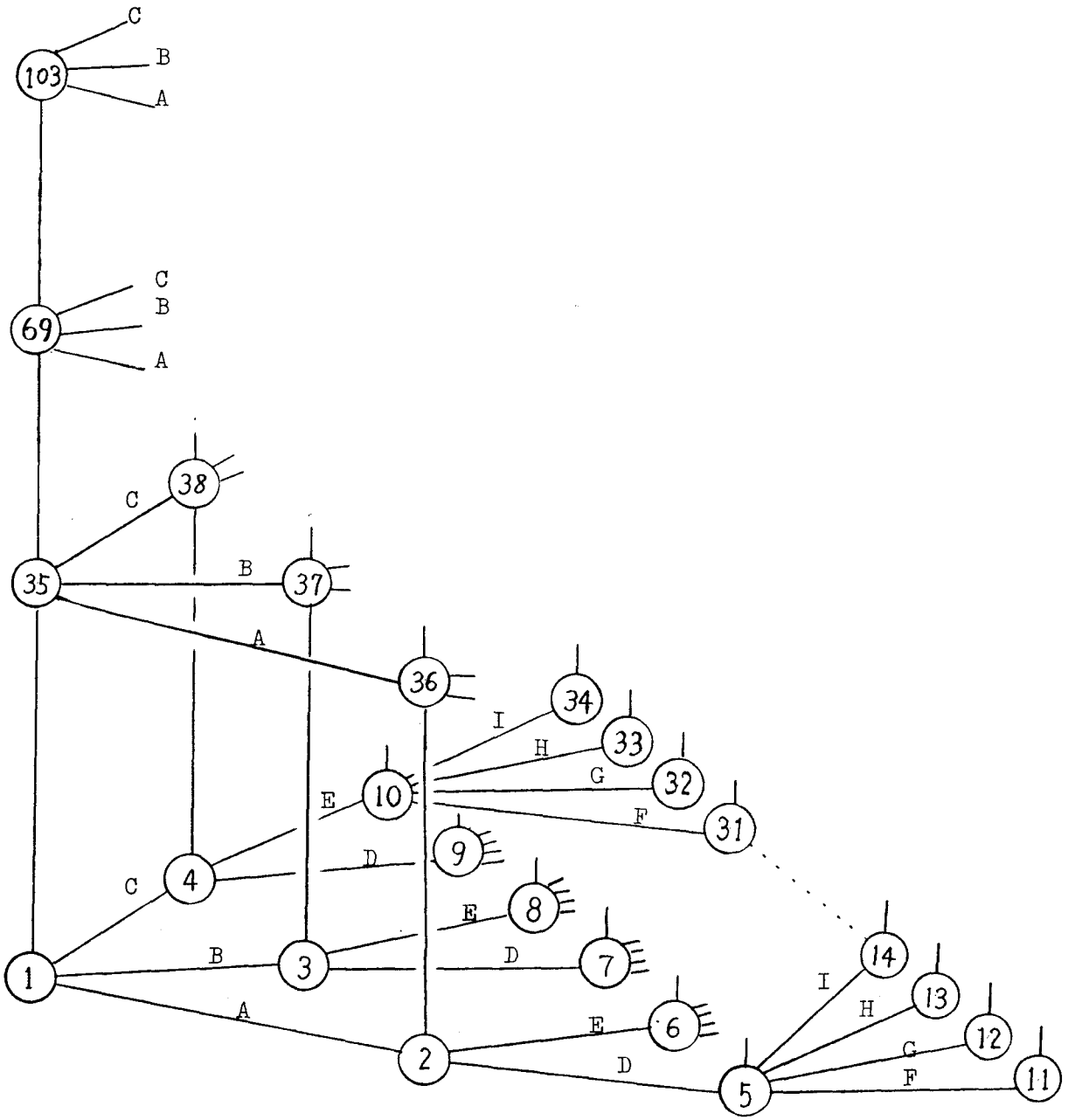
attention only on its position in it, without any care of its position in the entire TA-structure.

On the other hand, we indicate the demerits of TA-structures comparing with the data structures employed in the current high level programming languages.

(i) Only complete trees can be available. When we want to represent an irregular tree, we must embed it into some complete tree. Thus it is often the case that the availability of data cells deteriorates extremely, because of the so many unused cells.

(ii) Every data item stored in a cell of a TA-structure should have the same data type (e.g., integer type or real type), or more strictly, the same length.

These demerits are both inevitable limitations which stem from the realization scheme of TA-structures adopted. The relaxation of these limitations should be explored urgently; the use of other realization methods such as chaining by pointers may be needed together with our concentrating method, if necessary. It seems that a generalized TA-structure (and its labeling) such as exhibited in Fig.5.3 is the most relaxed structure which is "completely" directly accessible by an addressing function of "tolerable" complexity. In fact the addressing function in Fig.5.3 is fairely complex to compute and the cost of traversal is much high, comparing with the ordinary TA-structures.



	a_0	a_1	a_2	a_3
1	A	B	C	-
2	D	E	-	-
3	F	G	H	I

Figure 5.3 A generalized TA-structure and its labeling table

CHAPTER 6

CONCLUSION

In this thesis, we have discussed several problems concerning with data structures with addressing functions.

In Chapter 2 and Chapter 3, a model called a relational data graph is proposed, for studying uniformities in the structure of the "generalized" directed graphs underlying data structures. Several advantageous uniformities needed to design addressing functions are formulated and investigated in detail. Chapter 4 and Chapter 5 are devoted to the specific family of directly accessible data structures called TA-structures, each of which has a composite structure of trees and arrays. For TA-structures, their indexing methods, construction of addressing functions, and their labeling schemes are explained.

Although we have stated that the strong uniformities such as the two kinds of uniformly self-embeddabilities seem to be indispensable to devise simple index sets and efficient addressing functions, this is based on much intuitive ground and we have not suggested a systematic way to construct addressing functions for the class of relational data graphs with those uniformities.

To treat and discuss generally the construction process which is applicable to the wider class of relational data graphs such as those with root blocks, is a very difficult, but interesting and practically valuable persuit. For the success of this persuit, it seems an immediate task to establish a measure as far as possible for each of those quality criteria of addressing functions listed on Page 81. For example, it is possible

to treat the criterion (c) (complexity of traversal) as "a easy traversal means a traversal by (px + q)-type functions". This means that in $\Gamma = (C, R)$ (restricted to a functional graph) and its realization (σ, ρ) on N (the set of natural numbers), for each $c \in C$ and $f_i \in R$, the realization of f_i, namely $f_i\rho$ is given as follows.

$$(c\sigma)(f_i\rho) = p_i \times (c\sigma) + q_i \qquad (p_i, q_i: \text{constant})$$

Especially, in the case of finite arrays, for all $f_i \in R$, $p_i = 1$, and in the case of k-ary complete trees, for all successors $f_i \in R$, p_i may possibly be k.

In general, the three criteria on Page 81 are mutually conflicting and the weights assigned to each of those criteria are highly dependent on particular computing environments. It also seems an important and fruitful direction for further study to discuss the interplay among the criteria.

As for TA-structures, in spite of the disadvantages of TA-structures described previously, it seems a much significant work to design a programming language in which TA-structures are available and to construct a processor for the language, because of the various advantages of TA-structures outweighing their drawbacks.

TA-structures can be declared in an arbitrary programming language in which array structures are available, and allocated on a storage area. For we can transform a TA-structure $\Gamma^{(m,n)}$ into an array by linearly rearranging the data cells in each tree in $\Gamma^{(m,n)}$. For instance, for a TA-structure $\Gamma^{(m,n-1)}$ of dimensionality $\ll 2, k \gg$, we provide an array A of dimension (ρ_k^m, n); ρ_k^m is given on Page 83. Then each index $\langle \xi, j \rangle$

of $\Gamma^{(m,n-1)}$ is transformed into a new index $\langle \overline{a_1 \xi}, j \rangle$ of A.

In [41], we adopted Fortran as a language in which array structures are available. Based on Fortran, we design a language where TA-structures are declared and manipulated. It is an important work remained to dissolve the demerits of TA-structures and provide a user more convenient facilities.

LIST OF REFERENCES

[1] D.L. Child, Description of a set theoretic data structure, AFIPS Conf. Proc. FJCC 33, 1968, pp.557-564.

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

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

[4] H.-D. Ehrich, Theory of direct-access storage functions, in Information Processing 74 (J. Rosenfeld, ed.) North-holland, Amsterdam, 1974, pp.647-651.

[5] A.C. Fleck, Towards a theory of data structures, J. Comp. System Sciences, 5, 1971, pp.475-488.

[6] A.C. Fleck, Recent developments in the theory of data structures, Computer Languages, 5, 1978, pp.37-52.

[7] A.C. Fleck, K.-C. Liu, On the realization of data graphs, Univ. of Iowa Tech. Report, #67, June, 1973.

[8] S.P. Ghosh, File organization: the consecutive retrieval property, C. of ACM, 15, 1972, pp.802-808.

[9] H. Hellerman, Addressing multidimensional arrays, C. of ACM, 5, 1962, pp.204-207.

[10] Y. Inagaki, T. Sakabe, and T. Fukumura, Addressable approximations to nonaddressable data graphs, Proc. 2nd USA-Japan Computer Conference, 1975.

[11] T. Katayama, On the language theoretic class of LOREL data, J. of Inf. Processing Society of Japan, 14, 10, 1973, pp.754-761, (in Japanese)

[12] D.E. Knuth, The Art of Computer Programming I: Fundamental Algorithms, Addison-Wesly, Reading, Mass., 1968.

[13] D.E. Knuth, The Art of Computer Programming III: Sorting and Searching, Addison-Wesly, Reading, Mass., 1973.

[14] G.H. Mealy, Another look at data, AFIPS Conf. Proc. FJCC, 31, 1967, pp.525-534.

[15] J.L. Phaltz, Graph Structures, Journal of ACM, 19, 3, 1972, pp. 411-422.

[16] J.L. Phaltz, A. Rosenfeld, Web Grammars, Proc. Int. Joint Conf. on Artificial Intelligence, May, 1969, pp.609-619.

[17] A.L. Rosenberg, Data graphs and addressing schemes, J. Comp. System Sciences, 5, 1971, pp.193-238.

[18] A.L. Rosenberg, Symmetries in data graphs, SIAM J. Comput. 1, 1972, pp.40-65.

[19] A.L. Rosenberg, Addressable data graphs, J. of ACM, 19, 1972, pp. 309-340.

[20] A.L. Rosenberg, Suffixes of addressable data graphs, Inf. Contr. 23, 1973, pp.107-127.

[21] A.L. Rosenberg, Exploiting addressability in data graphs, in Computational Complexity: Courant Computer Science Symp. 7 (R. Rustin, ed.) Algorithmics Press, N.Y., 1973, pp.161-183.

[22] A.L. Rosenberg, Computed access in ragged arrays, in Information Processing 74 (J. Rosenfeld, ed.), North-Holland, Amsterdam, 1974, pp.642-646.

[23] A.L. Rosenberg, Allocating storage for extendible arrays, J. of ACM, 21, 1974, pp.652-670.

[24] A.L. Rosenberg, On storing arbitrarily many extendible arrays of arbitrary dimensions, Int'l J. Comp. Inf. Sci., 4, 1975, pp.189-196.

[25] A.L. Rosenberg, Managing storage for extendible arrays, SIAM J. Comput., 4, 1975, pp.287-306.

[26] A.L. Rosenberg, Direct-access storage of data structures, IBM Research Report, RC 6036, 1976.

[27] A.L. Rosenberg, L.J. Stockmeyer, Hashing schemes for extendible arrays, J. of ACM

[28] A.L. Rosenberg, What is a multilevel array?, IBM JRD, 19, 1975, pp.163-169.

[29] A. Rosenfeld, Algebraic Structures, Holden-Day, 1968.

[30] T. Sakabe, Y. Inagaki, and T. Fukumura, Infinite Hierarchy of Algebraic and Graphical Uniformities of Data Graphs, Trans. Inst. Elect. Commun. Engrs. of Japan, J60-D, 1977, pp.122-128, (in Japanese).

[31] B.A. Shneiderman, Data Structures: Description, Manipulation, and Evaluation, Ph.D. Thesis, SUNY at Stony Brook, 1973.

[32] L.J. Stockmeyer, Extendible array realizations with additive traversal, IBM Research Report RC-4578, 1973.

[33] T. Tsuji, Y. Ezawa, M. Mizumoto, J. Toyoda, and K. Tanaka, Generalized data graphs, Tech. Report of IECE of Japan, AL74-12, 1974.(in Japanese)

[34] T. Tsuji, Y. Ezawa, M. Mizumoto, J. Toyoda, and K. Tanaka, Relational data graphs and their uniformities, Record of the National Convention of IECE of Japan, 1975, p.1238 (in Japanese).

[35] T. Tsuji, Y. Ezawa, M. Mizumoto, J. Toyoda, and K. Tanaka, Relational data graphs, Trans. of IECE of Japan, 58-D, 9, 1975, pp.586-587(in Japanese)

[36] T. Tsuji, J. Toyoda, and K. Tanaka, Relational data graphs with uniformities and their several properties, Trans. of IECE of Japan, 59-D, 2, 1976, pp.109-11 (in Japanese).

[37] T.Tsuji, K. Tanaka, Tree-array composite structures induced from relational data graphs and their addressing functions, Record of the National Convention of IECE of Japan (Information Section), 1977, p.49, (in Japan).

[38] T. Tsuji, K. Tanaka, Tree-array composite structures and their addressing functions, Tech. Report of IECE of Japan, AL77-23, 1977 (in Japanese)

[39] T. Tsuji, J. Toyoda, and K. Tanaka, Relational data graphs and some properties of them, J. Comp. System Sciences, 15, 1, 1977, pp.17-34.

[40] T. Tsuji, K. Tanaka, Tree-array composite structures and their addressing functions, Trans. of IECE of Japan, Section D, (to appear) (in Japanese).

[41] T. Tsuji, K. Fujii, and K. Tanaka, On Fortran with tree-array composite data structures, Record of the National Convention of IECE of Japan, 1978 (in Japanese).

[42] W.M. Turski, A model for data structures and its applications I, Acta Informatica, 1, 1971, pp.26-34.

[43] W.M. Turski, A model for data structures and its applications II, Acta Informatica, 1, 1972, pp.282-289.