then $\psi'(\Omega) = \left\{\begin{matrix}(2,2,1,1,2)\\(2,2,2,1,2)\\(2,2,2,2,1)\end{matrix}\right\}$.

Since $\alpha = \psi'(\Omega)_{3,4,5}$, $\beta = \psi'(\Omega)_{2,3}$ and $\gamma = \psi'(\Omega)_{1,2}$, $\psi'(\Omega)$ is lossless. ///

Before we look at the correspondences between decompositions and syntheses, we define the following term which will be used in the following sections.

Definition: An n-ary relation α is constructable from Ω if there exists an n-synthesis $\psi(\Omega)$ such that $\alpha = \psi(\Omega)$.

Decompositions	Syntheses
derivable from α	constructable from Ω
$\beta = P_L(\alpha)$: Projection	$\alpha = \psi(\Omega)$: Synthesis
decomposition of α	loss-free synthesis of Ω
$\alpha = \underset{L_i}{*} P_{L_i}(\alpha)$	$\psi(L) = P_L(\alpha)$
maximal decomposition α^*	Ω: skeleton, ψ: loss-free full type 3
$\psi(\alpha^*) = \alpha$	$(\psi(\Omega))^* = \Omega$

Table 4.1

The correspondences between decompositions and syntheses are summarized in Table 4.1. Given a relation α, a set of projections $P_{L_i}(\alpha)$'s of α generate a set of relations Ω. Each relation of Ω is derivable from α.

Given a set of relations Ω a synthesis $\psi(\Omega)$ generates a relation α. α is constructable from Ω. A decomposition of α is a set of projections $P_{L_1}(\alpha)$, $P_{L_2}(\alpha),\ldots,P_{L_k}(\alpha)$ such that their natural join $P_{L_1}(\alpha)*P_{L_2}(\alpha)*\ldots*P_{L_k}(\alpha)$ is the same as α. A loss-free synthesis is a relation α such that for each essential factor L of the synthesis the L projection of α, $P_L(\alpha)$, is the same as the essential relation corresponding to L, i.e., $P_L(\alpha) = \psi(L)$ $\forall$ essential factor L.

4.4 Relational Spaces

If we regard projections and syntheses as operators on relations, we can think of the relation space which is generated from a set of relations by applying these operators. Here we are primarily concerned with the problem: what is the relation between the relation space of a given set of relations Ω and that of its skeleton Ω^*.

A relation α is constructable from a set of relations Ω if there exists a synthesis $\psi(\Omega)$ such that $\alpha = \psi(\Omega)$. Therefore we easily have the following:

Lemma 4.1 If α is constructable from Ω, and β is constructable from $\{\alpha\} \cup \Omega'$, then β is constructable from $\Omega \cup \Omega'$.

Definition: The type i m-constructable space, denoted by $\Gamma_i^m(\Omega)$, is the set of all m-ary relations which are constructable from Ω by type i m-syntheses, i.e., $\Gamma_i^m(\Omega) = \{\alpha \mid \alpha = \psi(\Omega)$ is an m-synthesis and ψ is type i$\}$, where i = 0, 1, 2, 3. In case of type 0, it may be referred to simply as the m-constructable space $\Gamma^m(\Omega)$.

The following is a simple result of the definitions.

Theorem 4.1 For any set of relations Ω,

$$\Gamma_0^m(\Omega) \supset \Gamma_1^m(\Omega) \supset \Gamma_2^m(\Omega) \supset \Gamma_3^m(\Omega).$$

Other simple facts about type i m-constructable spaces are: in general $\Omega \not\subset \Gamma_i^m(\Omega)$ and $\Gamma_i^m(\Gamma_i^m(\Omega)) = \Gamma_i^m(\Omega)$ by using Lemma 4.1.

On the other hand, it was earlier defined that a relation α is derivable from a relation β if α is an equivalent of some projection of β. It will be convenient to have the following operator.

Definition: The derivable space of a set of relations Ω, denoted by $\Delta(\Omega)$, is the set of all relations which are derivable from Ω. We may write $\Delta(\alpha)$ instead of $\Delta(\{\alpha\})$ if $\Omega = \{\alpha\}$.

Lemma 4.2 If α is derivable from β and β is derivable from γ, then α is derivable from γ.

It is straightforward; the proof is omitted.

Obviously in general $\Omega \not\subset \Delta(\Gamma_i^m(\Omega))$ for i = 1,2,3 while $\Omega \subset \Delta(\Gamma_0^m(\Omega))$ if Ω contains only relations of ranks m or less than m. Also it is self-evident that $\Omega \subset \Delta(\Gamma_i^m(\Omega))$ if Ω is a subset of the elementary relations of some m-ary relation. In the converse case, again $\Delta(\Delta(\alpha)) = \Delta(\alpha)$ but $\alpha \in \Gamma_i^m(\Delta(\alpha))\ \forall$ i since all elementary relations of α are in $\Delta(\alpha)$ provided that the rank of α is m. We summarize the above arguments below.

Theorem 4.2 Let α be an m-ary relation and Ω consist of relations of rank m or less, then the following are true:

(1) $\Omega \not\subset \Gamma_i^m(\Omega)$,

(2) $\Gamma_i^m(\Gamma_i^m(\Omega)) = \Gamma_i^m(\Omega)$,

(3) $\Omega \subset \Delta(\Gamma_0^m(\Omega))$,

(4) $\alpha \in \Delta(\alpha)$,

(5) $\Delta(\Delta(\alpha)) = \Delta(\alpha)$,

(6) $\alpha \in \Gamma_i^m(\Delta(\alpha))$.

Next we will extend the concept of m-constructable spaces. Hereafter we will not specify the type of syntheses used. We tacitly assume

that they are the most general type, type 0. But, of course, the following discussions are valid even when the syntheses are restricted to a certain type since the inclusion relations among their m-constructable spaces which were shown earlier are applicable. Now suppose that a set of relations is given. Then the set of relations which are constructable is infinite because the ranks of the syntheses which are used for constructions are not specified, which leads to the following concept.

Definition: The set of all relations that are constructable from a set of relations Ω is termed the constructable space of Ω, denoted by $\Gamma(\Omega)$. That is, $\Gamma(\Omega) = \cup_{i=0}^{\infty} \Gamma^i(\Omega)$.

Let us consider the following problem.

Lemma 4.3 For any set of relations Ω, if a relation is constructable from Ω, then it is also constructable from the skeleton of Ω, Ω^*, i.e.,

$$\Gamma(\Omega) \subset \Gamma(\Omega^*).$$

Proof Let a relation β be constructable from Ω, i.e., $\beta = \psi(\Omega)$ where $\psi: \Sigma \longrightarrow \hat{\Omega}$. We will show that $\beta = \psi'(\Omega^*)$ by some ψ' which is obtained by modifying ψ as follows. Let $\psi(x) = \alpha^i, \alpha^i \in \Omega$ and α^i decomposable, then $(\alpha^i)^* \subset \Omega^*$. For each such x we define $\psi'(x) = S^k$ where $|x| = k$ and $\psi'(y) = (\alpha^i)_y$ for each (k-1)-factor y of x where $(\alpha^i)_y$ denotes the (k-1)-ary partial relation corresponding to y except for those y's such that $\psi(y) \subset (\alpha^i)_y$, in which case we define $\psi'(y) = \psi(y)$. Then for those y's such that $\psi(y) = (\alpha^i)_y$ is decomposable, we apply the above process. Repeat this procedure until all lower relations are not decomposable; this corresponds to the maximal decomposition of α^i. Then we claim that the synthesis of Ω^* thus constructed, $\psi'(\Omega^*)$, yields β. For α^i is equivalent to, as a constraint for all $B \in \beta$, $\hat{y}\{B_y \in \psi(y)\} \wedge \{B_x \in \alpha^i\} = \{y \mid \psi(y) \overset{\wedge}{\subset} (\alpha^i)_y\} \{B_y \in \alpha(y)\} \wedge \{B_x \in \alpha^i\}$ which is, in turn, the same as $\{y \mid \psi(y) \overset{\wedge}{\subset} (\alpha^i)_y\}\{B_y \in \psi(y)\} \{y \mid \psi(y) \overset{\wedge}{\supset} (\alpha^i)_y\}\{B_y \in (\alpha^i)_y\}$

$\wedge \{B_x \varepsilon \alpha^i\}$. The same argument is applicable to the factors of ranks lower than k.

Q.E.D.

So far we have considered relational spaces for each operator separately. Next we define relational spaces which are obtained by using both projection and synthesis.

Definition: Let Ω be a set of relations, then the relation space of Ω, denoted by $\Sigma(\Omega)$, is the set of all relations which can be obtained from Ω by using any number of projections and syntheses.

The following lemma is a straightforward consequence of the definitions.

Lemma 4.4 If $\Omega \subset \Omega'$, then

(1) $\Gamma^m(\Omega) \subset \Gamma^m(\Omega')$,

(2) $\Gamma(\Omega) \subset \Gamma(\Omega')$,

(3) $\Delta(\Omega) \subset \Delta(\Omega')$,

(4) $\Sigma(\Omega) \subset \Sigma(\Omega')$.

Let Ω be a set of relations, then any relation of Ω can be constructable from its skeleton Ω^* by the definition. Therefore $\Omega \subset \Gamma(\Omega^*)$. Then since $\Sigma(\Omega) \subset \Sigma(\Gamma(\Omega^*)) = \Sigma(\Omega^*)$, we have the following.

Theorem 4.3 For any set of relation Ω,

$$\Sigma(\Omega) \subset \Sigma(\Omega^*).$$

In practice we may use other operations, especially set operations such as union, intersection, and complement, in relational systems. However the above results hold even if set operations are added to the operation set for relational spaces.

4.5 Compaction of Relations

Another interesting problem is how compactly can we express a set of relations? In this section we consider only full syntheses unless stated otherwise. Since the original relations are derivable from a lossless synthesis, lossless syntheses provide a means by which it is sufficient to keep only one relation instead of many relations. Obviously it is always possible, given a set of relations, to have a lossless synthesis if the rank of $\psi(\Omega)$ is increased as high as desired. Naturally what we want to have among such lossless syntheses will be one of the lowest possible rank.

Definition: (m-integration) A set of relations Ω is said to be m-integrable if there exists a full and lossless synthesis $\psi(\Omega)$ of rank m; such a synthesis $\psi(\Omega)$ is called an m-integration of Ω.

First, we look at what happens to the m-integrability if a set of relations is modified in certain ways.

Definition: A set of relations is redundant if some relation is derivable from another, and non-redundant otherwise.

Lemma 4.5 If a set of relations Ω is m-integrable, then so is a non-redundant subset Ω' of Ω.

Proof Since Ω is m-integrable, there is an m-ary synthesis $\psi(\Omega) \Rightarrow \psi: \Sigma \longrightarrow \hat{\Omega}$. Now let ψ' be a mapping $\psi': \Sigma \longrightarrow \Omega'$ such that $\psi'(x) = \psi(x)$ if $x \in \Sigma$ and $\psi(x) \subset \Omega' \subset \Omega$ and $\psi'(x) = S^{|x|}$ otherwise, then clearly $\psi'(\Omega')$ is lossless. Hence $\psi'(\Omega')$ is an m-integration of Ω'.

Q.E.D.

Lemma 4.6 If a set of relations Ω is m-integrable, then so is a set of relations Ω' which is obtained from Ω by replacing a relation $\alpha \in \Omega$ of rank k which is (k-1)-ary decomposable by all its (k-1)-ary partial

relations, i.e., $\Omega' = \{\Omega-\alpha\} \cup \{$all (k-1)-ary partial relations of $\alpha\}$.

Proof By the hypothesis, α is derivable from an m-integration of Ω, $\psi(\Omega)$. This is possible in two ways. First, α is not essential but α is derivable from $\psi(\Omega)$, or $\exists\ x \in \Sigma \ni \psi(x) - \alpha$ but $\psi(\Omega)_x \neq \alpha$. Secondly, α is essential, i.e., $\exists\ x \in \Sigma \ni \psi(x) = \alpha$ and α is derivable from $\psi(\Omega)_x$. Let ψ' be defined as follows. In the first case, let $\psi' = \psi$. In the second case we set $\psi'(x) = S^k$ for $x \in \Sigma \ni \psi(x) = \alpha$ and $\psi(\Omega)_x = \alpha$. We set the value of each (k-1)-factor of the x by ψ' to the corresponding (k-1)-ary partial relation of α. Now in order for $\psi'(\Omega')$ to be an m-integration of Ω', we have to show that $\psi'(\Omega')$ is lossless. Since α is derivable from $\psi(\Omega)$, clearly k (k-1)-ary partial relations of α are derivable from $\psi'(\Omega')$. Next suppose that there is a relation $\beta \in \Omega'$, which was derivable from $\alpha(\Omega)$ but is not derivable from $\psi'(\Omega')$. Then this means that it was caused by the replacement of α by its (k-1)-ary partial relations, which contradicts the assumption that α is (k-1)-ary decomposable, which guarantees that the restrictions used to construct $\psi'(\Omega')$ are essentially the same as those used to construct $\psi(\Omega)$. Therefore $\psi'(\Omega')$ is lossless, proving the lemma.

Q.E.D.

By applying the two procedures described in Lemmas 4.5 and 4.6, we can show the following.

Theorem 4.4 If a set of relations Ω is m-integrable, then so is its skeleton Ω^*.

Let us suppose, by limiting the above general case, that all relations in Ω are binary decomposable, then the following is an immediate result of Theorem 4.4.

Corollary 4.1 If a set of relations Ω is m-integrable and all relations in Ω are binary decomposable, then Ω' = {all binary partial relations of $\alpha \mid \alpha \in \Omega$} is also m-integrable.

So far in this section it has been shown that the modifications of Ω described above do not increase the rank of integrations. Next, what can we say about a set of relations Ω if it is known that Ω is m-integrable? It is easy to see that if a non-redundant set Ω consists of only binary relations and is m-integrable, the number of relations in Ω has to be less than $m(m-1)/2$, i.e, $|\Omega| \leq m(m-1)/2$. How about the general case?

Definition: The dimension of a set of relations Ω, denoted by $d(\Omega)$, is the number of all non-equivalent binary relations derivable from the relations in Ω.

Example 4.4 Let $\Omega = \{\alpha,\beta\}$ and let $\alpha, \beta \subset S^3$ be defined as follows:

$$\alpha = \begin{Bmatrix}(1,1,1)\\(1,1,2)\\(1,2,2)\\(2,1,1)\\(2,2,1)\\(2,2,2)\end{Bmatrix}, \qquad \beta = \begin{Bmatrix}(1,1,1)\\(1,1,2)\\(1,2,1)\\(1,2,2)\end{Bmatrix}.$$

Then all binary relations derived from α and β are:

$$\alpha_{12} = \alpha_{13} = \alpha_{23} = \alpha_{23} = \begin{Bmatrix}(1,1)\\(1,2)\\(2,1)\\(2,2)\end{Bmatrix}$$

$$\beta_{12} = \beta_{13} = \begin{Bmatrix}(1,1)\\(1,2)\end{Bmatrix}.$$

Therefore the dimension of Ω is two, i.e., $d(\Omega) = 2$. ///

We can conclude the following about the dimension by adding an additional constraint.

Theorem 4.5 If a set of relations which are all binary decomposable is m-integrable, then $d(\Omega) \leq m(m-1)/2$.

Proof By induction on the highest rank n of relations in Ω.

(i) Obviously it is true for n = 2, in which case d(non-redundant set of Ω) = $d(\Omega)$.

(ii) Suppose that it is true for $r < n$. Let $\{\alpha_i\}_{i\in I}$ be the relations of the highest rank n in Ω, then by Lemma 4.6 $\Omega' = \{\Omega - \{\alpha_i\}_{i\in I}\} \cup$ { all (n-1)-ary partial relations of $\alpha_i \mid \alpha_i \in \{\alpha_i\}_{i\in I}\}$ is also m-integrable. And clearly $d(\Omega') = d(\Omega)$. Since $d(\Omega') \leq m(m-1)/2$ by the hypothesis, $d(\Omega) \leq m(m-1)/2$.

Therefore, from (i) and (ii), it is true for any Ω.

Q.E.D.

Now since a synthesis is the construction of a higher rank relation from a set of relations Ω, and binary relations are the most fundamental, we finally consider the important special case in which Ω contains only binary relations. Therefore it is assumed to the end of this section that Ω contains only binary relations. First, we define the following convenient terms.

Definition: Let α be a binary relation, then the front of α is $f(\alpha) = \{a \mid (a,b) \in \alpha\}$ and the tail of α is $t(\alpha) = \{b \mid (a,b) \in \alpha\}$

Actually the front and the tail are not new concepts. In other words, $f(\alpha) = P_1(\alpha)$ and $t(\alpha) = P_2(\alpha)$. Also note that $f(\alpha), t(\alpha) \subset S$ and $f(\alpha) = t(\alpha^{-1})$, $t(\alpha) = f(\alpha^{-1})$.

Definition: Let α be a binary relation on S and $T \subset S$, then T is a base of α if $f(\alpha) = T$ or $t(\alpha) = T$.

Let Ω be non-redundant and $|\Omega| = m(m-1)/2$. Our final goal is to characterize the m-integrability of such Ω, which is the least possible case but the most desirable case from the data condensation point of view. First, suppose that Ω is m-integrable, then one of the conclusions is that the m-integration $\psi(\Omega)$ must be type 3. What can be said about binary relations in Ω?

Lemma 4.7 If $\Omega = \{\alpha_i \mid i = 1,2,\ldots,m(m-1)/2\}$ is non-redundant and m-integrable, then for each $\alpha_i \in \Omega$ $\exists$ two subsets of Ω, $\{\alpha_j\}_{j\in J}$ and $\{\alpha_k\}_{k\in K}$ such that $|J| = |K| = m-2$ and α_j has a base $f(\alpha_j)$ $\forall$ $j \in J$ and α_k has a base $t(\alpha_i)$ $\forall$ $k \in K$, and moreover $J \cap K = \emptyset$.

Proof Since all $m(m-1)/2$ relations are not equivalent and the number of binary partial relations of m-ary synthesis $\beta = \psi(\Omega)$ is $m(m-1)/2$, each partial relation is equivalent to exactly one α_i. Therefore let the relation in Ω which is equivalent to the partial relation β_{pq} be denoted by α_{pq}, and let us rename $\{\alpha_{pq}\}_{1 \le p < q \le m}$ as $\{\alpha'_{pq}\}_{1 \le p < q \le m}$ according to the rule that $\alpha'_{pq} = \alpha_{pq}$ if $\alpha_{pq} = \beta_{pq}$ and $\alpha'_{pq} = \alpha_{pq}^{-1}$ if $\alpha_{pq} = \beta_{pq}^{-1}$. Then since the set of binary partial relations of $\beta = \psi(\Omega)$, $\{\beta_{pq}\}_{1 \le p < q \le m}$, has the properties described in the lemma and there is a one-to-one correspondence between $\{\beta_{pq}\}_{1 \le p < q \le m}$ and $\{\alpha'_{pq}\}_{1 \le p < q \le m}$, the lemma follows immediately.

Q.E.D.

Before we go on to the converse of Lemma 4.7 so that the characterization of the m-integrability of our case is completed, the following is given.

Lemma 4.8 Let Ω be m-integrable, then the number of different bases, i.e., $f(\alpha_i)$'s and $t(\alpha_i)$'s, of binary relations α_i in Ω is at most m.

Proof Let $\beta = \psi(\Omega)$ be an m-ary synthesis since Ω is m-integrable, then $\exists$ a binary partial relation $\beta_{pq}(1 \leq p < q \leq m) \Rightarrow \beta_{pq} = \alpha_s$ or α_s^{-1} $\forall$ $\alpha_s \in \Omega$. On the other hand, $f(\beta_{ij}) = f(\beta_{k\ell})$ if $i = k$ and $t(\beta_{ij}) = t(\beta_{k\ell})$ if $j = \ell$. But there are only m different subscripts $1,2,\ldots,m$. This proves the first part of the lemma.

For the second half, $\alpha_i \neq \alpha_j$ if $\alpha_i \neq \alpha_j$ $\forall$ $\alpha_i, \alpha_j \in \Omega$ since Ω is non-redundant. Since there is a one-to-one correspondence between $\{\alpha_i$ or $\alpha_i^{-1} \mid \alpha_i \in \Omega\}$ and $\{\beta_{pq}\}_{1 \leq p < q \leq m}$, it is also true that $\beta_{ij} \neq \beta_{k\ell}$ if $ij \neq k\ell$ $\forall$ $\beta_{ij}, \beta_{k\ell} \in \{\beta_{pq}\}_{1 \leq p < q \leq m}$. Suppose that $f(\beta_{ij}) = f(\beta_{k\ell})$ $i \neq k$, there are three non-trivial cases ($m > 2$):

(i) $\exists$ $s \Rightarrow s < i,k$, then $\beta_{si} = \beta_{sk}$ since $f(\beta_{si}) = f(\beta_{sk})$ and $t(\beta_{si}) = t(\beta_{sk})$.

(ii) $\exists$ $s \Rightarrow i < s < k$, then $\beta_{is} = \beta_{sk}$ since $f(\beta_{is}) = t(\beta_{sk})$ and $t(\beta_{is}) = f(\beta_{sk})$.

(iii) $\exists$ $s \Rightarrow i,k < s$, then $\beta_{is} = \beta_{ks}$ since $f(\beta_{is}) = f(\beta_{ks})$ and $t(\beta_{is}) = t(\beta_{ks})$.

And all these cases contradict the assumption that $\beta_{ij} \neq \beta_{k\ell}$ $\forall$ $ij \neq k\ell$. Similarly it can also be shown that $f(\beta_{ij}) = t(\beta_{k\ell})$ $i \neq \ell$ and $t(\beta_{ij}) = t(\beta_{k\ell})$ $j \neq \ell$ for some β_{ij} and $\beta_{k\ell}$ induce contradictions. Therefore the number of different $f(\beta_{ij})$'s and $t(\beta_{ij})$'s, hence the number of different $f(\alpha_i)$'s and $t(\alpha_i)$'s, is at most m.

Q.E.D.

Theorem 4.6 Let Ω be a set of binary relations such that Ω is non-redundant and $|\Omega| = m(m-1)/2$, then Ω is m-integrable iff for each $\alpha_i \in \Omega$ $\exists$ two subsets of relations of Ω, $\{\alpha_j\}_{j \in J}$ and $\{\alpha_k\}_{k \in K}$, such that $|J| = |K| = m-2$, $f(\alpha_i)$ is a base of α_j $\forall$ $j \in J$ and $t(\alpha_i)$ is a base of α_k $\forall$ $k \in K$, and $J \cap K = \emptyset$.

Proof Since Lemma 4.7 already showed one way, we will prove the reverse way, that is, we will construct an m-integration $\psi(\Omega)$, given the conditions of the second half of the statement of the theorem.

Take one relation $\alpha_{12} \in \Omega$ and let $\Omega_1 = \{\alpha_{13}, \alpha_{14}, \ldots, \alpha_{1m}\}$ be a set of m-2 relations which have $f(\alpha_{12})$ as a base and $\Omega_2 = \{\alpha_{23}, \alpha_{24}, \ldots, \alpha_{2m}\}$ be a set of m-2 relations which have $t(\alpha_{12})$ as a base, which are both guaranteed by the hypothesis. Then we transform Ω_1 to $\Omega_1' = \{\alpha_{12}', \alpha_{13}', \ldots, \alpha_{1m}\}$ by reversing the relations if necessary so that $\alpha_{12} = \alpha_{12}'$ and $f(\alpha_{12}') = f(\alpha_{1i}')$ $\forall$ i. Also we rearrange Ω_2 to $\Omega_2' = \{\alpha_{23}', \alpha_{24}', \ldots, \alpha_{2m}'\}$ similarly so that $f(\alpha_{2i}') = t(\alpha_{12})$ $\forall$ $i \neq 1,2$. Note that $t(\alpha_{1i}') \neq t(\alpha_{ij}')$ $\forall$ $i \neq j$ since Ω is non-redundant. And, by Lemma 4.8, all different bases appear in Ω_1'. Therefore every $\alpha \in \{\Omega - \{\Omega_1 \cup \Omega_2\}\}$ can have neither $f(\alpha'_{12})$ nor $t(\alpha_{12}')$. For otherwise α becomes equivalent to some relation in $\{\Omega_1' \cup \Omega_2'\}$.

Next take α_{13}', then the only possibility for the set of m-2 relations which have $f(\alpha_{13}')$ as a base is $\{\Omega_1 - \Omega_{13}\}$. And the set of m-2 relations which have $t(\alpha_{13}')$ as a base has to contain exactly one relation in Ω_2, say α_{23}' without loss of generality, and m-3 relations in $\Omega - \{\Omega_1 \cup \Omega_2\}$. Let the set of Ω_2 without α_{23}' be denoted by Ω_3 and be arranged to $\Omega_3' = \{\alpha_{34}', \alpha_{35}', \ldots, \alpha_{3m}'\}$. Next take α'_{14} and let the set of m-2 relations which have $t(\alpha_{14}')$ as a base be $\Omega_4 \cup \{\alpha_{24}, \alpha_{34}\}$ or, equivalently, $\Omega_4' \cup \{\alpha_{24}', \alpha_{34}'\}$ as the transformed form.

We proceed this way. Then eventually we have one relation left in $\Omega - \{\Omega_1 \cup \Omega_2 \cup \ldots \cup \Omega_{m-2}\}$ since we take (m-i-1) relations from the remaining set each time we work on α_{1i}'. Let the last one be $\Omega_{m-1} = \{\alpha_{m-1,m}\}$ and $\Omega_{m-1}' = \{\alpha_{m-1,m}'\}$, then $t(\alpha_1',m-1) = t(\alpha_{2,m-1}') = \ldots = t(\alpha_{m-2,m-1}') = f(\alpha_{m-1,m}')$. Note that $t(\alpha_{1,m}') = t(\alpha_{2,m}') = \ldots = t(\alpha_{m-1,m}')$.

Let $\psi(x) = \alpha_x'$ if $|x| = 2$ and $\psi(x) = S^x$ if $|x| \neq 2$. Then we obtain

an m-synthesis of Ω, $\psi(\Omega)$, from which all $\alpha_i \in \Omega$ are derivable. That is, we obtained an m-integration of Ω.

Q.E.D.

Finally we give an example of the compaction of binary relations which will hopefully clarify the procedure described above.

Example 4.4 Let Ω be a non-redundant set of the following binary relations on S = {a,b,c}:

$$\alpha_1 = \begin{Bmatrix}(a,a)\\(a,b)\end{Bmatrix}, \quad \alpha_2 = \begin{Bmatrix}(a,a)\\(a,c)\end{Bmatrix}, \quad \alpha_3 = \begin{Bmatrix}(a,b)\\(a,c)\end{Bmatrix},$$

$$\alpha_4 = \begin{Bmatrix}(a,a)\\(a,c)\\(b,a)\\(b,c)\end{Bmatrix} \quad \alpha_5 = \begin{Bmatrix}(a,b)\\(a,c)\\(b,b)\\(b,c)\end{Bmatrix}, \quad \alpha_6 = \begin{Bmatrix}(a,b)\\(a,c)\\(c,b)\\(c,c)\end{Bmatrix}.$$

Note $|\Omega| = 6 = 4(4-1)/2$. Hence m = 4.

Then this set of relations satisfies the condition described in the theorem. For, let us examine the front and the tail of each relation.

$$f(\alpha_1) = f(\alpha_2) = f(\alpha_3) = \{a\}$$

$$t(\alpha_1) = f(\alpha_4) = f(\alpha_5) = \{a,b\}$$

$$f(\alpha_2) = f(\alpha_1) = f(\alpha_3) = \{a\}$$

$$t(\alpha_2) = t(\alpha_4) = f(\alpha_6) = \{a,c\}$$

$$f(\alpha_3) = f(\alpha_1) = f(\alpha_2) = \{a\}$$

$$t(\alpha_3) = t(\alpha_5) = t(\alpha_6) = \{b,c\}$$

$$f(\alpha_4) = t(\alpha_1) = f(\alpha_5) = \{a,b\}$$

$$t(\alpha_4) \quad t(\alpha_2) = f(\alpha_6) = \{a,c\}$$

$$f(\alpha_5) = t(\alpha_1) = f(\alpha_4) = \{a,b\}$$

$$t(\alpha_5) = t(\alpha_3) = t(\alpha_6) = \{b,c\}$$

$$f(\alpha_6) = t(\alpha_2) = t(\alpha_4) = \{a,c\}$$

$$t(\alpha_6) = t(\alpha_3) = t(\alpha_5) = \{b,c\}$$

Therefore we can compact these six binary relations into one 4-ary relation. Actually, following the procedure described above, $\Omega_1' = \{\alpha_1, \alpha_2, \alpha_3\}$ so that $f(\alpha_1) = f(\alpha_2) = f(\alpha_3)$. And $\Omega_2' = \{\alpha_4, \alpha_5\}$ so that $f(\alpha_4) = f(\alpha_5) = t(\alpha_1)$. At this point we have one relation α_6 left. Let $\Omega_3' = \{\alpha_6\}$, then $f(\alpha_6) = t(\alpha_2) = t(\alpha_4)$ and $t(\alpha_6) = t(\alpha_3) = t(\alpha_5)$. In this case we did not reverse any relation because we had arranged them that way in advance to avoid the complexity.

Now we have an 4-integration of Ω by defining ψ as follows:

$$\psi: \quad \begin{array}{ll} (1,2) \longrightarrow \alpha_1 & (2,3) \longrightarrow \alpha_4 \\ (1,3) \longrightarrow \alpha_2 & (2,4) \longrightarrow \alpha_5 \\ (1,4) \longrightarrow \alpha_3 & (3,4) \longrightarrow \alpha_6 \end{array}$$

and all other factors are non-essential.

Then, indeed,

$$\psi(\Omega) = \left\{\begin{array}{l} (a,a,a,b) \\ (a,a,a,c) \\ (a,a,c,b) \\ (a,a,c,c) \\ (a,b,a,b) \\ (a,b,a,c) \\ (a,b,c,b) \\ (a,b,c,c) \end{array}\right\}$$

and $P_{12}(\psi(\Omega)) = \alpha_1$, $P_{13}(\psi(\Omega)) = \alpha_2$, $P_{14}(\psi(\Omega)) = \alpha_3$,

$P_{23}(\psi(\Omega)) = \alpha_4$, $P_{24}(\psi(\Omega)) = \alpha_5$, $P_{34}(\psi(\Omega)) = \alpha_6$.

Therefore $\psi(\Omega)$ is lossless. ///

CHAPTER V

RELATIONAL DATA STRUCTURE

5.1 Introduction

Before we go back to the subject of relational systems, let us summarize some results relevant to this chapter.

1. Relations are sets.

2. There exist isomorphic relations.

3. Most properties of binary relations are extendable to higher rank relations.

4. Some relations are decomposable while some are not.

5. The set of elementary relations of a relation is enough for the lossless synthesis of the relation.

6. If a relation can be generated from a set of relations, then it can be generated from the skeleton of the set, i.e. $\Sigma(\Omega) \subset \Sigma(\Omega^*)$.

7. Up to $m(m-1)/2$ binary relations can be compacted into one m-ary relation.

The significance of the decomposition theorem is that one needs not only unary and binary relations but also higher rank relations to specify a relational system, i.e., there is no way to describe a system by only unary and binary relations. Let a relational system be (S,Ω), then by (6) above $\Sigma(\Omega) \subset \Sigma(\Omega^*)$. Therefore the replacement of Ω by Ω^* does not affect the power of the system at all. Now let $\alpha = \psi(\Omega^*)$ be a compaction of Ω^*, then $\Sigma(\Omega) \subset \Sigma(\alpha)$ since $\psi(\Omega^*)$ is lossless. This means that we can replace (S,Ω) by (S,α), one relation on S. Furthermore the process, decomposition and compaction, may possibly provide a reduction of the

storage space. However in practice one may not always want this type of compaction since not only storage space but also the efficiency of the processing is a big factor for the evaluation of real systems. There is always a trade-off between time and space.

Now let us look at some data structures for relational systems. Many previous relational systems [12,37] had built-in data structures inherent to their implementation languages even if it was known they needed more efficient data management. Most of them were implemented in LISP. For example, in SIR by Raphael the data structure consists of a set of elements S each of which has a property list. Each property list has the list of (relation name, elements) pairs, which indicate that the element is related to the elements in the pair by the relation in the pair. In short, they are structured by lists. It is quite obvious that we need a data structure independent of the embedding languages.

The current tendency in information processing makes the description of patterns and concepts more and more important. And, as mentioned earlier, data management will become critical in such systems because the amount of data is very large. On the other hand, data base systems are becoming increasingly flexible and powerful. McGee proposed a labeled graph model [25] and Codd described a relational model [7] for data management system. However it has been already pointed out that a labeled graph model is still too limited in its power of representation. How about Codd's model? Fundamentally his model is the most general. But the relational model, at least by his description, is primarily intended for formatted tables. His model assumes the existence of keys in all tables and, therefore, only uses decomposition into two partial relations. In this sense his relational model is very specialized. We have to extend his model in

order to use it for relational systems.

The relational data structure described here can be regarded as an extension of not only Codd's relational model but also of LEAP's associative data structure [28] and Childs' set theoretical data structure [4]. In an associative data structure, an association is a triple (A, O, V) where A is an attribute, O if an object, and V is a value. There are three tables, an attribute table, an object table and a value table, to speed up the process. LEAP's model can be regarded as the special case of the relational data structure which allows only unary and binary relations. This limitation makes it possible to find an efficient storage structure. A set theoretical data structure consists of a table of sets and a table of elements. Sets may be, for example, "male", "female", "white" and "black". Each element has a list of set names in which the element appears. For example, the element "John" may have a list of set names "male", "white", "35 years old" and "professor". Obviously the model uses only unary relations. Since relations are sets, one can define relations of any rank by using Childs' "complex"; however, it lacks operations on relations.

5.2 Definition of Relational Data Structure

The relational data structure D consists of a set of elements E, a set of relations R and a set of operations on relations O, i.e., D = (E, R, O). The structure is schematically shown in Fig. 5.1. The set of elements is represented by the element name list (ENL) and the element data (ED). Each element is assigned its own unique internal name, and the information about the element is kept in ED. For example, ED may keep the list of all relation names in which the element appears. The set of relations R is represented as follows. The relation name list (RNL) keeps all relations

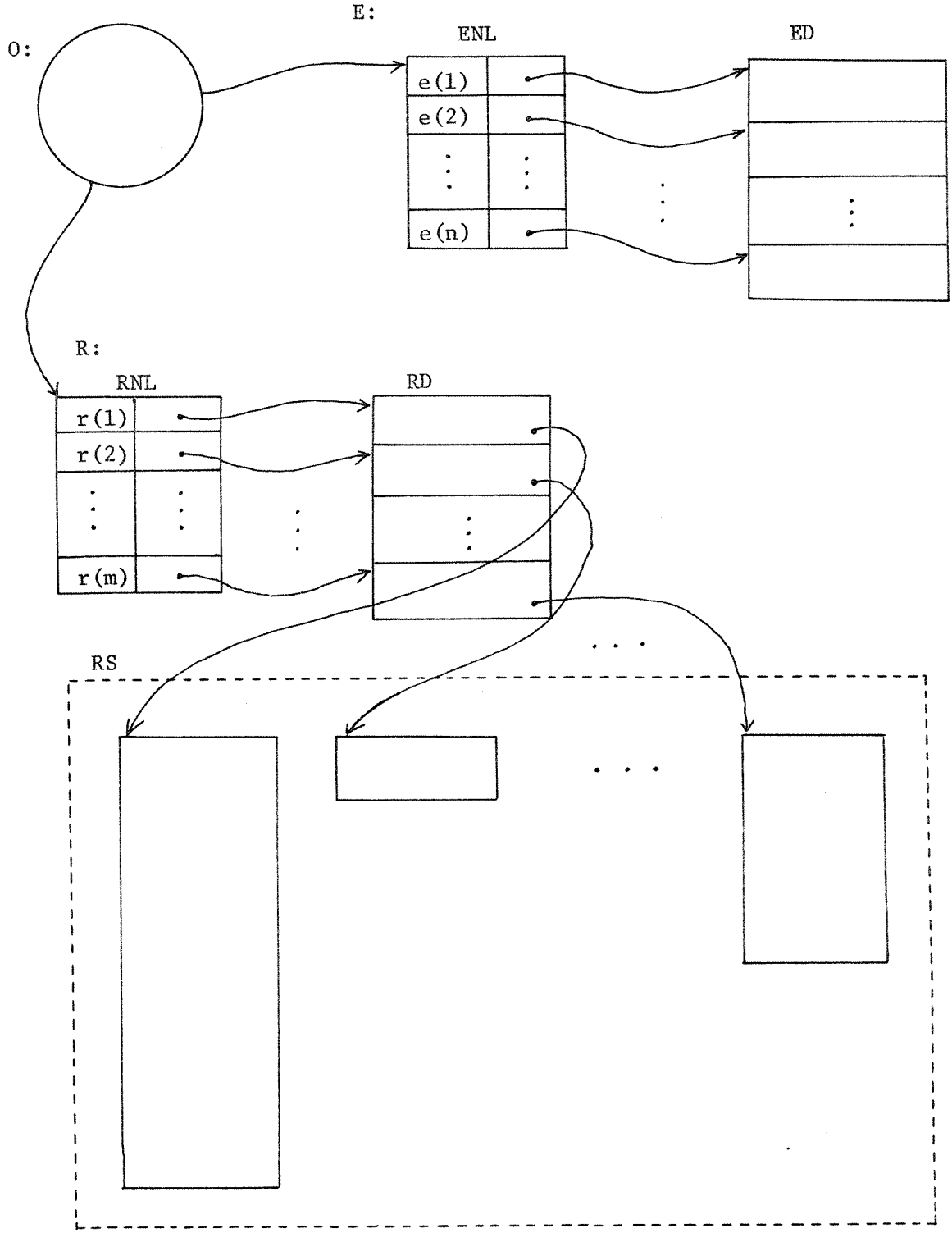


Fig. 5.1

known to the system by their internal names. Each relation in RNL has a pointer to the data of the relation in the relation data (RD). This RD holds various information about relations, including a pointer to the description of each relation. The description may be a table of n-tuples or a description of synthesis or projection. The symbol for this distinction may be kept in RD. The set of operations O contains all operations defined in the data structure and provides all necessary and fundamental operations for the efficient manipulation of data.

Since this is a data structure for relational systems, every operation involves at least either specific elements or specific relations. If some relation is specified, we go to RNL. If elements are specified, we go to ENL. One can think of many other supplementary components to speed up the processes. An extreme case may be some sort of inverted files. However, the most desirable thing in the sense of speed is the availability of associative memory. Anyway the above is the fundamental structure of the relational data structure. Next we explain each part of the data structure in more detail.

ENL is the table of element names with pointers. Each element is assigned a unique internal name at the time the element becomes known to the system. ENL is ordered for fast access to the elements. The use of hashing may make the system very complicated. Corresponding to each element name is the pointer to the record of the element.

ED is the set of records. Therefore it does not need to be ordered. Also the sizes of the records do not need to be the same. Each record may contain the code of the external name as well as the list of the relations in which the element appears.

RNL is the table of relation names. Each relation has its unique name internally. Relations also come and go like elements. Again RNL is ordered. Although some relations may have the same names as elements externally, their internal names should be different. Each entry in the table has a pointer to the record of information corresponding to the relation.

RD is the set of such records. Each record contains a pointer to the description of the relation itself. In addition, it will contain the external name and the symbol which indicates whether the description is a table or a description of a synthesis or projection, along with other necessary and convenient information about the relation. The function and the format of RD and ED may vary widely, depending on the application. For example, the record of a relation may contain the list of all elements which appear in the relation, or more extensively, the range of the relation, which may speed up some operations.

RS is the set of descriptions of relations. The relation may be a list of n-tuples or the description of a synthesis or projection. The structure of the table of n-tuples will be considered in Section 5.4. If the description is a synthesis, the mapping ψ has to be given along with the set of relations Ω; if it is a projection, it will be simply a pair consisting of a relation α and a factor L in $P_L(\alpha)$.

5.3 Definition of Operations

Since relations are sets, all set operations are intrinsically applicable here if the relations are of the same rank. Other relation operations are also given. For each operation below the abbreviation of the name may be attached.

1. UNION(R_1,R_2,...,R_k); U(R_1,R_2,...,R_k)

   $\cup_{i=1}^{k} R_i = \{A \mid \exists R_i \ni A \in R_i\}$

2. INDEXED UNION(I); IU(I)

   U(RELATION(I))

3. INTERSECTION(R_1,R_2,...,R_k); I(R_1,R_2,...,R_k)

   $\cap_{i=1}^{k} R_i = \{A \mid A \in R_i \ \forall\ i\}$

4. INDEXED INTERSECTION(I); II(I)

   I(RELATION(I))

5. SYMMETRIC DIFFERENCE(R_1,R_2); SD(R_1,R_2)

   $R_1 \oplus R_2 = \{A \mid (A \in R_1 \vee A \in R_2) \wedge A \notin (R_1 \cap R_2)\}$

6. RELATIVE COMPLEMENT(R_1,R_2); RC(R_1,R_2)

   $R_1 - R_2 = \{A \mid A \in R_1 \wedge A \notin R_2\}$

7. COMPLEMENT(R); C(R)

   $\overline{R} = \{A \mid A \in S^m \wedge A \notin R\}$

8. CARTESIAN PRODUCT(S_1,S_2,...,S_k); CP(S_1,S_2,...,S_k)

   $S_1 x S_2 x \ldots x S_k = \{(a_1,a_2,\ldots,a_k) \mid a_1 \in S_1 \wedge a_2 \in S_2 \wedge \ldots \wedge a_k \in S_k\}$

9. INVERSE(R); INV(R)

   $R^{-1} = \{(a_n,a_{n-1},\ldots,a_1) \mid (a_1,a_2,\ldots,a_n) \in R\}$

10. CARDINALITY(S); CARD(S)

    $|S|$: the number of members in S

11. RANGE(R); RAN(R)

    $P_1(R) \times P_2(R) \times \ldots \times P_n(R)$

12. PROJECTION(L,R); PRO(L,R)

    $P_L(R)$: projection of R by L

13. PERMUTATION(π,R): PERM(π,R)

    $\pi(R)$: permutation of R by π

14. RESTRICTION(L,S,R); REST(L,S,R)

$\{A \mid A \in R \wedge P_L(R) \in S\}$

15. RANK(R)

r(R): rank of R

16. SYNTHESIS(m,L_1:R_1,L_2:R_2,...,L_k:R_k); SYN(m,L_1:R_1,L_2:R_2,...,L_k:R_k)

m-synthesis by ψ: $\psi(L_i) = R_i$, i = 1,2,...,k

17. RELATION(E_1,E_2,...,E_k); R(E_1,E_2,...,E_k)

$\{R_i \mid$ the name of R_i is the same as that of $E_i\}$

18. ELEMENT(R_1,R_2,...,R_k); E(R_1,R_2,...,R_k)

$\{E_i \mid$ the name of E_i is the same as that of $R_i\}$

Also the following predicates are defined.

19. SUBSET(A,B); SUB(A,B)

T if $A \subset B$, F otherwise.

20. EQUAL(A,B); E(A,B)

T if A = B, F otherwise.

21. EQUIVALENT(A,B); EQUIV(A,B)

T if $|A| = |B|$, F otherwise.

22. DISJOINT(A,B); DIS(A,B)

T if $A \cap B = \emptyset$, F otherwise.

23. DECOMPOSABLE(L_1,L_2,R); DEC(L_1,L_2,R)

T if $R = P_{L_1}(R) * P_{L_2}(R)$, F otherwise.

24. REGULAR DECOMPOSABLE(k,R); RDEC(k,R)

T if $R = (R_k)^*$, F otherwise.

5.4 Representation of Relations

In Section 5.2 we did not mention how the relations can be represented internally. Of course any relation of rank m can be intuitively represented as an n by m table where n is the cardinality of the relation. In this section we consider the representation of relations.

First, we will examine the logical, i.e., machine independent, representation of relations. Since unordered sets and ordered sets (sequences) are involved in the description of relations, these two types of sets should be distinguished in some way in their representation. It seems very natural to represent sets by trees and ordered sets by lists since in a set there should not be any precedence relation among members while in a sequence there should be. For example, let us look at the set S = {a,b,c}, (1) of Fig. 5.2. Since each of a, b and c is a member and, there is no

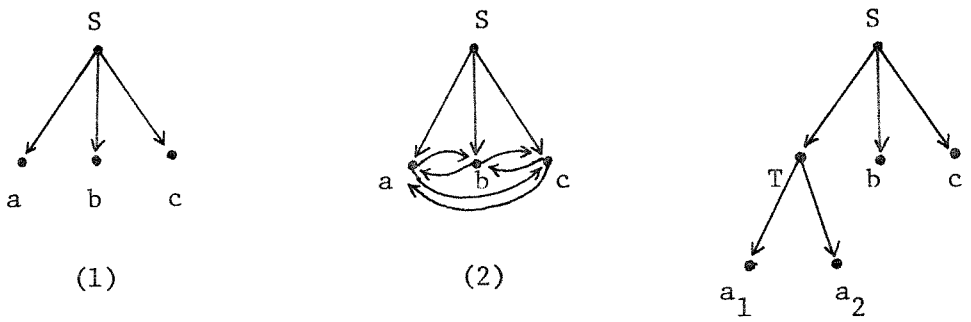


Fig. 5.2

precedence relation among them, each has a direct arc from the node S, which indicates that each is a member of S. One may add arcs between each pair

as shown in (2) of Fig. 5.2. But this would be costly. If a is a set $T = \{a_1, a_2\}$ instead of an element, a is replaced by a tree T as shown in (3) of Fig. 5.2. On the other hand, if S is ordered, i.e. S = (a,b,c), S may be represented by a list as in (1) of Fig. 5.3. Since the set is

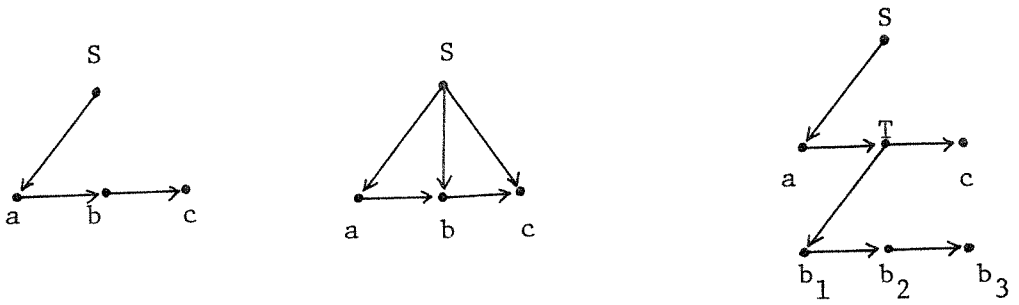


Fig. 5.3

ordered, any element in the set can be specified only by its relative position. Again more costly structures can be considered. Refer to Lowenthal [24] for more variations of the representation of tree structure. If one of the elements is a sequence, say $b = T = (b_1, b_2, b_3)$, then b is replaced by a list T as shown in (3) of Fig. 5.3. In general, we can represent any nested expression of sets and ordered sets this way. For example, W = {a,b,{{c}},(a,{b,d},c),((a,b),c)} will be represented by Fig. 5.4. Note that the number of vertical levels (the number of vertical arcs to the node) corresponds to the depth of the nest which contains the element. Also note that the structure of {a} is identical to that of (a), which is reasonable because the order of an ordered set containing

just one element does not have any sequence.

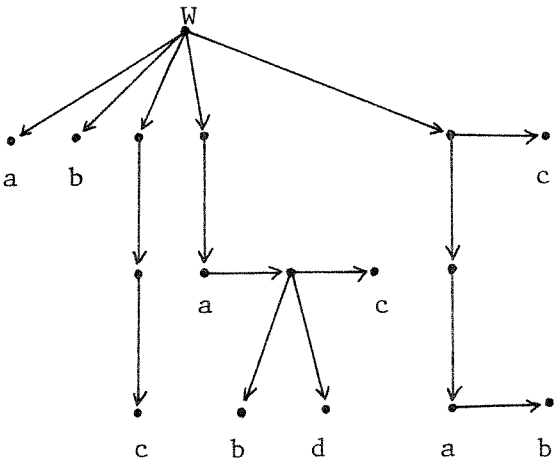


Fig. 5.4

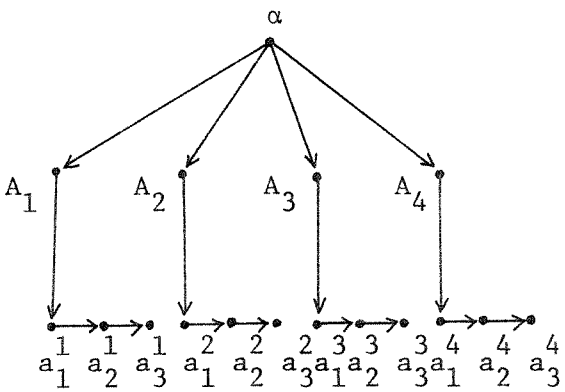


Fig. 5.5

Now back to the representation of relations; a relation is represented by a tree each node of which is replaced by a list. For example, a ternary relation α with four members, $\alpha = \{A_i \mid A_i = (a_1^i, a_2^i, a_3^i),\ i = 1,2,3,4\}$, will be represented as shown in Fig. 5.5. However this does not solve the problem completely since this is a logical structure. We have to find an efficient and small storage, i.e., physical, structure of relations. If we ignore the recurrences of elements in a relation, the minimum storage representation of the relation β seems to be n x m sequential locations each of which keeps the element name, where $m = r(\beta)$ and $n = |\beta|$. However, when we consider the efficiency of the operations, we will see immediately that this is not desirable. For example, such a representation is not flexible for various set operations.

There are two simple structures available for the representation of sets. One is lists and the other is arrays. Let us consider the

representation of ternary relation α. Then we have two different representations for relations, (2) and (3) in Fig. 5.6, by applying arrays to sets and lists to sequences in case of (2), and lists to sets and arrays to sequences in case of (3). However, when we notice that a relation has the same rank throughout the life of the relation while the set of n-tuples is time-variant, i.e., may vary according to changes in the set of elements and in the outside world, (3) seems to be more flexible than (2). The structure (4) in Fig. 5.6 obtained by applying lists to both sets and

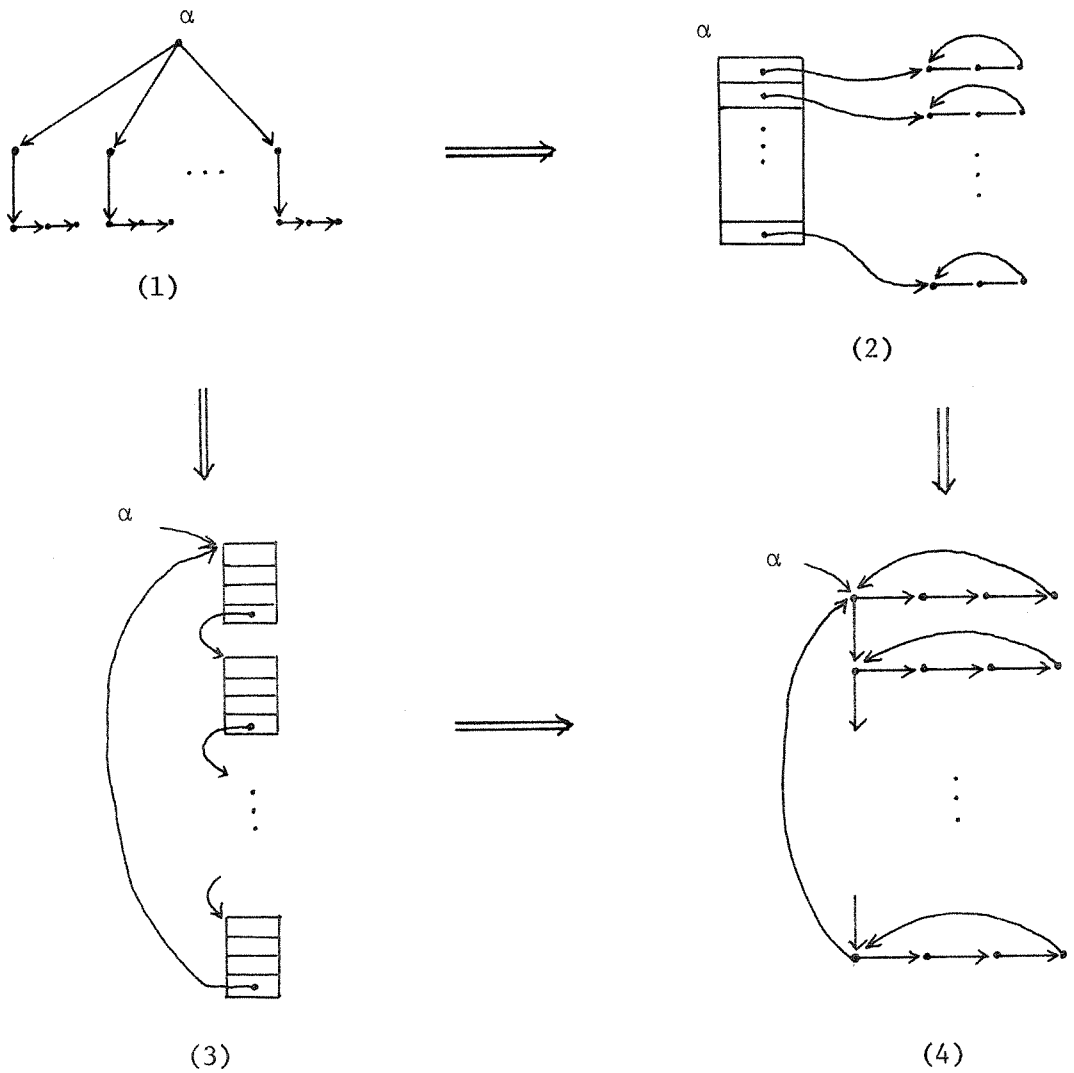


Fig. 5.6

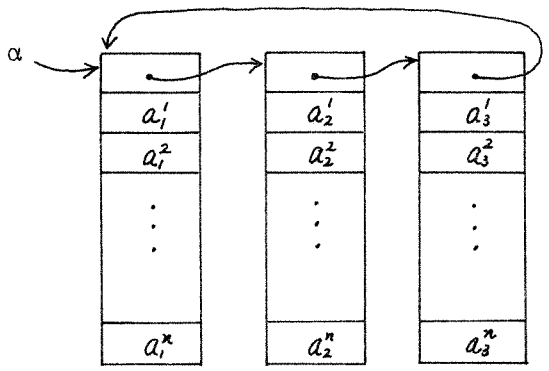


Fig. 5.7

sequences is essentially the same as (3) but may require more space. When we consider permutation and decomposition operations, the structure in Fig. 5.7 seems very attractive. But it cannot be efficient for overall operations.

5.5 Some Applications

The relational data structure described here is general enough to handle the structures of sets, relations, and tables. Therefore, one of the most significant features of the data structure is that all these structures can be used simultaneously and in a uniform manner. Some potential fields for the application include data management systems, pattern recognition, semantic information processing, graph theory, and network analysis. In the sequel we will give some examples.

It should be noted that the design of an efficient data base is the responsibility of the user. All relations have to be defined in advance if they are referred to without specification. But the point is that one can choose which relations should have real data and which should be only described by other relations in order to optimize the speed and the space since some relations are obtained by projection, synthesis and other relational operations and since some relations are used frequently and others are rarely used.

Example 5.1 Let us consider family lineage. Given a population of people, unary relations specify special classes in the population. For example, the sets of males, females, children, boys, girls, those who are older than 40, etc., are given by the corresponding unary relations. Then we have various binary relations, e.g., father, mother, brother, sister, son, daughter, aunt, senior by more than 10 years. Also we have higher rank relations. For example, parent, grandfather-father-son and three brothers are ternary relations.

In the case of family lineage, most high rank relations are binary decomposable. As a matter of fact, three binary relations, "husband-wife", "brothers or sisters" and "parent-child" relations, along with appropriate unary relations may be enough to generate any other relations of family lineage. Therefore, to make the example more interesting, we add the two tables described below.

Let the data of the following relations be actually given:

Unary relations:

S: a population of people

M: the set of males

F: the set of females

X: any subset of S

Binary relations:

$\alpha = \{(x,y) \mid x \text{ is the husband of } y\}$

$\beta = \{(x,y) \mid x \text{ is a brother or a sister of } y\}$

$\gamma = \{(x,y) \mid x \text{ is a parent of } y\}$

Ternary relation:

$\delta = \{(a,b,c) \mid a \text{ is the household head of } b \text{ and } c \text{ is the age of } b \ \forall \ b \in S\}$

4-ary relation:

$\lambda = \{(a,b,c,d) \mid$ b is the blood type, c is the hair color, and d is the eye color of a $\forall a \in S\}$

Then the following are some examples which show how one may obtain specific relations including sets by using the operations of the relational data structure defined earlier.

1. the set of ages A

   $A = PRO(3,\delta)$

2. the set of blood types B

   $B = PRO(2,\lambda)$

3. the set of hair colors HC

   $HC = PRO(3,\lambda)$

4. the set of eye colors EC

   $EC = PRO(4,\lambda)$

5. the relation ω "is the wife of"

   $\omega = INV(\alpha)$

6. the relation μ "is the mother of"

   $\mu = REST(1,F,\gamma)$

7. the relation $\gamma\mu$ "is the grandmother of"

   $\gamma\mu = PRO((1,3),SYN(3,(1,2):\mu,(2,3):\gamma))$

8. the set of father FA of the people in X

   $FA = I(M,PRO(1,REST(2,X,\gamma)))$

9. the number p of mother-child pairs whose blood types are the same

   $p = CARD(SYN(3,(1,20:\mu,(1,3):PRO((1,2),\lambda),(2,3):PRO((1,2),\lambda)))$

10. the relation κ "is a cousin of"

    $\kappa = PRO((1,4),SYN(4,(1,2):INV(\gamma),(2,3):\beta,(3,4):\gamma))$

11. the set N of people with no brother or sister

$$N = C(PRO(1,\beta))$$

12. the number q of households whose head is not related to any other member by blood or marriage

$$q = CARD(PRO(1,REST((1,2),U(\beta,\gamma,INV(\gamma),\alpha,INV(\alpha),PRO((1,3),SYN(3,(1,2):\gamma,(2,3):\gamma))),\delta)))$$

13. the number r of households whose head is of an age in a subset of A' of A

$$r = CARD(PRO(1,REST(3,A',\delta)))$$

14. the set of people T whose blood type is b ε B, whose hair color is h ε HC and whose eye color is e ε EC.

$$T = PRO(1,REST(2,b,REST(3,h,REST(4,e,\lambda))))$$ ///

Example 5.2 Let G = (S,Γ) be a graph, then the sets of various subgraphs can be expressed as high rank relations. And they are binary decomposable. Therefore one needs only the binary "connection" relation Γ.

Unary relation:

S: the set of nodes

Binary relation:

$$\alpha = \{ (a,b) \mid (a,b) \varepsilon \Gamma \}$$

1. the set of complete subgraphs K_4 of order 4

$$K_4 = SYN(4,(1,2):\alpha,(1,3):\alpha,(1,4):\alpha,(2,3):\alpha,(2,4):\alpha,(3,4):\alpha)$$

2. the set of simple cycles C_4 of order 4

$$C_4 = SYN(4,(1,2):\alpha,(2,3):\alpha,(3,4):\alpha,(1,4):\alpha,(1,3):C(\alpha),(2,4):C(\alpha))$$

3. the set of nodes which are directly connected from X, R

$$R = PRO(2,(REST(1,X,\alpha)))$$

4. the set of all paths of length 3 from a node x to a node y, P_{xy}

$$P_{xy} = SYN(4,(1,2):REST(1,\{x\},\alpha),(2,3):\alpha,(3,4):REST(2,\{y\},\alpha))$$ ///

CHAPTER VI

SOME APPLICATIONS OF RELATIONAL SYSTEMS

6.1 Introduction

In this chapter two different applications of relational systems and, hence, the use of decompositions and syntheses of relations, namely pattern recognition and graph theory, are presented. First, we formalize pattern classification by using the concept of relational system and apply it to the problem of synthesis of patterns. Secondly, an algorithm to find all cliques by synthesizing complete subgraphs from smaller complete subgraphs is described.

6.2 Pattern Recognition

6.2.1 Relational pattern description

In this section an attempt is made to apply relational systems to pattern description. Although the word "pattern" is used, it should not be limited to patterns in a narrow sense, i.e. pictures, but rather a pattern should be interpreted as any system as mentioned in Chapter 1. The importance of pattern descriptions cannot be over-emphasized because they are critical for both pattern classification and pattern analysis[11,34].

Definition: A pattern P is a pair $P = (S, \{R^i\}_{i\in I})$ where S is the set of primitives of the pattern and each R^i is a relation of rank n_i on S, i.e. $R^i \subset S^{n_i}$.

Example 6.1 Let us consider the pattern shown in Fig. 6.1. The pattern may be expressed by $P = (S, \{R^1,R^2,R^3,R^4,R^5\})$ where $S = \{a,b,c\}$ and

R^1 = "square" = {(a),(c)},

R^2 = "left of" = $\left\{\begin{matrix}(a,b)\\(b,c)\\(a,c)\end{matrix}\right\}$,

R^3 = "rectangle" = {(b)},

R^4 = "right of" = $\left\{\begin{matrix}(b,a)\\(c,b)\\(c,a)\end{matrix}\right\}$,

R^5 = "between" = {(a,b,c)}. ///

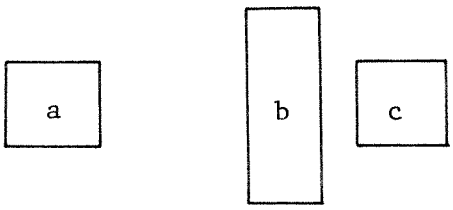


Fig. 6.1

It should be remarked that all primitives are already determined and the relations can be detected. Also it should be noted that these primitives and relations depend primarily on one's concern and the possibility of identifying the primitives and the relations. Although methods are being developed in this direction[14,20,23], we are not concerned with this topic here.

Some of the terms defined earlier which are used here are: The projection or partial relation corresponding to a factor L of a relation α is the relation α_L whose domain is restricted to the domain L of S^n. A permutation of a relation α corresponding to a permutation π is a relation $\pi(\alpha) = \{(a_{\pi(1)},a_{\pi(2)},\ldots,a_{\pi(n)}) \mid (a_1,a_2,\ldots,a_n) \in \alpha\}$. An m-synthesis of a set of relations is an m-ary relation each of whose binary factors is covered by some relation of the set or a trivial relation.

Definition: A subset of relations $\{R_k\}_{k\in K}$ of a set of relations $\{R_i\}_{i\in I}$, i.e. $K \subset I$, is a generating set of $\{R_i\}_{i\in I}$ if every relation R_i is obtained from $\{R_k\}_{k\in K}$ by applying the operations of projection, permutation and synthesis to them. It is said to be minimal if no subset of it is a generating set.

The motivation of the definition of generating sets is clear. A generating set is sufficient to provide all information about the pattern which can be obtained from the original set of relations. We write $\{\alpha_i\}_{i\in I} \overset{*}{\Rightarrow} \{\beta_j\}_{j\in J}$ to denote that $\{\alpha_i\}_{i\in I}$ is a generating set of $\{\beta_j\}_{j\in J}$.

Definition: Two elements a and b in S are ambiguous if the interchange of a and b in all relations of the pattern leaves the relations unchanged. They are said to be unambiguous otherwise. An element in S is said to be unique if the element is unambiguous with any other element in S.

What does ambiguity mean? It means that even though two ambiguous elements have different names, they are identical characteristically in the pattern, i.e. with respect to the relations defined in the pattern. Therefore, for example, if one is retrieved by some condition expressed by using the relations and other elements, then the other is also retrieved. Does decomposition help disambiguate ambiguous elements? No. Actually we have the following.

Theorem 6.1 Two elements in S are ambiguous iff they are ambiguous with respect to a generating set of the pattern.

The proof is trivial. Also it is easy to see that syntheses do not help disambiguate if they are loss-free.

Definition: A pattern is unambiguous if any two elements of S are unambiguous.

Example 6.2 The pattern is rectangular as shown in Fig. 6.2. The primitives are edges, a, b, c and d. Let us choose the following four non-isomorphic relations, two binary relations α and β, one ternary relation γ, and one 4-ary

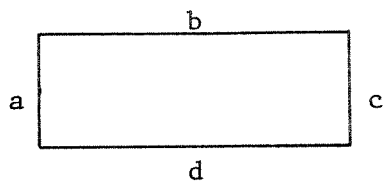


Fig. 6.2

relation δ. And let us assume that these are all essential as far as our present interest is concerned.

$$\alpha = \begin{Bmatrix} (a,b), (b,a) \\ (b,c), (c,b) \\ (c,d), (d,c) \\ (a,d), (d,a) \end{Bmatrix} \qquad \beta = \begin{Bmatrix} (a,c), (c,a) \\ (b,d), (d,b) \end{Bmatrix}$$

$$\gamma = \begin{Bmatrix} (a,b,c), (c,b,a) \\ (b,c,d), (d,c,b) \\ (c,d,a), (a,d,c) \\ (d,a,b), (b,a,d) \end{Bmatrix} \qquad \delta = \begin{Bmatrix} (a,b,c,d), (a,d,c,b) \\ (d,a,b,c), (d,c,b,a) \\ (c,d,a,b), (c,b,a,d) \\ (b,c,d,a), (b,a,d,c) \end{Bmatrix}$$

Here α may be the relation "adjacent to", "connected" or "perpendicular", β the relation "indirectly connected", or "parallel", γ the relation "between", and δ the relation "circuit", in the physical and structural meaning.

Now the pattern P in Fig. 6.2 is represented by $P = (S, \{R_i\})$ where $S = \{a,b,c,d\}$ and $\{R_i\} = \{\alpha,\beta,\gamma,\delta\}$. Then the followings are the simple consequences of the definitions above.

(1) $\alpha \neq \beta$.

(2) $\gamma_{13} = \beta$.

(3) $\gamma_{12} = \gamma_{23} = \alpha$.

(4) $\delta_{ijk} = \gamma$, where i, j and k are distinct.

(5) $\delta_{13} = \delta_{24} = \beta$.

(6) $\delta_{14} = \delta_{41} = \delta_{ij} = \alpha$ where $i,j = 1,2,3,4$ and $i-j = 1$.

(7) $\xi = \{\xi_{12} = \alpha \wedge \xi_{13} = \beta \wedge \xi_{14} = \alpha \wedge \xi_{23} = \alpha \wedge \xi_{24} = \beta \wedge \xi_{34} = \alpha\} = \delta$.

(8) $\zeta = \{\zeta_{12} = \zeta_{23} = \alpha \wedge \zeta_{13} = \beta\} = \gamma$.

Therefore $\{\alpha,\beta\} \overset{*}{\Rightarrow} \{\alpha,\beta,\gamma,\delta\}$, $\{\gamma\} \overset{*}{\Rightarrow} \{\alpha,\beta,\gamma,\delta\}$, and $\{\delta\} \overset{*}{\Rightarrow} \{\alpha,\beta,\gamma,\delta\}$. In other words, each of $\{\alpha,\beta\}$, $\{\gamma\}$ and $\{\delta\}$ is a minimal generating set of $\{\alpha,\beta,\gamma,\delta\}$.

Finally note that a and c, and b and d are ambiguous because they

are interchangeable as also seen in the figure. Therefore none of them are unique. ///

6.2.2 Relational pattern classification

Next we consider pattern description in light of pattern classification. Suppose that we have a set of patterns $\{P_j\}_{j\in J} = \{(S_j, \{R_{ji}\})\}$ such that all S_j's are the same and each R_{ji} has the same structural meaning for all j. In other words, for any different j_1 and j_2, $R_{j_1 i}$, $R_{j_2 i} \subset S^{n_i}$ have the same structural meaning although it may be that $R_{j_1 i} \neq R_{j_2 i}$. It may be thought that the relation R_{ji} varies not from time to time but from pattern to pattern in the set. In this sense we denote the relation which represents the R_{ji}'s simply by R_i.

Definition: R_i is a discriminant relation of P_1 and P_2 if R_i distinguishes P_1 from P_2, i.e. if $R_{1i} \neq R_{2i}$.

Definition: A set of relations $\{R_i\}_{i\in I}$ is a discriminant relation set of $\{P_j\}_{j\in J}$ if $\{R_i\}_{i\in I}$ distinguishes each pattern from others, i.e. if there exists a discriminant relation in $\{R_i\}_{i\in I}$ for each pair of patterns in $\{P_j\}_{j\in J}$. A discriminant relation set is minimal if any proper subset of the set cannot be a discriminant relation set.

Definition: $\{R_i\}_{i\in I}$ is a characteristic relation set of P_j if $\{R_i\}_{i\in I}$ distinguishes P_j from all other patterns in $\{P_j\}_{j\in J}$. A minimal characteristic relation set is also defined similarly.

Therefore a discriminant relation set is always a characteristic relation set for any pattern while a characteristic relation set of any pattern is not necessarily a discriminant relation set of the set of patterns.

Definition: The active primitive set S^a of a pattern $P = (S, \{R_i\}_{i\in I})$

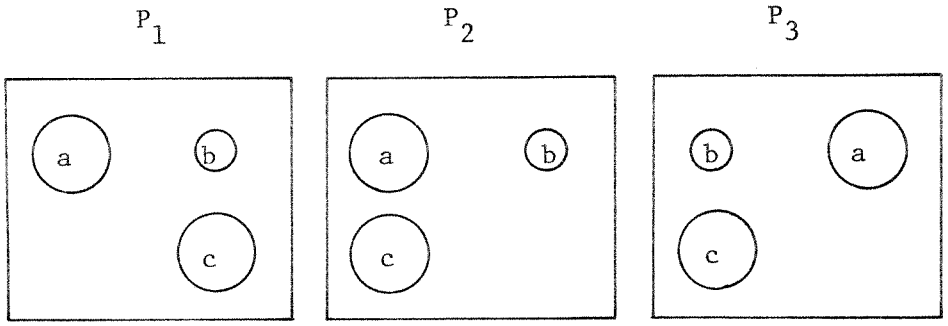
is a set of elements $S^a = \{x \mid \exists R_i \rightarrow x \text{ is an element of } R_i\}$.

Since we are talking of patterns which are supposed to be distinct, it is quite conceivable that the active primitive sets S_j^a's of the patterns P_j's are also different. Noting this point, the procedure to distinguish two patterns P_1 and P_2 may go as follows.

1. $|S_1^a| = |S_2^a|$? If not, they are different.
2. $S_1^a = S_2^a$? If not, they are different.
3. Try to find a relation R_i such that $R_{1i} \neq R_{2i}$. If there is one, they are different.
4. Take a new S and repeat 1 to 3. Or it is decided that they are the same.

An example is given to illustrate the terms defined above.

Example 6.4 Suppose that S = {a,b,c,d} although d does not appear in the patterns P_1, P_2 and P_3 as shown in Fig. 6.3. Then the active primitive



P_1	P_2	P_3
$\alpha = \{(a,b)\}$	$\alpha = \{(a,b)\}$	$\alpha = \{(b,a)\}$
$\beta = \{(a,b),(c,b)\}$	$\beta = \{(a,b),(c,b)\}$	$\beta = \{(a,b),(c,b)\}$
$\gamma = \{(b,c)\}$	$\gamma = \{(a,c)\}$	$\gamma = \{(b,c)\}$

Fig. 6.3

set is $S_i^a = \{a,b,c\}$ for i = 1,2,3. Suppose further that we have chosen three

binary relations on S, α = "to the left of", β = "larger than", and γ = "above". The abstract relations corresponding to these physical relations for the patterns are also given under each pattern in Fig. 6.3.

Note that the relation β is not useful at all in classifying the patterns P_1, P_2 and P_3. α is a discriminant relation of P_1 and P_3, and of P_2 and P_3. γ is a discriminant relation of P_1 and P_2, and P_2 and P_3. Neither α nor γ alone can be a discriminant relation set of $\{P_1,P_2,P_3\}$ although they are the characteristic relation set of P_3 and P_2, respectively. Therefore $\{\alpha,\gamma\}$ is the only minimal discriminant relation set of $\{P_1, P_2, P_3\}$. ///

6.2.3 Synthesis of patterns

In this section a special type of pattern synthesis problem is considered: Given a discriminant set, we construct patterns which satisfy the discriminant set condition as well as other necessary relations. Since, specifically, we are going to synthesize characters, the topic of character recognition is briefly described.

The character recognition may be conveniently categorized as follows [17]:

1) recognition of special stylized fonts,

2) recognition of machine imprinted fonts,

3) recognition of hand-written characters.

Even though during the last several years the developments in 2) and 3) have been so remarkable that the rate of erroneous recognition has reached 1/2 % in some systems which is enough for practical use, still many users prefer to adopt the approach 1) because of the efficiency and the cost of the systems. Today several typical stylized fonts are used in the U. S. These fonts were designed by various ad hoc techniques and experiences. It

is very conceivable that the designers used all sorts of ad hoc techniques to design the characters which are easily discriminated from each other. One important thing, however, in the design of sylized fonts is that they must be easily recognized by human beings too. Here we will describe a systematic method for designing stylized fonts by using the set of numerals, 1, 2,..., 9, 0, as an example.

The ordinary figures of these numerals are given in Fig. 6.4. First we have to determine primitives. Each primitive should be easily recognizable. Furthermore in general the fewer the number of primitives, the better. Another condition is that any two characters must be distinguishable by the primitives and the relations on primitives. By checking the numerals one by one we can find that at least a circle, a line and a half circle have to be in the set of primitives. We can distinguish numerals using only unary relations, i.e. these three primitives, as described in Table 6.1.

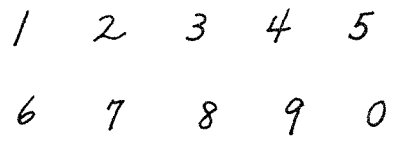


Fig. 6.4

circle	0, 6, 8, 9
line	1, 4, 6, 7, 9
half circle	2, 3, 5

Table 6.1

Circle	two circles	8
	one circle	0
Line	circle + line	6, 9
	line	1
	line + something	4, 7
Half Circle	one half circle + something	2, 5
	two half circles	3

Table 6.2

If we classify numerals using these three primitives, we obtain the result in Table 6.2. This means that we have to distinguish only three pairs: (6,9), (4,7) and (2,5). This can be easily accomplished by either adding more primitives or by using relations on the primitives we already have. At this point we should recall that the designed numerals have to be easily recognizable by human beings.

Let us look at each pair in order. First, 6 and 9 can be distinguished by the relation between the circle and the line. In other words, in "6" the line is onto the circle while the circle is onto the line in "9". It may be advantageous to think that 6 is obtained by rotating 9 by 180^{o}. Next, for 4 and 7 we feel that two more primitives, i.e. $\ulcorner$ and $\llcorner$, are necessary. Finally, these two additional primitives can also be used for 2 and 5.

The figures of numerals designed this way are given in Fig. 6.5. One may be surprised to find how few primitives and combinations actually produce numerals.

Fig. 6.5

6.3 Complete Subgraphs

6.3.1 Synthesis of complete subgraphs

The concept of complete subgraph in graph theory is very fundamental, yet not much study of them has been done. On the other hand, the covering of edges by edges has not been studied although the covering of nodes by edges or edges by nodes of graphs has been much investigated[2,15,29].

Since relations are our subject in this study, and not only edges which are binary relations but also complete subgraphs can be thought of as well-behaving relations, we will study complete subgraphs with respect to the covering of edges by edges. In other words, let $G = (X, \phi)$ be a graph where X is a set of nodes and ϕ is a binary relation on X, i.e. $\phi \subset X^2$. Since we consider graphs without cycles, $\phi = \phi^{-1}$ and $(x,x) \notin \phi \; \forall x \in X$. As usual, (x,y) and (y,x) are denoted by one edge (undirected arc) in graphs. Then the relational system G can be conveniently represented by $(X, \{\phi_i\})$ where each ϕ_i represents the relation of complete subgraphs of order i, assuming that what we are interested in about this graph is complete subgraphs. Therefore $\phi_1 \subset X$, $\phi_2 \subset X^2, \ldots, \phi_n \subset X^n$ where $|X| = n$. Let $[\phi_i]$ represent the set of elements x such that $\exists (a_1,a_2,\ldots,a_i) \in \phi_i$ and $x = a_j$ for some $j = 1,2,\ldots,i$. Then it follows immediately from the definitions that $X \subset \bigcup_{i=1}^{n} [\phi_i]$, therefore,

$X = \cup_{i=1}^{n}[\phi_i]$ and $[\phi_1] \supset [\phi_2] \supset \ldots \supset [\phi_n]$.

We showed earlier that complete subgraphs of order 3 as an example of the triangle relation is binary decomposable. Here we generalize the claim. The proof is straightforward and omitted.

Lemma 6.1 The relation of complete subgraphs of order k is (k-1)-ary decomposable.

In this section we are considering an application of synthesis of relations. It was stated earlier that complete subgraphs of various orders can be thought of as a set of relations on the graph. And complete subgraphs of order 3 can be obtained from complete subgraphs of order 2, and the complete subgraphs of order 4 can be obtained from the complete subgraphs of order 3, and so on although the converse is not always possible. In other words, we generate all complete subgraphs by repeatedly synthesizing complete subgraphs of order k from the complete subgraphs of order k-1. Before we give the complete algorithm for this, the presentation of the following fact which will be used in the algorithm is due.

Theorem 6.2 Let α be a relation of the complete subgraphs of order k ($k \geq 3$), then

$$\alpha = {}^{\alpha}_{L_1} * {}^{\alpha}_{L_2} * {}^{\alpha}_{L_3}$$

where L_1, L_2 and L_3 are any three different (k-1)-ary partial relations of α.

Proof Since $A = (a_1, a_2, \ldots, a_k)$ is a complete subgraph of order k if $(a_i, a_j) \in \phi \ \forall a_i \neq a_j$, $i,j = 1,2,\ldots,k$, we only have to make sure that a_i and a_j are connected $\forall a_i \neq a_j$. In other words, a_i and a_j are both contained in some complete subgraph of order k-1 $\forall a_i, a_j$. Suppose that one (k-1)-ary partial relation α_{L_1} covers (2,3,...,k)-factor and another α_{L_2} covers (1,3,4,...,k), without loss of generality, since $\alpha_{L_1} = \alpha_{L_2}$ otherwise. Then

just one binary factor (1,2) has been covered by neither α_{L_1} nor α_{L_2}. But it is then obvious that any other (k-1)-ary partial relation α_{L_3} different from α_{L_1} and α_{L_2} can cover the factor.

Q.E.D.

This theorem provides a drastic reduction of time for synthesis of complete subgraphs. We can use this theorem for those syntheses in which only (k-1)-ary factors are essential and all other factors are non-essential. This is an example of type 2 synthesis. Of course it is not true for general relations. This specialty comes from the property of strong symmetry of the relations. The following example shows that the theorem is not true in general.

Example 6.5 Let $\alpha = \{(1,1,2,2,2),(2,1,1,2,2),(2,2,2,2,2)\}$ be a 5-ary relation on $S = \{1,2\}$ as shown in Fig. 6.6, then α is clearly 4-ary decomposable. Since

$$\alpha_{2345} = \begin{Bmatrix}(1,2,2,2)\\(1,1,2,2)\\(2,2,2,2)\end{Bmatrix}$$

$$\alpha_{1345} = \begin{Bmatrix}(1,2,2,2)\\(2,1,2,2)\\(2,2,2,2)\end{Bmatrix}$$

$$\alpha_{1245} = \begin{Bmatrix}(1,1,2,2)\\(2,1,2,2)\\(2,2,2,2)\end{Bmatrix},$$

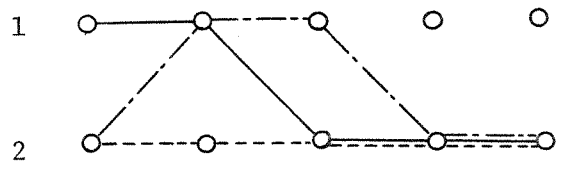


Fig. 6.6

$(2,1,2,2,2) \in \alpha_{2345} * \alpha_{1345} * \alpha_{1245}$ although $(2,1,2,2,2) \notin \alpha$. ///

6.3.2 Description of algorithm

In the sequel, to eliminate redundancy in the complete subgraphs generated, each relation of the complete subgraphs of order k contains only

those members $(a_1,a_2,\ldots,a_k)$ such that $\{a_1,a_2,\ldots,a_k\}$ is a complete subgraph of order k and $a_1 < a_2 < \ldots < a_k$ by the predetermined ordering, i.e. $\{A = (a_1,a_2,\ldots,a_k) \mid A \in \phi_k$ and $a_1 < a_2 < \ldots < a_k\}$.

Now an algorithm to find all complete subgraphs of a given graph is described. The following symbols are used in the algorihtm.

C^i: the ordered list of all complete subgraphs of order i.

C^i_j: the j-th complete subgraph of order i in the list C^i.

L: the initial list of edges.

Limit(i): the number of members in the list C^i, i.e. the number of different complete subgraphs of order i.

The macro structure of the algorithm is given by the flowchart shown in Fig. 6.7.

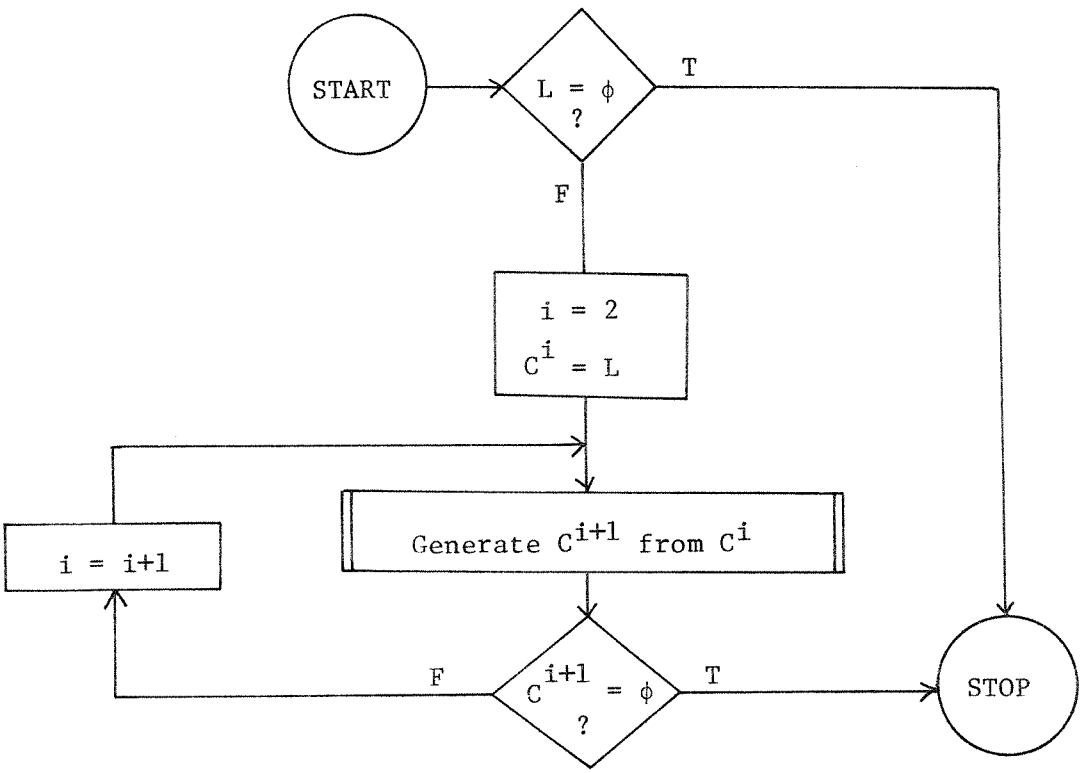


Fig. 6.7

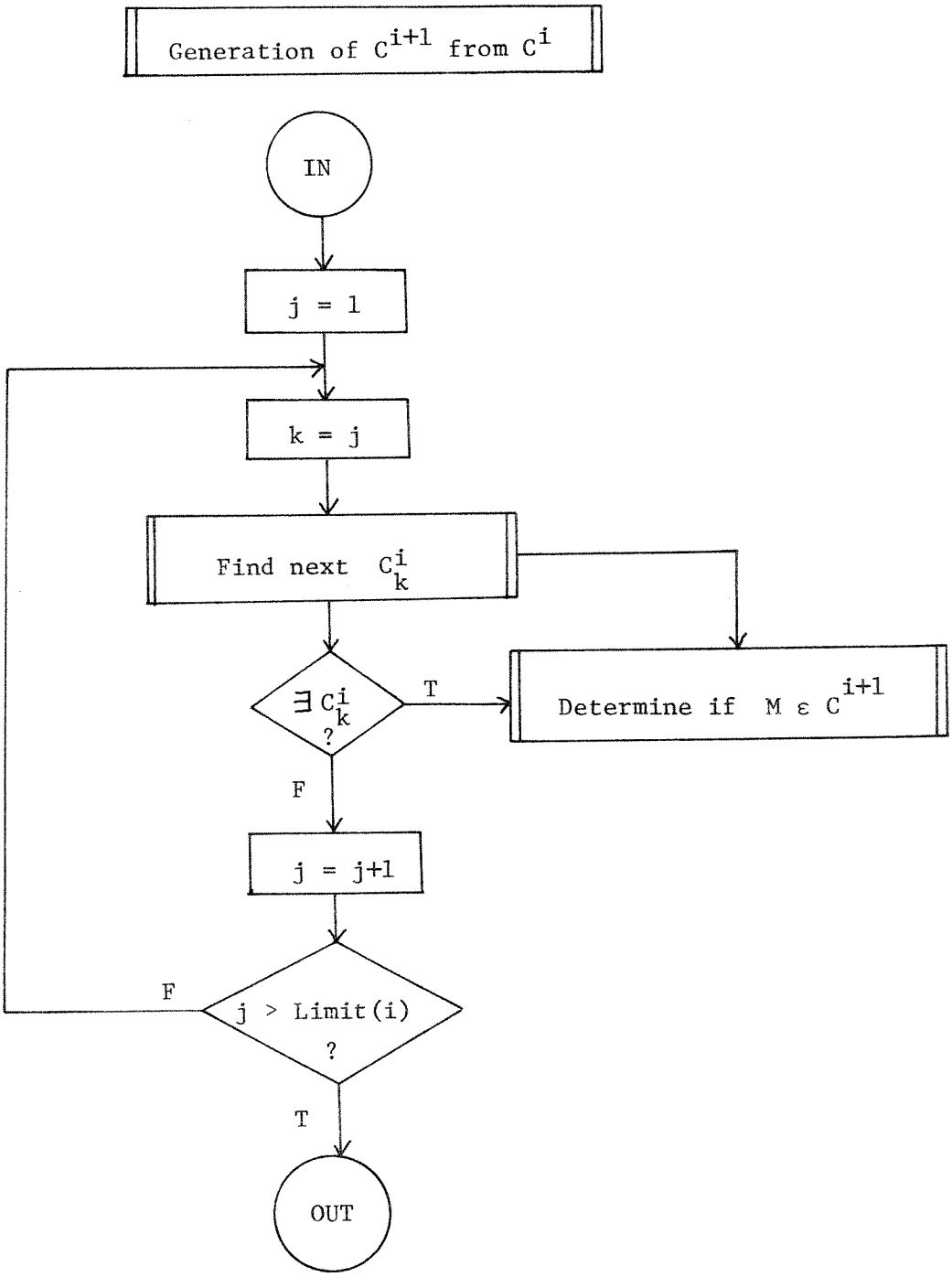


Fig. 6.8

How do we generate C^{i+1} from C^i? According to Theorem 6.2, C^{i+1} can be generated from the synthesis of C^i's which may be any three i-ary

factors of the (i+1)-ary relation C^{i+1}. Therefore we only have to generate a natural join of C^i, C^i and C^i, i.e. $C^i * C^i * C^i$. The number of the operations will not grow as fast as order $\{\text{Limit (i)}\}^3$ because we will exploit the property that $a_1 < a_2 < \ldots < a_i$ for any $(a_1,a_2,\ldots,a_i) \varepsilon C^i$. The flowchart of an algorithm for this operation is given in Fig. 6.8. For each C^i_j we look for C^i_k such that the last i-1 elements of C^i_j are exactly the same as the

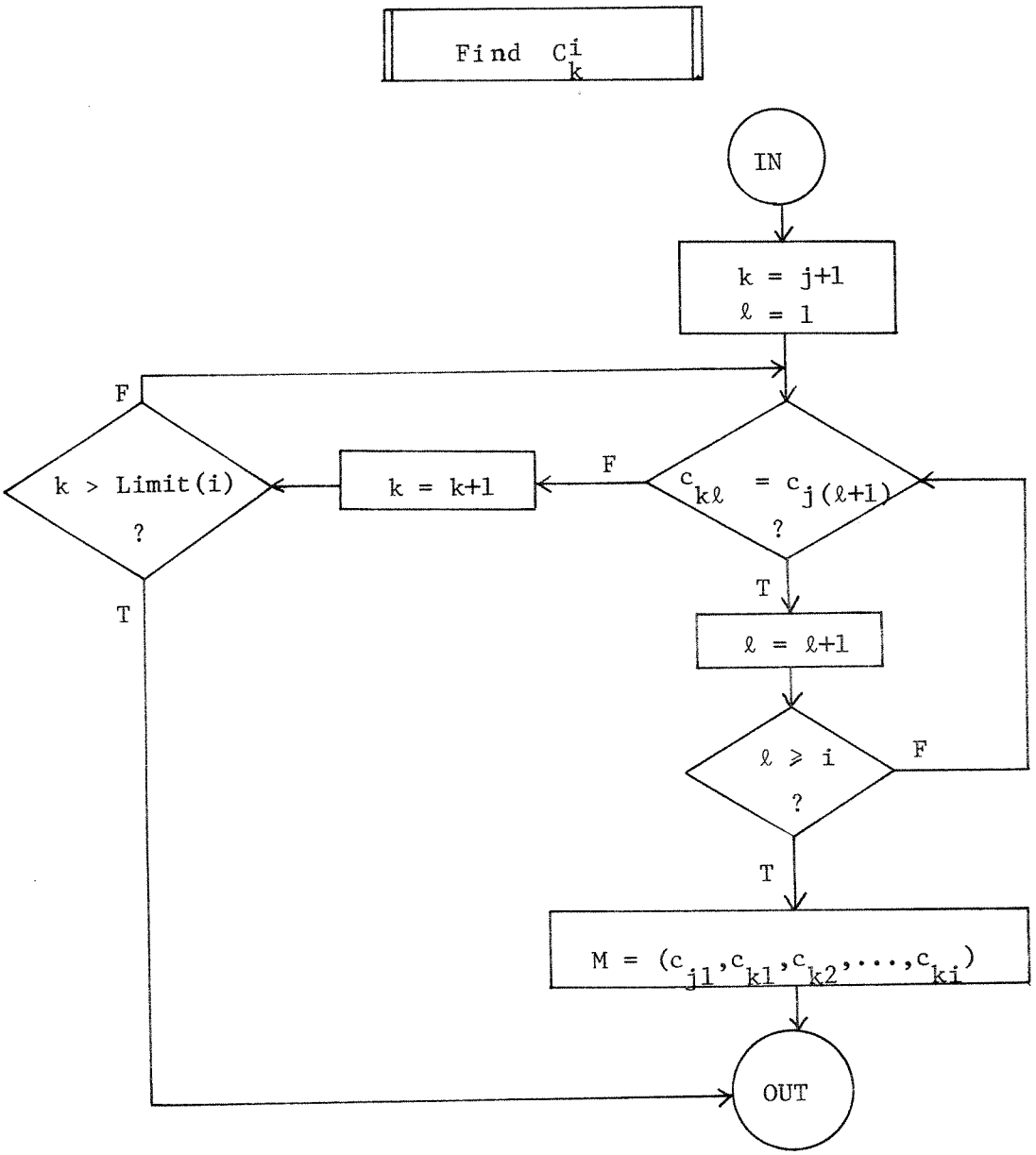


Fig. 6.9

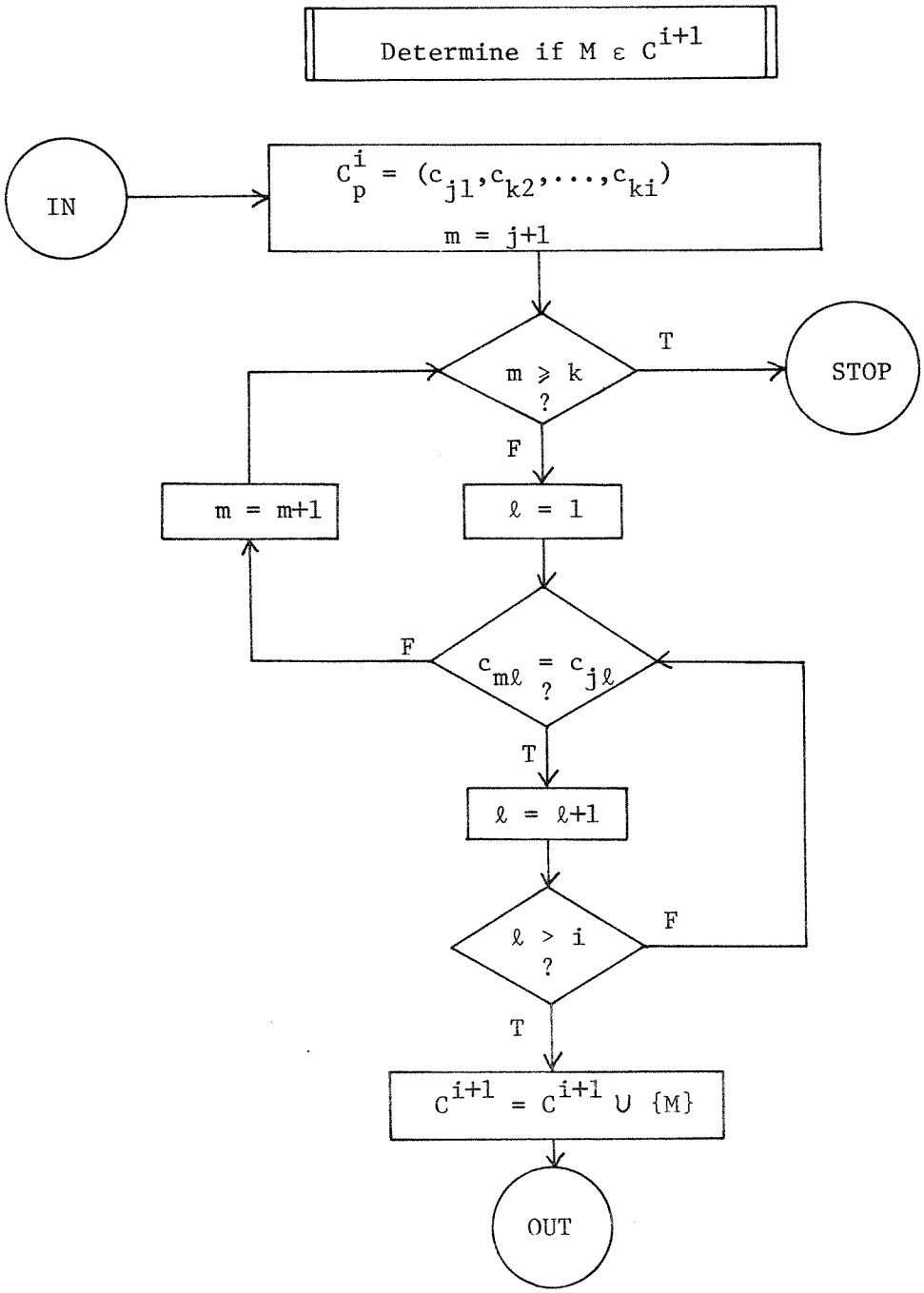


Fig. 6.10

first i-1 elements of C_k^i. For the (i-1)-composition of C_j^i and C_k^i, denoted by $M = (c_{j1}, c_{k1}, c_{k2}, \ldots, c_{ki})$, to be a member of C^{i+1}, there must exist C_p^i in C^i such that C_p^i is the same as some i-factor of M different from C_j^i and C_k^i.

Then clearly C_p^i is located between C_j^i and C_k^i since C^i is ordered and $c_{p1} = c_{j1}$ and $c_{p2} = c_{k2}$. Therefore only a partial portion of C^i has to be searched for C_k^i and C_p^i.

Fig. 6.8 gives the flowchart of an algorithm to generate C^{i+1} from C^i, which was explained above. For each table C^i we need, in order to accomplish the goal, three pointers, one for j, one for k and one for p. Among them the following relations always hold: $j < k$, and $j < p < k$. The flowchart shown in Fig. 6.9 gives an algorithm to find C_k^i's.

Also an algorithm to determine whether M is a member of C^{i+1} is shown in Fig. 6.10. This completes the algorithm that generates all complete subgraphs of a given graph.

6.3.3 Discussion

1. It is easy to show that the algorithm just described terminates for all finite graphs. Let the number of nodes in the graph be n, then clearly $C^{n+1} = \phi$ in any case. Hence the algorithm ends up with STOP in Fig. 6.7. Also we can show that each subroutine does not have an infinite loop.

2. If M is determined to be in C^{i+1}, then $M \notin C^{i+1}$ until it is added by the operation $C^{i+1} = C^{i+1} \cup \{M\}$, that is to say, each C_j^{i+1} is generated only once by the algorithm.

3. The algorithm can be easily modified so that single nodes are also regarded as complete subgraphs of order 1.

4. Since the purpose of the algorithm is to generate all complete subgraphs, it is not difficult to generate all maximal complete subgraphs, i. e. cliques, by modifying the algorithm slightly.

Two algorithms to find all cliques are found in the recent literature [1,3,27]. There is a fundamental difference between the algorithm given

here and those in the references. Namely, the former is edge-oriented while the latter are node-oriented. In the latter case each node is tried to generate all candidates for cliques of order 2, and then each of these candidates is tried to generate all candidates for cliques of order 3, and so on. The level of the algorithm given here is the same as that of the algorithm of Bierstone [27] in the sense that both are exhaustive methods. On the other hand, the algorithm by Bron et al [3] tries to generate all cliques without generating all complete subgraphs, if possible, by using the branch and bound technique. Of course we do not claim that our approach is always better than the others. However, it is obviously advantageous to use this approach when the number of edges is relatively small. Furthermore, this approach will become definitely attractive if associative memories are available.

CHAPTER VII

CONCLUSION

7.1 Summary

In this study we first mention the areas which use relations such as pattern recognition, semantic information processing, and data management systems, and point out the importance of the uses of relations in such areas. We can abstract these systems using the model of relational systems. A relational system is a pair $(S, \{\alpha_i\}_{i\in I})$ where S is the set of objects in the system and $\{\alpha_i\}_{i\in I}$ is the set of all relations on S which specify the characteristics of the system. Since we are concerned with these systems with respect to information processing, the internal representation of rational systems and its efficient manipulation is quite important. Therefore our goal is the development of a data structure suitable for relational systems which is here called the relational data structure.

First of all, in order to deal with relations we find that little is known about higher rank relations. Therefore we start this study by analyzing higher rank relations and by giving definitions which are used later. We pursue the analysis of higher rank relations by analogy with binary relations. That is, in Chapter 2 we try to regard binary relations as a special case of high rank relations so that the only difference between binary relations and higher rank relations is the rank. Along with the extension of various terms for binary relations to higher rank relations, we present some possible extensions of the properties of reflexivity, symmetry and transitivity. And then using these extended properties compatibility and equivalence relations are extended to higher rank relations.

Then we move to the decomposition of relations. Our original motivation to study the decomposition of relations is the possibility that higher rank relations can be decomposed into lower rank relations, and hence, eventually, to binary relations. This is desirable since lower rank relations are generally simpler, more intuitive, and easier to handle, and we know much more about binary relations. In Chapter 3 we define the most general and natural form of decomposition. Even though the possibility turns out to be false, in general, it is still desirable to reduce the ranks of relations so that we may have simpler relations and we can keep only a set of common lower rank relations for a given set of relations. After mentioning some aspects of decompositions, we describe maximal decompositions in which each resulting relation is not further decomposable. These relations which are not further decomposable are called elementary relations. Since there are non-decomposable relations, we mention how one might approach them if he nevertheless wants to reduce their ranks. At the end of the chapter some characterizations of decompositions are given.

In a relational system $(S, \{\alpha_i\}_{i\in I})$ each relation α_i can be of any rank. Even if they are all necessary relations, some relations may be obtained as projections of others and some relations may be constructed from the combinations of lower rank relations especially if some of the relations can be decomposed into lower rank relations. And if we decompose them into lower rank relations, we may need to obtain the original relations from their decompositions. On the other hand, if we can compact a set of relations into one relation, it is quite possible that we have a reduction in storage space. These aspects motivate the consideration of the converse problem to decomposition, which is called synthesis.

In Chapter 4, after defining the m-synthesis in its most general

form, we give various types of synthesis by placing restrictions on the mapping function. Then we mention the importance of loss-free syntheses which are those from which all original relations which took part in the synthesis can be derived. Next we consider the consistency problem by introducing the concept of relational spaces. Our main result is that $\Sigma(\Omega) \subset \Sigma(\Omega^*)$, which says that all relations that can be obtained from a set of relations by using any number of projections and syntheses can also be generated from the skeleton of Ω. Finally, because of the complexity of general cases, we primarily concentrate on the compaction of binary relations and determine when $m(m-1)/2$ different binary relations can be put together into one m-ary relation.

Then, in Chapter 5, we develop a relational data structure based on the results obtained in the previous chapters since none of the existing data structures seem to be suitable for the representation of relational systems. It turns out that the data structure is general enough to represent sets, relations and tables, regarding sets as unary relations and relations and tables as higher rank relations. It is pointed out that Codd's relational model of data bases is too specialized for formatted tables, i.e., data bases with keys, and Childs' set theoretical model uses only sets while LEAP's associative model is concerned with only binary relations. Fundamental operations are defined for the relational data structure. These operations include set theoretical ones as well as relational operations defined in the previous chapters. This relational data structure is intended to be independent of machines and also the operations are supposed to be independent of the embedding languages.

Finally, two applications of the concept of relational system are presented in Chapter 6. First, we formalize the problem of pattern classifi-

cation by using relational systems and apply it to the problem of synthesis of patterns. The idea is illustrated by using the set of Arabic numerals. Secondly, an algorithm to find all cliques by synthesizing complete subgraphs from smaller complete subgraphs is described. This approach is different from the existing ones in that the former is edge-oriented while the latter is node-oriented.

7.2 Remaining Problems

Some open problems are pointed out as we proceed through the study. We will list some further problems below.

1. Although it seems very difficult, it would be desirable to have a better method for checking decomposability which is convenient enough for practical use. It is true, however, that we can easily determine the decomposability of a given relation quite often if the relation is meaningful in the real world, i.e., not an arbitrary table and the like. Therefore the troublesome relations in this respect are mainly tables. And if the tables have keys, then we may use the results by Rissanen and Delobel [33].

2. There is a fundamental problem for the relational systems approach. That is the fact that every relation has to have its unique specified rank. And if the ranks of relations are different, the relations are different. This seems to be a limitation of the approach. Sometimes we do not want to specify the rank for some relations. For example, cliques can be relations of rank 2, 3, ..., and n in general, depending on the number of nodes in the cliques. Therefore it seems more natural to have a relation "clique" without specified rank than to have many different relations for "clique", such as "clique of order 2" of rank 2, "clique of order 3" of rank 3, and so on. It is probable that the solution to this problem will give

the relation system approach more flexibility.

3. In this study we exclude irregular decompositions, i.e., non-regular decompositions, partly because regular decompositions are less complicated in the process, i.e., automatic, in a sense, compared with irregular ones, and partly because irregular decompositions are difficult to find and characterize. We have not found a good approach to irregular decompositions of relations excluding tables with keys, i.e., Codd's decomposition. Irregular decompositions are attractive because they decompose relations into fewer relations than regular decompositions and some relations which cannot be decomposed by regular decompositions may be decomposed by irregular ones. Therefore the study of irregular decompositions seems to be fruitful. Furthermore we do not know much about the relationships between irregular and regular decompositions. For example, sometimes we notice that some of the relations in a regular decomposition are redundant, i.e., the original relation can be constructed from a synthesis without these relations.

4. The relational data structure described here has not been implemented. Therefore the efficiency and storage requirements are not certain at this moment. Also in order to implement the data structure, more details will have to be worked out.

7.3 Conclusion

In short, this study pays its attention to relational systems as an abstract model of various important systems. Since relational systems generally contain higher rank relations, relations, especially higher rank relations are studied. Two important concepts of relations, decompositions and syntheses, are developed. Also various related problems are considered.

It is concluded that this study contributes to the theory of relations and that the usefulness of the study is demonstrated by presenting a relational data structure and some applications.

BIBLIOGRAPHY

1. Augustson, J. G., and Minker, J., "An Analysis of Some Graph Theoretical Cluster Techniques," J. ACM, Vol. 17, No. 4 (October 1970) pp 571-588.

2. Berge, C., Graphs and Hypergraphs, North-Holland Pub. Co., London, 1973.

3. Bron, C., and Kerbosch, J., "Algorithm 457: Finding All Cliques of an Undirected Graph H," Comm. ACM, Vol. 16, No. 9 (September 1973) pp 575-577.

4. Childs, D. L., "Description of a set-theoretical data structure," FJCC, 1968, pp 557-561.

5. Childs, D. L., "Feasibility of a set-theoretic data structure - a general structure based on a reconstituted definition of relation," Proc. Information Processing 68, North-Holland, Amsterdam (1969) pp 162-172.

6. Clowes, M. B., "Pictorial relationships - a syntactic approach," Machine Intelligence 4, ed. by Meltzer, B., and Michie, D., American Elsevier, New York, 1969, pp 361-383.

7. Codd, E. F., "A Relational Model of Data for Large Shared Data Banks," Comm. ACM, Vol. 13, No. 6 (June, 1970) pp 377-387.

8. Coles, L. S., "An on-line question-answering system with natural language and pictorial input," Proc. ACM 23rd Nat. Conf. 1968, pp 157-167.

9. Evans, T. G., "Descriptive pattern analysis techniques," in Automatic interpretation and classification of images, ed. by Grasselli, A., Academic Press, New York, 1969.

10. Evans, T. G., "A Program for the Solution of a Class of Geometric-Analogy Intelligence-Test Questions," in Semantic Information Processing, ed. by Minsky, M., MIT Press, Cambridge, 1968.

11. Fu, K. S., "Linguistic Approach to Pattern Recognition," Chapter 4 in Applied Computation Theory, ed. by Yeh, R. T., Prentice Hall, 1974.

12. Gray, J. C., "Compound data structures for computer aided design: a survey," Proc. ACM Nat. Conf. (August 1967) pp 355-365.

13. Guzman, A., "Analysis of Curved Line Drawings Using Context and Global Information," in Machine Intelligence 6, ed. by Meltzer, B., and Michie, D., American Elsevier, New York, 1971, pp 325-375.

14. Guzman, A., "Description of a Visual Scene into Three Dimensional Bodies," Proc. Fall Joint Computer Conference, Vol. 33 (December 1968) pp 291-304.

15. Harary, F., Graph Theory, Addison-Wesley, Reading, Mass., 1969.

16. Harary, F., Norman, R. Z. and Cartwright, D., Structural Models: an introduction to the theory of directed graphs, Wiley, New York, 1965.

17. Holt, A. W., "Comparative Religion in Character Recognition Machines," IEEE Computer Group News (November 1968) pp 3-11.

18. Jardine, N. and Sibson, R., "A Model for Taxonomy," Math. Biosci., Vol. 2 (1968) pp 465-482.

19. Kevin, V. and Whitney, M., "A Relational Data Management System (RDMS)," Research Publication GMR-1293 (October, 1972) GM Research Laboratories, Warren, Michigan.

20. Kochen, M., "Automatic question-answering of English-like questions about simple diagrams," J. ACM, Vol. 16, No. 1 (January, 1969) pp 26-48.

21. Language Structure Group of the CODASYL Development Committee, "An Information Algebra, Phase I Report," Comm. ACM, Vol. 5, No. 4 (April, 1962) pp 190-204.

22. LISP 1.5 Programmer's Manual, MIT Press, Cambridge, Mass., 1965.

23. Londe, D. L. and Simmons, R. F., "NAMER: a pattern-recognition system for generating sentences about relations between line drawings," Proc. ACM Nat. Conf. (August, 1965) pp 162-175.

24. Lowenthal, E. I., "A Functional Approach to the Design of Storage Structures for Generalized Data Management Systems," Ph.D. Diss., Dept. of Computer Sciences, The University of Texas, Austin, 1971.

25. McGee, W. C., "File Structures for Generalized Data Management," Proc. Information Processing 68, North-Holland, Amsterdam (1969) pp 1233-1239.

26. Miller, W. F. and Shaw, A. C., "Linguistic methods in picture processing - a survey," FJCC (December, 1968) pp 279-290.

27. Mulligan, G. D. and Corneil, D. G., "Corrections to Bierstone's Algorithm for Generating Cliques," J. ACM, Vol. 19, No. 2 (April, 1972) pp 244-247.

28. Newman, W. M. and Sproull, R. F., Principles of Interactive Computer Graphics, McGraw-Hill, New York, 1973.

29. Ore, O., Theory of Graphs, American Mathematical Society Colloquium Publications, Vol. XXXVIII, American Mathematical Society, Providence, Rhode Island, 1962.

30. Preparata, F. P. and Yeh, R. T., Introduction to Discrete Structures, Addison-Wesley, Reading, Mass., 1973.

31. Quillian, M. R., "Semantic Memory," Chapter 4, Semantic Information Processing, ed. by Minsky, M., MIT Press, 1968.

32. Raphael, B., "SIR: A Computer Program for Semantic Information Retrieval," Chapter 2, Semantic Information Processing, ed. by Minsky, M., MIT Press, 1968.

33. Rissanen, J. and Delobel, C., "Decomposition of Files, A Basis for Data Storage and Retrieval," (unpublished paper).

34. Rosenfeld, A., Picture Processing by Computer, Academic Press, New York, 1969.

35. Simmons, R. F. and Slocum, J., "Generating English Discourse from Semantic Networks," Natural Language Research for Computer Assisted Instruction, Tech. Report NL-3, University of Texas, Austin, November 1970.

36. Tesler, L., Enea, H. and Colby, K. M., "A Directed Graph Representation for Computer Simulation of Belief Systems," Mathematical Biosciences, Vol. 2, Nos. 1/2 (February 1968) pp 19-40.

37. William, R., "A survey of data structures for computer graphics systems," Computing Surveys, Vol. 3, No. 1 (March 1971) pp 1-21.

38. Winston, P. H., "Learing Structural Descriptions from Examples," Tech. Report MAC TR-76, Project MAC, MIT, Cambridge, Mass. (September 1970).

39. Yeh, R. T., "Generalized Pair Algebra with Application to Automata Theory," J. ACM, Vol. 15, No. 2 (April 1968) pp 304-316.

40. Yeh, R. T., "Toward an Algebraic Theory of Fuzzy Relational Systems," Technical Report TR-25, Dept. of Computer Sciences, University of Texas, Austin, July 1973.

VITA

Sung Yang Bang was born in Ichinomiya, Japan, on May 8, 1942, as the third son of Eun Soo Bang and Jum Boon Lee both of whom are Koreans. After completing his work at the Gifu High School, Gifu, Japan, in 1961, he entered Kyoto University, Kyoto, Japan. He received the degree of Bachelor of Science with a major in Electrical Engineering from the university in March 1966. In the same year he went back to Korea to enter the Graduate School of Seoul National University. During the following two years, he studied at the school. He was awarded the degree of Master of Science in Electrical Engineering in February 1969. Then, in September of the same year, he entered the Graduate School of the University of Texas at Austin. Since then he has been a teaching assistant and a research assistant of the Department of Computer Sciences in the university.

Permanent address: 1807 Brazos
Austin, Texas 78701

This dissertation was typed by Sally Sawyer.