BANACH CENTER PUBLICATIONS
VOLUME 2

ON INFORMATION STORAGE AND RETRIEVAL SYSTEMS

WITOLD LIPSKI, Jr.

Institute of Computer Science, Polish Academy of Sciences, PKiN, P.O. Box 22, 00-901 Warsaw

WIKTOR MAREK

Institute of Mathematics, Polish Academy of Sciences, Śniadeckich 8, P.O. Box 137, 00-950 Warsaw

CONTENTS

0. Introduction

This paper reports the activities of the participants of the Seminar on Information Storage and Retrieval Systems conducted at the Banach Center during the Mathematical Foundations of Computer Science Semester, and of the group formed at that seminar to continue the studies.

The activities are continued at seminars at the Institute of Computer Science of the Polish Academy of Sciences. The purpose of this work is an attempt to create a sort of common language to describe what is being done by the designers and by the users of information storage and retrieval systems. Moreover, it is expected to facilitate the teaching of the subject, being a proposal of uniform foundations.

We do not claim that the purpose has been fully achieved, although a concrete proposal has developed.

We hope that extensive use of the methods of discrete mathematics (by which we mean logic and combinatorial mathematics) should help towards a better understanding and a better organization of activities connected with information storage and retrieval.

Historically, the subject was started independently by Wong and Chiang [26] and Pawlak [24]. The ideas of [24] were elaborated by Marek and Pawlak in [19], [20], and finally in [21]. These papers were concerned mainly with a formalization of the subject and investigated the logical properties of the adjoined language. Then, in [16], combinatorial problems were tackled, connected with organizing the memory of a computer while implementing an information storage and retrieval system. Further developments along this line were presented in [11], [12], [13], [14], [17]. All this is presented in some detail in the present paper. The more recent developments are outlined in the last section, entitled "Perspectives".

The paper is organized as follows:

In Section 1 a very simple information storage and retrieval system is presented. Clearly, this system is not taken from real life, since it consists of only 50 objects. Nevertheless, it may serve as a good example of the kind of systems that will be dealt with in the remainder of the study. Then, in the next two sections, the syntax and the semantics of a formal language tailored to deal with the problem, is introduced. This language is a kind of intermediate language between propositional and predicate calculi. As in the latter, there are two levels of the language: terms and formulas. Terms are interpreted as sets of objects (documents or records) of a system, whereas formulas are interpreted as truth values (truth or falsity). A query submitted to a system can be either a term or a formula. In the first case the response of the system is the set of records relevant to the query. In the second case the response is "yes" or "no".

We introduce the set of axioms for terms and formulas. Terms are ruled by the axioms of Boolean algebra enriched by certain specific axioms, for formulas we have the propositional calculus axioms. In the resulting formal system we are able to prove theorems, and to transform, in a uniform way, terms and formulas into equivalent, more suitable forms. Consequently, we can, to some extent, shift the burden from processing large files of documents to processing strings of symbols (expressions of our language), the latter being far easier. The normal forms described in Section 4 are examples of expressions suitable for the computer, into which we transform the queries entering a system. Moreover, these forms are the main tool of proving, in Section 5, the completeness theorems which confirm that our axioms were adequately chosen. In the next two sections algebraic properties of our systems are investigated; in particular, such notions as a subsystem, a quotient system, etc. are introduced. In Section 8 an implementational proposal, based on transforming the queries into a normal form, is presented. In the next few sections certain combinatorial problems are studied, connected with the organization of the memory of an information storage and retrieval system implemented on a computer. Classes of families of sets are introduced, which can be effectively stored in storage devices of different types, and their structure is studied in a systematic way. A number of implementational algorithms based on this theory is presented. In the last section some recent developments are outlined. In particular, certain proposals concerning possible extensions of the language and some implementational methods are included.

Throughout the text we use the standard mathematical notation, such as $\mathscr{P}(X)$ (the power set of X), $|X|$ (the cardinality of X), $\mathscr{D}S$, $\mathscr{R}S$ (the domain and the range of a function S, respectively).

Having been active participants and organizers of the seminar on information storage and retrieval at the semester and after it, we want to express our gratitude to Professor Z. Pawlak who continuously stimulated the activities of our group and actively participated in them.

1. An example

Suppose an information storage and retrieval system contains information about a group of 50 human beings. We say that these human beings are the *objects* of our system. Let each object be classified in respect of three *attributes*: sex, profession and age. For each attribute we have a set of associated *descriptors*:

sex

α) male,
β) female;

profession

a) clerk,
b) farmer,
c) other professions,
d) no profession;

age

A) less than 25,
B) between 25 and 50,
C) more than 50.

Our classification is exhaustive, in the sense that the descriptors within each attribute exhaust all the possibilities. If a classification does not satisfy this condition then new descriptors of the type *other* profession, *no* profession can be added to satisfy it. The second important feature of our classification is that it is exclusive, i.e., within one attribute two descriptors exclude one another. Again, if this is not satisfied for two descriptors a_1 and a_2, then we replace a_1, a_2 by three descriptors: "a_1 and not a_2", "a_1 and a_2", "a_2 and not a_1". Each object can be given its *description*—the sequence of descriptors corresponding to it, one descriptor for each attribute. If our objects are indexed by natural numbers $1, 2, \ldots, 50$, then the list of descriptions and corresponding objects may be, for instance, as follows:

αaA $\{4\}$,
αaB $\{19\}$,
αaC $\{11\}$,
αbA $\{17, 28, 46\}$,
αbB $\{1, 20, 33, 38, 44, 50\}$,

αbC $\emptyset$,
αcA $\emptyset$,
αcB $\{7\}$,
αcC $\{16, 24, 29, 40, 43\}$,
αdA $\{10, 30, 42\}$,
αdB $\emptyset$,
αdC $\{3, 25, 39\}$,
βaA $\{2, 6, 9, 23, 34, 35, 49\}$,
βaB $\emptyset$,
βaC $\emptyset$,
βbA $\{8, 21, 31, 37, 45\}$,
βbB $\{12\}$,
βbC $\emptyset$,
βcA $\{13, 22, 32, 41, 48\}$,
βcB $\emptyset$,
βcC $\{15, 27\}$,
βdA $\{5\}$,
βdB $\{14, 36, 47\}$,
βdC $\{18, 26\}$.

From this list we can determine, for each descriptor, the set of objects having an occurrence of this descriptor in their descriptions:

$$U(\alpha) = \{4, 19, 11, 17, 28, 46, 1, 20, 33, 38, 44, 50, 7, 16, 24, 29, 40, 43, 10, 30, 42, 3, 25, 39\},$$

$$U(\beta) = \{2, 6, 9, 23, 34, 35, 49, 8, 21, 31, 37, 45, 12, 13, 22, 32, 41, 48, 15, 27, 5, 14, 36, 47, 18, 26\},$$

$$U(a) = \{4, 19, 11, 2, 6, 9, 23, 34, 35, 49\},$$

$$U(b) = \{17, 28, 46, 1, 20, 33, 38, 44, 50, 8, 21, 31, 37, 45, 12\},$$

$$U(c) = \{7, 16, 24, 29, 40, 43, 13, 22, 32, 41, 48, 15, 27\},$$

$$U(d) = \{10, 30, 42, 3, 25, 39, 5, 14, 36, 47, 18, 26\},$$

$$U(A) = \{4, 17, 28, 46, 10, 30, 42, 2, 6, 9, 23, 34, 35, 49, 8, 21, 31, 37, 45, 13, 22, 32, 41, 48, 5\},$$

$$U(B) = \{19, 1, 20, 33, 38, 44, 50, 7, 12, 14, 36, 47\},$$

$$U(C) = \{11, 16, 24, 29, 40, 43, 3, 25, 39, 15, 27, 18, 26\}.$$

Conversely, from the above list one can easily determine the previous one, e.g., to the description αbA corresponds the set $U(\alpha) \cap U(b) \cap U(A)$, and so on.

The method based on storing the second list in the memory of a computer is known as the method of *inverted files*. The first list is the basis for the method to be described in this paper (see Section 8). Notice that in the method of inverted files each object is stored as many times as there are attributes, in our case 3 times, whereas in the first list each object occurs exactly once.

A user may submit to the system queries of the type "men of age less than 25", "women of clerical profession or with no profession", etc. The responses for these queries are the following sets of objects: $\{4, 17, 28, 46, 10, 30, 42\}$ and $\{2, 6, 9, 23, 34, 35, 49, 5, 14, 36, 47, 18, 26\}$, respectively. An example of another type of query is "do all men of age less than 25 have a profession?". The response is then "no". All these queries will be formulated in a special formal language to be described in the next few sections.

2. Syntax and axiomatization

In this section we shall define the syntax of a formal language set up to describe certain sets of objects and to make assertions about information storage and retrieval (i.s.r.) systems.

Let A be a nonempty set. Its elements will be referred to as *descriptors*. With each such set A we associate a formal language called the *description language* and denoted by $\mathscr{L}_A$.

DEFINITION 2.1 (*Alphabet of the language* $\mathscr{L}_A$). The *alphabet* of $\mathscr{L}_A$ consists of:

(i) all the descriptors from A taken as constants of the language,

(ii) constants T, F,

(iii) symbols for Boolean operations $\sim$, $+$, $\cdot$, $\rightarrow$,

(iv) symbol for equality $=$,

(v) logical constants $\diagup$, $\diagdown$ (truth and falsity),

(vi) logical connectives $\neg$, $\vee$, $\wedge$, $\Rightarrow$.

Notice that our language does not contain variables.

DEFINITION 2.2 (*Terms of the language* $\mathscr{L}_A$). The set $\mathscr{T}$ of *terms* of $\mathscr{L}_A$ is the least set $\mathscr{T}'$ satisfying the following two conditions:

(i) $a \in \mathscr{T}'$ for each $a \in A$, $T \in \mathscr{T}'$, $F \in \mathscr{T}'$,

(ii) if $t, s \in \mathscr{T}'$ then $\sim t, t+s, t \cdot s, t \rightarrow s \in \mathscr{T}'$.

Parentheses are used, if necessary, in the obvious way. As will turn out later, the order of a sum or product is immaterial, and so we shall abbreviate finite sums and products as $\sum_{i \in J} t_i$ and $\prod_{i \in J} t_i$, respectively. Intentionally, terms are names of certain features of objects, more "complex" than those expressed by descriptors.

DEFINITION 2.3 (*Formulas of the language* $\mathscr{L}_A$). The set $\mathscr{F}$ of *formulas* of $\mathscr{L}_A$ is the least set $\mathscr{F}'$ satisfying the following two conditions:

(i) $\diagup \in \mathscr{F}'$, $\diagdown \in \mathscr{F}'$, if $t, s \in \mathscr{T}$ then $t = s \in \mathscr{F}'$,

(ii) if $\Phi, \Psi \in \mathscr{F}'$ then $\neg\Phi, \Phi \vee \Psi, \Phi \wedge \Psi, \Phi \Rightarrow \Psi \in \mathscr{F}'$.

As in the case of terms, we insert parentheses if necessary, and we abbreviate finite disjunctions and conjunctions to $\bigvee_{k \in K} \Phi_k$ and $\bigwedge_{k \in K} \Phi_k$, respectively. Formulas of the forms $\diagup$, $\diagdown$, $t = s$ are called *atomic*. We shall always use the letters t, s (possibly with indices) to denote terms, and the letters Φ, Ψ (possibly with indices) to denote formulas.

Now we are going to define a set of formulas of the language $\mathscr{L}_A$ which will serve as the set of *axioms* for the theory of i.s.r. systems. In order to do this we assume that there are descriptors of "different kinds" in A. More exactly, there is a set I of *attributes*, and to each attribute $i \in I$ there corresponds a finite nonempty set $A_i \subseteq A$, in such a way that $\bigcup_{i\in I} A_i = A$ and, for all $i, j \in I$, $i \neq j \Rightarrow A_i \cap A_j = \emptyset$. If $a \in A_i$, then we say that a is a *descriptor of attribute* i. This descriptor may then be conceived of as a possible value of attribute i. Instead of speaking about the partition $\{A_i\}_{i\in I}$ we may consider the equivalence relation R_I on A such that the family of its equivalence classes is indexed by the set I and $A/R_I = \{A_i\}_{i\in I}$.

DEFINITION 2.4 (*Axiomatization*). We assume as axioms:

(i) Substitutions of the axioms of Boolean algebra (see Lyndon [18], p. 30) for terms (including $t \to s = \sim t+s$), and the axioms of equality (see Lyndon [18], p. 58).

(ii) For each $a \in A$ the formula

$$a = \sim \sum_{\substack{bR_Ia \\ b\neq a}} b.$$

(iii) Substitutions of the propositional calculus axioms (see e.g., Shoenfield [25]) for formulas.

Note that the restriction imposed on R_I, namely that all its equivalence classes should be finite, is essential in (ii). In the case when some A_i is infinite the sum on the right hand side of (ii) may be undefined. We could overcome this obstacle by allowing infinite sums operator into the language. This leads to a parallel, more general theory. However, we shall not pursue the matter here.

As an inference rule we take *modus ponens* (i.e. from Φ and $\Phi \Rightarrow \Psi$ we infer Ψ). The resulting formalized system—consisting of the language $\mathscr{L}_A$, the set of axioms and the inference rule—enables us to prove formulas. We write $\vdash \Phi$ if Φ is *provable*, i.e. if there exists a proof of Φ. We also denote by $\mathscr{A} \vdash \Phi$ the fact that there exists a derivation from a set $\mathscr{A}$ of formulas as premises to the formula Φ as the conclusion (see Lyndon [18], p. 44). Usually we write $\psi \vdash \Phi$ instead of $\{\psi\} \vdash \Phi$. The following theorem often enables us to simplify proofs.

THEOREM 2.5 (*Deduction theorem*). $\mathscr{A} \cup \{\Psi\} \vdash \Phi$ *iff* $\mathscr{A} \vdash \Psi \Rightarrow \Phi$.

Proof. If $\mathscr{A} \vdash \Psi \Rightarrow \Phi$ then, since we have modus ponens as the rule of inference, $\mathscr{A} \cup \{\psi\} \vdash \Phi$. Assume that $\mathscr{A} \cup \{\psi\} \vdash \Phi$ and let $\Phi_0, \Phi_1, \ldots, \Phi_n = \Phi$ be a derivation of Φ from $\mathscr{A} \cup \{\Psi\}$. Using the propositional calculus theorems

$$\Psi \Rightarrow \Psi,$$

$$\Phi_i \Rightarrow (\Psi \Rightarrow \Phi_i),$$

$$\big(\Psi \Rightarrow (\Phi_i \Rightarrow \Phi_j)\big) \Rightarrow \big((\Psi \Rightarrow \Phi_i) \Rightarrow (\Psi \Rightarrow \Phi_j)\big),$$

one can easily prove, by induction on i, that for each i, $\mathscr{A} \vdash \Psi \Rightarrow \Phi_i$. For details the reader is referred to Lyndon [18], p. 50. ■

3. Semantics, interpretation of terms and formulas

We shall now define an interpretation of the language $\mathscr{L}_A$. This interpretation assigns a subset of a fixed set to each term, and a truth value, truth or falsity, to each formula.

DEFINITION 3.1 (*Basic definition*). An *information storage and retrieval system* (an *i.s.r. system* for short) is a quadruple

$$\mathscr{S} = \langle X, A, R_I, U \rangle$$

where

(i) X is a set called the *set of objects*,

(ii) A is a nonempty set of descriptors and R_I is an equivalence on A as described in Section 2.

(iii) $U\colon A \to \mathscr{P}(X)$ is a function satisfying the following two conditions:

(1) If aR_Ib and $a \neq b$ then $U(a) \cap U(b) = \emptyset$.

(2) $\bigcup \{U(b)\colon bR_Ia\} = X$, for each $a \in A$.

Note that conditions (1) and (2) can be expressed equivalently by one condition, namely

$$U(a) = X \setminus \bigcup \{U(b)\colon aR_Ib \wedge a \neq b\} \quad \text{for each } a \in A.$$

The interpretation of terms will now be defined as a natural extension of the function U.

DEFINITION 3.2 (*Valuation of terms*). Let $\mathscr{S} = \langle X, A, R_I, U \rangle$ be an i.s.r. system. The *value of a term* t in $\mathscr{S}$, denoted by $||t||_{\mathscr{S}}$, is defined inductively as follows:

(i) $||a||_{\mathscr{S}} = U(a)$ for each $a \in A$,

(ii) $||T||_{\mathscr{S}} = X$,

(iii) $||F||_{\mathscr{S}} = \emptyset$,

(iv) $||\sim t||_{\mathscr{S}} = X \setminus ||t||_{\mathscr{S}}$,

(v) $||t+s||_{\mathscr{S}} = ||t||_{\mathscr{S}} \cup ||s||_{\mathscr{S}}$,

(vi) $||t \cdot s||_{\mathscr{S}} = ||t||_{\mathscr{S}} \cap ||s||_{\mathscr{S}}$,

(vii) $||t \to s||_{\mathscr{S}} = (X \setminus ||t||_{\mathscr{S}}) \cup ||s||_{\mathscr{S}}$.

Thus terms, as we promised, serve as descriptions of sets of objects.

Unlike the terms, formulas will take as values the truth values $\diagup$ (truth) and $\diagdown$ (falsity).

DEFINITION 3.3 (*Valuation of formulas*). Assume that the values of the terms are already defined. We define inductively the *value of a formula* Φ in $\mathscr{S}$, denoted by $||\Phi||_{\mathscr{S}}$, as follows:

(i) $||\diagup||_{\mathscr{S}} = \diagup$, $\quad ||\diagdown||_{\mathscr{S}} = \diagdown$;

(ii) $||t = s||_{\mathscr{S}} = \begin{cases} \diagup & \text{if } \quad ||t||_{\mathscr{S}} = ||s||_{\mathscr{S}}, \\ \diagdown & \text{otherwise;} \end{cases}$

(iii) $||\neg \Phi||_{\mathscr{S}} = \begin{cases} \diagup & \text{if } \quad ||\Phi||_{\mathscr{S}} = \diagdown, \\ \diagdown & \text{if } \quad ||\Phi||_{\mathscr{S}} = \diagup; \end{cases}$

(iv) $\|\Phi \vee \Psi\|_{\mathscr{S}} = \begin{cases} \checkmark & \text{if } \|\Phi\|_{\mathscr{S}} = \checkmark \text{ or } \|\Psi\|_{\mathscr{S}} = \checkmark, \\ \nearrow & \text{otherwise;} \end{cases}$

(v) $\|\Phi \wedge \Psi\|_{\mathscr{S}} = \begin{cases} \checkmark & \text{if } \|\Phi\|_{\mathscr{S}} = \checkmark \text{ and } \|\Psi\|_{\mathscr{S}} = \checkmark, \\ \nearrow & \text{otherwise;} \end{cases}$

(vi) $\|\Phi \Rightarrow \Psi\|_{\mathscr{S}} = \begin{cases} \checkmark & \text{if } \|\Phi\|_{\mathscr{S}} = \nearrow \text{ or } \|\Psi\|_{\mathscr{S}} = \checkmark, \\ \nearrow & \text{otherwise.} \end{cases}$

If $\|\Phi\|_{\mathscr{S}} = \checkmark$ then we also write $\mathscr{S} \models \Phi$ and say that Φ is *true* in $\mathscr{S}$ (otherwise we write $\mathscr{S}$ non $\models \Phi$ and say that Φ is *false* in $\mathscr{S}$).

Sometimes, when $\mathscr{S}$ is clear from the context, we write simply $\|t\|$, $\|\Phi\|$ instead of $\|t\|_{\mathscr{S}}$, $\|\Phi\|_{\mathscr{S}}$.

THEOREM 3.4 (*Adequacy of axiomatization*). *If Φ is an axiom* (see Def. 2.4), *then* $\|\Phi\|_{\mathscr{S}} = \checkmark$.

Proof. Our valuation was defined in such a way as to make the axioms of groups (i) and (iii) true. The axioms of group (ii) are also true, by the remark after Definition 3.1. ■

DEFINITION 3.5. Let $\mathscr{S} = \langle X, A, R_I, U\rangle$ be an i.s.r. system and let $x \in X$.

(a) The *information* on x in $\mathscr{S}$ is a function $f_x\colon I \to A$ such that for all $i \in I$, $f_x(i) \in A_i$ and $x \in U(f_x(i))$.

(b) The *description* of x in $\mathscr{S}$ is the term $t_x = \prod_{i\in I} f_x(i)$.

It is easy to see that f_x is uniquely defined. Moreover, if we know f_x for all $x \in X$ then we can reconstruct the function U by the formula

$$U(a) = \{x\colon f_x(i) = a \text{ for the unique } i \in I \text{ such that } a \in A_i\}.$$

In the definition of description we must assume that I is finite, since we do not allow terms of infinite length in our language. This definition is correct since we identify products with factors occurring in a different order.

DEFINITION 3.6. An i.s.r. system $\mathscr{S}$ is *selective* if for each $x \in X$, $\|t_x\|_{\mathscr{S}} = \{x\}$.

Thus a selective system is one in which different objects have always different descriptions, i.e. are distinguishable in our language.

4. Normal forms for terms and formulas

Normal form theorems which we are now going to prove play an important role in the theory of i.s.r. systems. On the one hand, they are the main tool for proving completeness results in the next section, on the other hand they have deep implementational consequences (see Section 8).

It will be convenient to use the following notation: $a^1 = a$, $a^0 = \sim a$ for each $a \in A$, similarly Φ^1 denotes Φ and Φ^0 denotes $\neg\Phi$. We also assume that if $J = \emptyset$ then $\sum_{i\in J} t_i$ and $\prod_{i\in J} t_i$ denote F and T, respectively. From now on we consider only the case when the set of attributes I is finite.

DEFINITION 4.1. (a) A term t is *primitive* if $t = \prod_{j\in J} a_j^{\varepsilon_j}$ where each ε_j is 1 or 0.

(b) A term t is in *normal additive form* if $t = \sum_{j\in J} t_j$ where each t_j is primitive.

(c) A term is *positive* if $\neg$, $\to$ do not occur in it.

(d) A term is *simple* if it is of the form $\prod_{i\in I} a_i$ where $a_i \in A_i$ for all $i \in I$. We denote the set of all simple terms by $\mathscr{T}_0$.

(e) A term t is of *standard form* if $t = \sum_{j\in J} t_j$ where each t_j is simple and all t_j's are different.

Notice that each description (see Def. 3.5(b)) is a simple term.

THEOREM 4.2 (*Normal forms for terms*). (a) *For each term t there is a term s in normal additive form such that* $\vdash t = s$.

(b) *For each term t there is a positive term s in normal additive form such that* $\vdash t = s$.

(c) *For each term t there is a term s in standard form such that* $\vdash t = s$.

Proof. (a) The reasoning used in this case is a standard one. Only the axioms of group (i) are needed here. The reader is referred to any standard textbook on Boolean algebras.

(b) By (a) we may assume that t is already in normal additive form. From each axiom of group (ii) one can easily obtain $\vdash \sim a = \sum_{\substack{bR_Ia \\ b\neq a}} b$. Substituting the right-hand side in every place where the left-hand side occurs, we eliminate negation from t. A repeated application of the distributive law completes the proof.

(c) By (b) it is clear that we may assume that t is a positive primitive term. If two different $a, b \in A_i$ occur in t for some $i \in I$, we have $\vdash t = F$, since then one can derive $a \cdot b = F$ from the axiom $a = \sim \sum_{\substack{bR_Ia \\ b\neq a}} b$. Therefore we may assume that at most one $a \in A_i$ occurs in t for each $i \in I$. By adding a to both sides of $\sim a = \sum_{\substack{bR_Ia \\ b\neq a}} b$, we obtain $\vdash \sum_{bR_Ia} b = T$. Let $i \in I$ be an attribute such that no $b \in A_i$ occurs in t. Using $\vdash t = t \cdot T$, we get $\vdash t = t \cdot \sum_{b\in A_i} b$ and, by the distributive law, $\vdash t = \sum_{b\in A_i} t \cdot b$. Thus we have diminished by one the number of attributes which were not represented in t. A repeated application of the above procedure completes the proof. ■

Notice that the finitness of I was essential only in (c). It is clear that there are exactly $N = \prod_{i\in I} |A_i|$ different simple terms (recall that $|A_i|$ is the cardinality of A_i) and consequently 2^N different terms in standard form.

DEFINITION 4.3. (a) A formula is *elementary* if it has the form $t = F$ where t is a simple term.

(b) A formula is *primitive* if it has the form $\bigwedge_{j\in J} \Phi_j^{\varepsilon_j}$ where each Φ_j is elementary and ε_j is 1 or 0.

(c) A formula is in *normal disjunctive form* if it is of the form $\bigvee_{j\in J} \Psi_j$ where each Ψ_j is primitive.

(d) A formula is *positive* if $\neg$, $\Rightarrow$ do not occur in it.

(e) A formula Φ is *basic* if it is of the form $\bigwedge_{k\in K} \Phi_k^{\varepsilon_k}$ where each ε_k is 1 or 0, each Φ_k is elementary and all the N elementary formulas occur in Φ, each one exactly once.

(f) A formula is in *standard form* if it has the form $\bigvee_{l\in L} \Psi_l$ where each Ψ_l is basic and all Ψ_l's are different.

We assume that if $J = \emptyset$ then $\bigvee_{j\in J} \Phi_j$ and $\bigwedge_{j\in J} \Phi_j$ denote $\not|$ and $|$, respectively. Φ and Ψ are said to be *provably equivalent* iff $\vdash \Phi \Leftrightarrow \Psi$ where $\Phi \Leftrightarrow \Psi$ abbreviates $\Phi \Rightarrow \Psi \wedge \Psi \Rightarrow \Phi$.

THEOREM 4.4. (a) *For each atomic formula $t = s$ there is a provably equivalent positive primitive formula.*

(b) *For each formula Φ there is a provably equivalent formula Ψ in normal disjunctive form.*

(c) *For each formula Φ there is a provably equivalent formula Ψ in standard form.*

Proof. (a) By Theorem 4.2(c) we may transform t and s to standard form. Let a simple term t_0 occur in either t or s but not in both. Multiplying both sides of $t = s$ by t_0 we obtain $\vdash s = t \Rightarrow t_0 = F$. Moreover, we can easily prove: $s = t \Leftrightarrow t_1 = F \wedge t_2 = F \wedge \ldots \wedge t_k = F$ where $t_1, t_2, \ldots, t_k$ are all simple terms occurring in t or s but not in both (it is convenient to make use of the Deduction Theorem here).

(b) By (a) we may replace in Φ each equality of terms by a positive primitive formula. Then, using the propositional calculus axioms (which are essentially the same as those of Boolean algebra), we apply the standard procedure similar to that used in the proof of Theorem 4.2(a).

(c) By (b) it is clear that we may assume that Φ is a primitive formula. Now we apply a procedure similar to that used in the proof of Theorem 4.2(c)—instead of $\vdash \sum_{b R_I a} b = T$ we exploit $\vdash t = F \vee t \neq F \Leftrightarrow |$ ($t \neq F$ abbreviates $\neg(t = s)$). ■

Also the form $\bigwedge_{j\in J} \bigvee_{l\in L} \Psi_{jl}^{\varepsilon_{jl}}$ dual to the standard form can be obtained (we take the standard form of $\neg\Phi$ and then we negate it using De Morgan's Laws).

It is clear that there are exactly $N = \prod_{i\in I} |A_i|$ elementary formulas, 2^N basic formulas and 2^{2^N} formulas in standard form. Notice that the elementary formulas

are elementary pieces of information about an i.s.r. system. By Theorem 4.4 every assertion about an i.s.r. system expressible in the language $\mathscr{L}_A$ can be built up from elementary formulas by means of logical connectives.

5. Completeness properties

Let $\mathscr{A}$ be a set of formulas and let Φ be a formula. We shall write $\mathscr{A} \models \Phi$ to denote the fact that $\mathscr{A}$ *semantically implies* Φ, that is, for all i.s.r. systems $\mathscr{S}$ such that each $\Psi \in \mathscr{A}$ is true in $\mathscr{S}$, $||\Phi||_{\mathscr{S}} = |$.

From Theorem 3.4 and the fact that $||\Phi||_{\mathscr{S}} = |$ and $||\Phi \Rightarrow \Psi|| = |$ implies $||\Psi||_{\mathscr{S}} = |$, we obtain the following

THEOREM 5.1 (*Adequacy of inference*). *If $\vdash \Phi$ then for all* i.s.r. *systems $\mathscr{S}$, $||\Phi||_{\mathscr{S}} = |$. More generally, if $\mathscr{A} \vdash \Phi$ then $\mathscr{A} \models \Phi$.* ■

Obviously, we assume here (and in the rest of this section) that the language $\mathscr{L}_A$ and the axiom system are fixed, and so the phrase "for all i.s.r. systems" refers to all i.s.r. systems with fixed A and R_I.

Consider a fixed i.s.r. system $\mathscr{S}$. Using our rule of inference, we may generate from our axioms certain formulas which, by Theorems 3.4 and 5.1, are true in $\mathscr{S}$. Clearly, not every formula which is true in $\mathscr{S}$ can be obtained in this way. If it were so, then each formula would be either true in each i.s.r. system or false in all i.s.r. systems. Nevertheless we shall soon prove that the set of formulas obtained in this way is the "maximal possible" in the sense that the converse of Theorem 5.1 holds. Thus no "better axiomatization" of our theory is possible.

Let us notice that each set of formulas $\mathscr{A}$ is equivalent to a single formula Φ, in the sense that $\mathscr{A} \vdash \Phi$ and for every $\Psi \in \mathscr{A}$, $\Phi \vdash \Psi$. Indeed, we can transform each $\Psi \in \mathscr{A}$ into the standard form, then delete all repeating formulas and take the conjunction of the remaining finite number of formulas (notice that if $\mathscr{A} = \emptyset$ then we obtain $|$ as Φ).

We say that a set of formulas $\mathscr{A}$ is *deductively consistent* if $\mathscr{A}$ non $\vdash \not|$.

THEOREM 5.2 (*Consistency theorem*). *If a set of formulas $\mathscr{A}$ is deductively consistent then there exists an* i.s.r. *system $\mathscr{S}$ such that for all $\Psi \in \mathscr{A}$, $||\Psi||_{\mathscr{S}} = |$.*

Proof. It is clear that we may replace $\mathscr{A}$ by the equivalent single formula Φ in the standard form. Φ is not $\not|$, since that would contradict the consistency of $\mathscr{A}$. Thus Φ contains at least one basic formula Φ_0, say $\bigwedge_{t\in\mathscr{T}^+} (t \neq F) \wedge \bigwedge_{s\in\mathscr{T}^-} (s = F)$ ($\mathscr{T}^+ \cup \mathscr{T}^- = \mathscr{T}_0$, $\mathscr{T}^+ \cap \mathscr{T}^- = \emptyset$). To prove the theorem it is sufficient to construct an i.s.r. system $\mathscr{S}$ such that $||\Phi_0||_{\mathscr{S}} = |$. To this end let us take the Cartesian product $\mathrm{P}_{i\in I} A_i$ and its subset $X = \{f \in \mathrm{P}_{i\in I} A_i : \prod_{i\in I} f(i) \in \mathscr{T}^+\}$. Now define a function U: $A \to \mathscr{P}(X)$ as follows:

$$U(a) = \{f \in X : f(i) = a \text{ for the unique } i \text{ such that } a \in A_i\}.$$

Then $\mathscr{S} = \langle X, A, R_I, U\rangle$ is the required i.s.r. system. ■

From the above construction it is clear that for every basic formula Φ (or equivalently for every subset $\mathscr{T}^+ \subseteq \mathscr{T}_0$) there is a selective i.s.r. system $\mathscr{S}_\Phi$ such that $||\Phi||_{\mathscr{S}_\Phi} = \bigvee$ (or equivalently for each $t \in \mathscr{T}_0$, $||t||^{\text{lo}}_{\mathscr{S}_\Phi} \neq \emptyset$ iff $t \in \mathscr{T}^+$). We shall see later that the most important among these systems is $\mathscr{S}_{\max}$—the system in which all simple terms have nonempty values (i.e. $\mathscr{T}^+ = \mathscr{T}_0$). Conversely, it is easy to see that to every system $\mathscr{S}$ there corresponds a unique basic formula $\Phi_\mathscr{S}$ such that $||\Phi_\mathscr{S}||_\mathscr{S} = \bigvee$ (we simply take $\mathscr{T}^+ = \{t \in \mathscr{T}_0 : ||t||_\mathscr{S} \neq \emptyset\}$). Moreover, the single formula $\Phi_\mathscr{S}$ provides a complete description of $\mathscr{S}$:

THEOREM 5.3. *For every formula* Ψ *and* i.s.r. *system* $\mathscr{S}$

$$||\Psi||_\mathscr{S} = \bigvee \quad \textit{iff} \quad \Phi_\mathscr{S} \vdash \Psi.$$

Proof. Suppose $||\Psi||_\mathscr{S} = \bigvee$. We may assume that Ψ is in the standard form. Since $\Phi_\mathscr{S}$ is the only basic formula true in $\mathscr{S}$, it must occur in Ψ. Hence $\vdash \Phi_\mathscr{S} \Rightarrow \Psi$ and, by modus ponens, $\Phi_\mathscr{S} \vdash \Psi$. Conversely, if $\Phi_\mathscr{S} \vdash \Psi$ then by the fact that $||\Phi_\mathscr{S}||_\mathscr{S} = \bigvee$ and by the adequacy of inference, $||\Psi||_\mathscr{S} = \bigvee$. ■

THEOREM 5.4 (*Completeness theorem*). *Let* $\mathscr{A}$ *be a set of formulas and let* Ψ *be a formula. If* $\mathscr{A} \models \Psi$ *then* $\mathscr{A} \vdash \Psi$.

Proof. We may assume that $\mathscr{A}$ consists of a single formula Φ, and that both Ψ and Φ are in standard form. Suppose that $\mathscr{A}$ non $\vdash \Psi$. Then there must be a basic formula Φ_0 occurring in Φ but not in Ψ (otherwise Ψ could be written as $\Phi \vee \Phi_1$, and then trivially $\Phi \vdash \Psi$). But consider the system $\mathscr{S}_{\Phi_0}$. We have $||\Phi||_{\mathscr{S}_{\Phi_0}} = \bigvee$ and $||\Psi||_{\mathscr{S}_{\Phi_0}} = \bigwedge$, which contradicts the assumptions of our theorem. Thus $\mathscr{A} \vdash \Psi$. ■

In particular, for $\mathscr{A} = \emptyset$, we get the following

COROLLARY 5.5. *If, for all* i.s.r. *systems* $\mathscr{S}$, $||\Phi||_\mathscr{S} = \bigvee$ *then* $\vdash \Phi$. ■

Let us introduce the following relations:

$$\begin{aligned} t \approx s \quad &\text{iff} \quad \vdash t = s, \\ t \leqslant s \quad &\text{iff} \quad \vdash t \cdot s = t, \\ t \sim_\mathscr{S} s \quad &\text{iff} \quad ||t||_\mathscr{S} = ||s||_\mathscr{S}, \\ t \leqslant_\mathscr{S} s \quad &\text{iff} \quad ||t||_\mathscr{S} \subseteq ||s||_\mathscr{S}. \end{aligned}$$

From the completeness theorem we can easily deduce that $t \approx s$ iff for every system $\mathscr{S}$, $t \approx_\mathscr{S} s$. In other words, $\approx = \bigcap_\mathscr{S} \approx_\mathscr{S}$ ($\approx$ and $\approx_\mathscr{S}$ are binary relations on $\mathscr{T}$, i.e. $\approx$, $\approx_\mathscr{S} \subseteq \mathscr{T} \times \mathscr{T}$). Similary, $t \leqslant s$ iff for every system $\mathscr{S}$, $t \leqslant_\mathscr{S} s$

Corollary 5.5 also implies that the formulas Φ and Ψ are provably equivalent iff in each i.s.r. system $\mathscr{S}$, $||\Phi||_\mathscr{S} = ||\Psi||_\mathscr{S}$ (from now on we shall say simply "equivalent" instead of "provably equivalent").

If Φ is (equivalent to) a positive formula, then it is particularly easy to check whether $\vdash \Phi$.

THEOREM 5.6. *If* Φ *is a positive formula then the following three conditions are equivalent*:

(i) $\vdash \Phi$.

(ii) *For all* i.s.r. *systems* $\mathscr{S}$, $||\Phi||_\mathscr{S} = \bigvee$.

(iii) $||\Phi||_{\mathscr{S}_{\max}} = \bigvee$.

Proof. (i) $\Leftrightarrow$ (ii) is just Corollary 5.5, (ii) $\Rightarrow$ (iii) is trivial.

(iii) $\Rightarrow$ (i). Let Φ be an atomic formula $t = s$. By Theorem 4.4(a), Φ is equivalent to a positive primitive formula $\bigwedge^*_{j \in J} t_j = F$. But the last formula is true in $\mathscr{S}_{\max}$ only when $J = \emptyset$. Hence if $||t = s||_{\mathscr{S}_{\max}} = \bigvee$ then $\vdash t = s$. The easy extension to arbitrary positive formulas is left to the reader. ■

It is easy to see that different terms in the standard form always have different values in $\mathscr{S}_{\max}$. Thus the standard form of each term is unique (up to the order of simple terms and the order of descriptors within each simple term). Similarly from the Consistency Theorem it follows that the standard form of each formula is unique.

We say that $\mathscr{S}$ is *equivalent* to $\mathscr{S}'$ (in symbols $\mathscr{S} \equiv \mathscr{S}'$) iff for every Φ, $||\Phi||_\mathscr{S} = ||\Phi||_{\mathscr{S}'}$. It is easy to see that $\mathscr{S} \equiv \mathscr{S}'$ iff the basic formulas $\Phi_\mathscr{S}$ and $\Phi_{\mathscr{S}'}$ coincide. Obviously, $\equiv$ is an equivalence relation. There is a one-one correspondence between its equivalence classes and all basic formulas. Moreover, by Theorem 5.3 each such class can be given a complete axiomatization by adding the basic formula corresponding to it to our axioms listed in Definition 2.4. Since there are 2^N different basic formulas, there are also 2^N classes of equivalent systems (recall that $N = \prod_{i \in I} |A_i|$). For each system $\mathscr{S}$ there is an equivalent selective system $\mathscr{S}'$ (we can take $\mathscr{S}_{\Phi_\mathscr{S}}$ as $\mathscr{S}'$). Notice that the standard form of a formula determines explicitly all the (equivalence classes of) i.s.r. systems in which it is true.

We now come to the problem of limitations of the expressive power of our language $\mathscr{L}_A$. In any particular i.s.r. system it is impossible to distinguish, by means of $\mathscr{L}_A$, any two objects with the same description. Also we are not able to express the cardinality of the value of a term, though we can state that this value is empty (or nonempty, or coincides with the whole set of objects). These are the limitations of the first level of $\mathscr{L}_A$ (terms). Similarly, we cannot distinguish in $\mathscr{L}_A$ any two equivalent systems, and this is the limitation of the second level of $\mathscr{L}_A$ (formulas).

Our theory is decidable (see Lyndon [18], p. 21). The decision procedure to find out whether $\vdash \Phi$ consists in transforming Φ into the standard form and checking whether all the 2^N basic formulas occur in it. One can even introduce a Gentzen-style mechanized proof finding procedure (by a modification of a Hao-Wang algorithm). This will be, however, the subject of a separate publication.

Finally, let us observe that all the completeness results of this section could be proved by standard methods (cf. Lyndon [18]) usually adopted in proving analogous results for the predicate calculus (notice that each model for our axioms is a special kind of Boolean algebra which is, by Stone's representation theorem, isomorphic to an i.s.r. system). But the methods used here are more constructive and give a better insight into our theory. Moreover, they provide many implementational suggestions.

6. Algebraic properties of i.s.r. systems

DEFINITION 6.1. An i.s.r. system $\mathscr{S}_1 = \langle Y, B, R_J, U_1\rangle$ is a *subsystem* of $\mathscr{S} = \langle X, A, R_I, U\rangle$ (in symbols $\mathscr{S}_1 \subseteq \mathscr{S}$) iff

(i) $Y \subseteq X$,

(ii) $B \subseteq A$,

(iii) $R_J = R_I \cap (B \times B)$,

(iv) $U_1(a) = U(a) \cap Y$ for each $a \in B$.

Notice that $\mathscr{S}_1$ is uniquely determined by Y and B. If $B \subseteq A$ then each term of $\mathscr{L}_B$ is, in particular, a term of $\mathscr{L}_A$. The adequacy of our definition is shown by the following

LEMMA 6.2. *Let $\mathscr{S}_1 = \langle Y, B, R_J, U_1\rangle$, $\mathscr{S} = \langle X, A, R_I, U\rangle$ and let $\mathscr{S}_1 \subseteq \mathscr{S}$. Then for each term t of the language $\mathscr{L}_B$,*

$$||t||_{\mathscr{S}_1} = ||t||_{\mathscr{S}} \cap Y.$$

Proof. By the induction on the complexity of t. If t is a single descriptor a then the desired equality is nothing else but (iv) in Definition 6.1. If t is T or F then the condition is seen immediately. Assume now that t is $\sim s$. Then

$$\begin{aligned} ||t||_{\mathscr{S}_1} &= ||\sim s||_{\mathscr{S}_1} = Y \setminus ||s||_{\mathscr{S}_1} = Y \setminus (||s||_{\mathscr{S}} \cap Y) \\ &= X \cap Y \setminus (||s||_{\mathscr{S}} \cap Y) = (X \setminus ||s||_{\mathscr{S}}) \cap Y = ||\sim s||_{\mathscr{S}} \cap Y = ||t||_{\mathscr{S}} \cap Y. \end{aligned}$$

If $t = s_1 \cdot s_2$ then we have

$$\begin{aligned} ||t||_{\mathscr{S}_1} &= ||s_1 \cdot s_2||_{\mathscr{S}_1} = ||s_1||_{\mathscr{S}_1} \cap ||s_2||_{\mathscr{S}_1} = ||s_1||_{\mathscr{S}} \cap Y \cap ||s_2||_{\mathscr{S}} \cap Y \\ &= ||s_1||_{\mathscr{S}} \cap ||s_2||_{\mathscr{S}} \cap Y = ||s_1 \cdot s_2||_{\mathscr{S}} \cap Y = ||t||_{\mathscr{S}} \cap Y. \end{aligned}$$

The cases of the operations $+$, $\to$ are similar. ■

We get two important cases in Definition 6.1 assuming that $B = A$ or $Y = X$. We then write $\mathscr{S}_1 \overset{*}{\subseteq} \mathscr{S}$ or $\mathscr{S}_1 \subseteq \mathscr{S}$, respectively. If $\mathscr{S}_1 \subseteq \mathscr{S}$ then we write also $\mathscr{S}_1 = \mathscr{S}|_Y$ and we say that $\mathscr{S}_1$ is the *restriction* of $\mathscr{S}$ to Y. The following "interpolation property" holds: if $\mathscr{S}_1 \subseteq \mathscr{S}$ then there exist $\mathscr{S}'$, $\mathscr{S}''$ such that $\mathscr{S}_1 \subseteq \mathscr{S}' \overset{*}{\subseteq} \mathscr{S}$ and $\mathscr{S}_1 \overset{*}{\subseteq} \mathscr{S}'' \subseteq \mathscr{S}$. Moreover, $\mathscr{S}'$ and $\mathscr{S}''$ are unique. It is easy to see that for each system $\mathscr{S}_\Phi$ constructed in the preceding section $\mathscr{S}_\Phi \overset{*}{\subseteq} \mathscr{S}_{\max}$.

DEFINITION 6.3. Let $\{\mathscr{S}_j\}_{j\in J}$ be a family of i.s.r. systems with the same set of objects ($\mathscr{S}_j = \langle X, A^j, R_{I_j}, U_j\rangle$), and suppose moreover that, for all $i, j \in J$,

$$i \neq j \Rightarrow A^i \cap A^j = \emptyset \wedge I_i \cap I_j = \emptyset.$$

We define the *direct sum* of $\{\mathscr{S}_j\}_{j\in J}$ as follows:

$$\bigoplus_{j\in J} \mathscr{S}_j = \langle X, A, R_I, U\rangle$$

where

$$A = \bigcup_{j\in J} A^j, \quad I = \bigcup_{j\in J} I_j, \quad R_I = \bigcup_{j\in J} R_{I_j}, \quad U = \bigcup_{j\in J} U_j.$$

For instance, if we take $\mathscr{S} = \langle X, A, R_I, U\rangle$ and, for each $i \in I$, $\mathscr{S}_i = \langle X, A_i, A_i \times A_i, U \mid A_i\rangle$ then $\mathscr{S}_i \overset{*}{\subseteq} \mathscr{S}$ and $\mathscr{S} = \bigoplus_{i\in I} \mathscr{S}_i$.

Now we shall describe a hierarchical construction. Of two equivalence relations S, R on a set A we say that S is *finer* than R (in symbols $S \prec R$) if $S \subseteq R$. We denote the family of the equivalence classes of S by A/S and the equivalence class of an element $a \in A$ by a/S.

DEFINITION 6.4. Let S be an equivalence on A, $S \prec R_I$, and let $\mathscr{S} = \langle X, A, R_I, U\rangle$ be an i.s.r. system. We define the *quotient system* $\mathscr{S}/S$ as follows:

$$\mathscr{S}/S = \langle X, A/S, R_I/S, U/S\rangle$$

where

(i) R_I/S is an equivalence on A/S defined (uniquely) by the formula

$$a/S \; R_I/S \; b/S \Leftrightarrow aR_Ib;$$

(ii) U/S: $A/S \to \mathscr{P}(X)$ is defined as follows:

$$U/S(a/S) = \bigcup \{U(b) : bSa\}.$$

Roughly speaking, we "glue together" certain descriptors (namely those forming the equivalence classes of S) to get descriptors expressing "more general" features. Thus the expressive power of the language $\mathscr{L}_{A/S}$ is less than that of $\mathscr{L}_A$.

DEFINITION 6.5 (*Hierarchical construction*). Let $\mathscr{S} = \langle X, A, R_I, U\rangle$ be an i.s.r. system and let $S_1 \prec S_2 \prec \ldots \prec S_n \prec R_I$ be an increasing sequence of equivalence relations on A. We define $\mathscr{S}_{S_1 \ldots S_n}$ as follows:

$$\mathscr{S}_{S_1 \ldots S_n} = \mathscr{S} \oplus \Big(\bigoplus_{i=1}^{n} \mathscr{S}/S_i\Big).$$

We left to the reader the proof of the following

THEOREM 6.6. *For each $a \in A$ we have*

$$||a/S_{i+1} = \sum_{b/S_i \; S_{i+1}/S_i \; a/S_i} b/S_i||_{\mathscr{S}_{S_1 \ldots S_n}} = \bigvee. \; ■$$

The hierarchical construction is used when our system is "too fine". If our system is "too crude" we can perform, in a sense, an inverse construction. Roughly speaking, for each $a \in A$ we take a partition $\{X_1, X_2, \ldots, X_k\}$ of the set $U(a)$, we add to our language constants $a_1, a_2, \ldots, a_k$ and put $U(a_i) = X_i$. For the details the reader is referred to Marek and Pawlak [21].

7. Describable sets

DEFINITION 7.1. Let $\mathscr{S} = \langle X, A, R_I, U\rangle$ be an i.s.r. system and let $Y \subseteq X$. Y is *describable* in $\mathscr{S}$ if there is a term t such that $||t||_{\mathscr{S}} = Y$. We denote the family of all subsets of X describable in $\mathscr{S}$ by $\mathfrak{B}(\mathscr{S})$.

It can easily be checked that the describable sets form a Boolean algebra. We shall need some auxiliary notions connected with families of sets.

DEFINITION 7.2. (Cf. Kuratowski and Mostowski [10], p. 20.) Let $\mathfrak{M} = \{M_1, M_2, \ldots, M_n\} \subseteq \mathscr{P}(X)$ and let $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n \in \{0, 1\}$.

Sets of the form

$$S_{\mathfrak{M}}(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n) = M_1^{\varepsilon_1} \cap M_2^{\varepsilon_2} \cap \ldots \cap M_n^{\varepsilon_n},$$

where

$$M^{\varepsilon} = \begin{cases} M & \text{if} \quad \varepsilon = 1, \\ X \setminus M & \text{if} \quad \varepsilon = 0, \end{cases}$$

are called the *components* of the family $\mathfrak{M}$.

LEMMA 7.3 (see Kuratowski and Mostowski [10]).

(a) *Different components are disjoint.*

(b) *The union of all components of a family $\mathfrak{M} \subseteq \mathscr{P}(X)$ is X.*

(c) *Each set $Y \subseteq X$ obtained from sets of a family $\mathfrak{M} \subseteq \mathscr{P}(X)$ by means of Boolean operations is the disjoint union of a certain number of components of $\mathfrak{M}$.* ■

By the *components in an* i.s.r. *system* $\mathscr{S} = \langle X, A, R_I, U\rangle$ are meant the components of the family $\mathfrak{M} = \{U(a): a \in A\} \subseteq \mathscr{P}(X)$.

Condition (iii) in Definition 3.1 (or the axioms of group (ii) in Definition 2.4) implies that the components of $\{U(a): a \in A\}$ can be obtained in a much simpler way than in the general case, namely as the values of simple terms. Indeed, let $A = \{a_1, a_2, \ldots, a_n\}$ and let us take

$$S_{\mathfrak{M}}(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n) = U(a_1)^{\varepsilon_1} \cap U(a_2)^{\varepsilon_2} \cap \ldots \cap U(a)^{\varepsilon_n}.$$

Using the above-mentioned conditions we can easily deduce that this component can be nonempty only if the following condition is satisfied: for each $i \in I$ there is exactly one descriptor $a_{j_i} \in A_i$ such that $\varepsilon_{j_i} = 1$ (i.e. $U(a_{j_i})^{\varepsilon_{j_i}} = U(a_{j_i})$). But then our component is the value of the simple term $\prod_{i \in I} a_{j_i}$. Notice that this implies that only $\prod_{i \in I} n_i$ components can be nonempty, though there are many more, namely

$$2^n = 2^{\sum_{i \in I} n_i} = \prod_{i \in I} 2^{n_i},$$

distinct sequences $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$. Now it is clear that coding the components by these sequences (in other words, using the components written as $S_{\mathfrak{M}}(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)$) would be utterly uneconomical. A much better coding method will be described in the next section.

THEOREM 7.4. *Let $\mathscr{S} = \langle X, A, R_I, U\rangle$ be an* i.s.r. *system.*

(a) *Different components in $\mathscr{S}$ are disjoint.*

(b) *The union of all components in $\mathscr{S}$ is X.*

(c) *Each set $Y \subseteq X$ describable in $\mathscr{S}$ is the disjoint union of all components in $\mathscr{S}$ contained in Y.*

We shall prove (c). $Y = \|t\|_{\mathscr{S}}$ for some t, hence

$$Y = \|\sum_{j \in J} t_j\|_{\mathscr{S}} = \bigcup_{j \in J} \|t_j\|_{\mathscr{S}}$$

where $\sum_{j \in J} t_j$ is the standard form of t. ■

It is easy to see that the Boolean algebra $\mathfrak{B}(\mathscr{S})$ is a subalgebra of $\mathscr{P}(X)$ generated by $\{U(a): a \in A\}$ and that components in $\mathscr{S}$ are its *atoms* (i.e., minimal elements in $\mathfrak{B}(\mathscr{S}) \setminus \{\emptyset\}$).

THEOREM 7.5. *A system $\mathscr{S} = \langle X, A, R_I, U\rangle$ is selective iff $\mathfrak{B}(\mathscr{S}) = \mathscr{P}(X)$* (recall that we consider only the case where I, and consequently A, are finite).

Proof. If $\mathfrak{B}(\mathscr{S}) = \mathscr{P}(X)$ then obviously $\mathscr{S}$ is selective. If $\mathscr{S}$ is selective then all $x \in X$ have different descriptions, and hence X is finite. For any $Y \subseteq X$ we thus have $Y = \|\sum_{y \in Y} t_y\|_{\mathscr{S}}$, i.e., Y is describable. ■

Notice that for infinite I (but a finitary language $\mathscr{L}_A$) it is easy to construct a selective system with an (infinite) indescribable set (by the cardinality argument).

THEOREM 7.6. *Let $Y \subseteq X$ be indescribable in $\mathscr{S} = \langle X, A, R_I, U\rangle$. Then there is a system $\mathscr{S}' = \langle X, A', R_{I'}, U'\rangle$ such that $\mathscr{S} \overset{*}{\subseteq} \mathscr{S}'$ and $Y \in \mathfrak{B}(\mathscr{S}')$.*

Proof. We give the names a, a' to Y and $X \setminus Y$, respectively, take $A' = A \cup \{a, a'\}$ and extend U to A' by putting $U'(a) = Y$ and $U'(a') = X \setminus Y$. $R_{I'}$ is defined in the natural way ($I' = I \cup \{i\}$ where the attribute i corresponds to the new descriptors). ■

It is clear that by the repeated application of the above procedure we may construct, for any system $\mathscr{S}$ (with a finite set of objects), a selective system $\mathscr{S}'$ such that $\mathscr{S} \overset{*}{\subseteq} \mathscr{S}'$. If the set of objects is infinite, we need an infinite number of "new" attributes.

Notice that if $\mathscr{S}_1 \subseteq \mathscr{S}$, say $\mathscr{S}_1 = \mathscr{S}|_Y$, then each component in $\mathscr{S}_1$ is the intersection of the corresponding component in $\mathscr{S}$ with Y. Thus, in general, fewer components are nonempty in $\mathscr{S}_1$ than in $\mathscr{S}$. If $\mathscr{S}_1 \overset{*}{\subseteq} \mathscr{S}$ then each component in $\mathscr{S}_1$ is the disjoint union of a certain number of components in $\mathscr{S}$. It is similarly when $\mathscr{S}_1 = \mathscr{S}/S$ (see Def. 5.4).

8. Implementation: general remarks

We now pass to implementational questions, more precisely to problems connected with the organization of the memory of a computer in which the objects of an i.s.r. system are stored.

Normal form theorems suggest the following implementational proposal: Objects belonging to the same component should be stored "together" in adjacent storage locations (this will be given a more precise meaning in the next section). Then each query entering the system can be transformed, by purely syntactical means, into

the standard form. The response of the system is then the disjoint union of the components indicated explicitly by this standard form.

Similarly, each query in the form of an assertion about our system can be transformed into an equivalent form built up from elementary formulas, each stating that a certain component is empty, by means of logical connectives. To determine the value (true or falsity) of the query we *need not retrieve* the components occurring in the transformed query, we only have to know, for each particular component, whether it is empty or not. Notice that here the standard form is not necessarily the best one. Rather, these are forms minimizing the number of occurrences of elementary formulas that are desirable.

When we implement an i.s.r. system using the method described above, it is necessary to enumerate the components. For each $i \in I$, let $|A_i| = n_i$. We may index the attributes, and the descriptors within each attribute, by nonnegative integers. Thus we may assume that $I = \{1, 2, \ldots, k\}$ and each component may be identified by a sequence $\langle b_1, b_2, \ldots, b_k\rangle$ where $0 \leqslant b_i < n_i$, for $i = 1, 2, \ldots, k$. Now we shall present a simple method of coding each such sequence by a single number a, $0 \leqslant a < n_1 \cdot n_2 \cdot \ldots \cdot n_k = N$.

We may assume that for each $i \in I$, $n_i \neq 1$ since otherwise the sequence $\langle b_1, b_2, \ldots, b_k\rangle$ would always have 0 at the ith position.

We define

$$
\begin{aligned}
u_1 &= n_2 \cdot n_3 \cdot \ldots \cdot n_k,\\
u_2 &= n_3 \cdot n_4 \cdot \ldots \cdot n_k,\\
&\ldots\ldots\ldots\ldots\\
u_j &= n_{j+1} \cdot n_{j+2} \cdot \ldots \cdot n_k,\\
&\ldots\ldots\ldots\ldots\\
u_{k-1} &= n_k,\\
u_k &= 1.
\end{aligned}
$$

Now we define our coding function $\varphi\colon C \to \{0, 1, \ldots, N-1\}$ where

$$C = \{\langle b_1, b_2, \ldots, b_k\rangle\colon (\forall i \in I)\ 0 \leqslant b_i < n_i\},$$

by the formula

$$\varphi(b_1, b_2, \ldots, b_k) = \sum_{i=1}^{k} b_i \cdot u_i.$$

Notice that the above formula strongly resembles the formula for the relative address of the element $A[b_1, b_2, \ldots, b_k]$ of a k-dimensional array declared in ALGOL 60 as

$$A[0\colon n_1-1, 0\colon n_2-1, \ldots, 0\colon n_k-1].$$

It can easily be proved that φ is a bijection and that for each a, $0 \leqslant a < N$,

$$\varphi^{-1}(a) = \langle b_1, b_2, \ldots, b_k\rangle$$

where

$$
\begin{aligned}
b_1 &= \text{entier}\left(\frac{a}{u_1}\right),\\
b_2 &= \text{entier}\left(\frac{a - b_1 \cdot u_1}{u_2}\right),\\
&\ldots\ldots\ldots\ldots\\
b_i &= \text{entier}\left(\frac{a - \sum_{j=1}^{i-1} b_j u_j}{u_i}\right).
\end{aligned}
$$

The formulas above provide an iterative procedure for obtaining the sequence $\langle b_1, \ldots, b_k\rangle$ from its code.

It is easy to see that the linear ordering $\preccurlyeq$ defined by

$$\langle b_1, b_2, \ldots, b_k\rangle \preccurlyeq \langle b'_1, b'_2, \ldots, b'_k\rangle \quad \text{iff} \quad \varphi(b_1, b_2, \ldots, b_k) \leqslant \varphi(b'_1, b'_2, \ldots, b'_k)$$

is the *lexicographic ordering*, i.e. $\langle b_1, b_2, \ldots, b_k\rangle \preccurlyeq \langle b'_1, b'_2, \ldots, b'_k\rangle$ if for some i, $b_1 = b'_1, b_2 = b'_2, \ldots, b_{i-1} = b'_{i-1}$, $b_i < b'_i$ or $\langle b_1, b_2, \ldots, b_k\rangle = \langle b'_1, b'_2, \ldots, b'_k\rangle$.

Notice that the method of coding described above is especially convenient when we increase the number of attributes.

Other methods of coding are also possible. For instance, we may replace each "weight" u_j by a $u'_j = 2^{k_j}$ such that $u_j \leqslant u'_j$. Then the binary representation of $\varphi(b_1, b_2, \ldots, b_k)$ is the concatenation of the binary representations of all b_i's. Thus, in using this method it is sufficient to know where "to slice" the code of $\langle b_1, b_2, \ldots, b_n\rangle$ to obtain the binary representations of the numbers $b_1, b_2, \ldots, b_k$.

9. f-graphs and admissible families of sets

In the preceding section we pointed out that objects belonging to the same component should be stored in "adjacent storage locations". Now we are going to make this statement more precise and to show how a suitable ordering of the components can make our organization more efficient.

Let us suppose that our storage consists of a set L of *storage locations*, and that in each storage location one object can be stored. In order to speak about "adjacent storage locations" we need an additional structure on L. We take as this structure a partial function $W\colon L \to L$ interpreted as follows: $W(l)$ is the storage location inspected immediately after l. Thus W reflects our retrieval method, which is, in turn, often determined by the physical structure of the storage media. However, sometimes it may be determined by software means. For instance, in the case of random-access memory, we can use chaining techniques: in each storage location l we store (apart from an object) a pointer indicating the storage location to be inspected immediately after l.

We say that the storage locations belonging to a set $K \subseteq L$ are *adjacent* if they form a segment, i.e. if there is an $l \in L$ such that

$$K = \{l, W(l), \ldots, W^{|K|-1}(l)\}$$

(where $W^0(l) = l$, $W^{n+1}(l) = W(W^n_{\mathbf{I}}(l))$).

Assume now that the objects of an i.s.r. system $\mathscr{S} = \langle X, A, R_I, U\rangle$ are stored in the memory according to a function $\varphi\colon L \underset{\text{onto}}{\to} X$ ($\varphi(l)$ is the object stored in the storage location l).

First we shall consider the case where φ is one-one, and then (in Section 15) we shall apply the theory developed for this particular case to the general case. In the first case it is convenient to identify each storage location with the object stored in it (and consequently W with $S = \varphi W \varphi^{-1}$).

Usually there is a class of singled out terms $\mathscr{H} \subseteq \mathscr{T}$ which are for some reasons important (e.g., there is a simple method for obtaining from their values the value of any other term—as it is in the case where $\mathscr{H} = \mathscr{T}_0$, the class of all simple terms). We are then interested in a file organization, i.e., in an arrangement of the objects in storage locations satisfying the following two conditions:

(i) There is no redundant storage of objects, each object is stored exactly once.

(ii) The value of each singled out term is a set consisting of objects stored in adjacent storage locations.

Such an organization enables us to retrieve the values of the singled out terms in a particularly easy way. Since the value of each singled out term forms a segment, we need only to know the locations of the first and the last objects of this segment. To each family of terms $\mathscr{H}$ there corresponds a family of subsets of X, $\mathfrak{M} = \{||t||_S\colon t \in \mathscr{H}\}$. Thus the problem of finding a file organization satisfying (i) and (ii) is equivalent to the following abstract combinatorial problem:

For a given family $\mathfrak{M} \subseteq \mathscr{P}(X)$ construct a partial function $S\colon X \to X$ (if there exists one) such that for each $M \in \mathfrak{M}$ there is an $x \in X$ such that $M = \{x, S(x), \ldots, S^{|M|-1}(x)\}$.

This problem and its modifications will be studied in the next few sections Now we shall give some more formal definitions.

Let X be a set and let $S \subseteq X \times X$ be a partial function such that $S(x) \neq x$ for each $x \in \mathscr{D}S$. It is convenient to treat $\langle X, S\rangle$ as a directed graph with the set of vertices X and the set of edges S. $\langle X, S\rangle$ will be referred to as an *f-graph* on X (S is a *successor function*, $S(x)$ is the *successor* of x). A set $B \subseteq X$ is a *segment* in $\langle X, S\rangle$ iff either $B = \emptyset$ or $B = \{x, S(x), \ldots, S^{|B|-1}(x)\}$ for some $x \in X$. Such an x is a *head* of B and $S^{|B|-1}(x)$ is the *end* corresponding to this head. If, in addition, $S^{|B|}(x) = x$ then B is a *cycle*. A segment is *proper* if it contains no cycles, *final* if $B \setminus \mathscr{D}S \neq \emptyset$ or $B = \emptyset$, and *initial* if $B \setminus \mathscr{R}S \neq \emptyset$ or $B = \emptyset$. If a segment $B \neq \emptyset$ is not a cycle, then its unique head and end are denoted by $h(B)$ and $e(B)$, respectively.

An f-graph $\langle X, S\rangle$ is *linear* if X is a final segment in it (i.e., if it consists of a single elementary path), *cyclic* if X is a cycle in it (i.e. if it consists of a single elementary cycle), *acyclic* if there is no cycle in it (in this case each of its connected components is easily seen to be a rooted tree—see Harary [7] for graph-theoretical notions used here). These types of f-graphs correspond to different types of storage, e.g., magnetic tape (linear f-graphs), drums, disks (cyclic f-graphs), random-access

memory organized by using chaining techniques (all the types of f-graphs including the general one). We denote by $\mathscr{F}(X)$ the class of all f-graphs on X and by $\mathscr{LF}(X)$, $\mathscr{CF}(X)$, $\mathscr{AF}(X)$ the subclasses of linear, cyclic and acyclic f-graphs, respectively. It is convenient to call cyclic also all f-graphs $\langle X, S\rangle$ with $|X| \leqslant 1$ (then $S = \emptyset$).

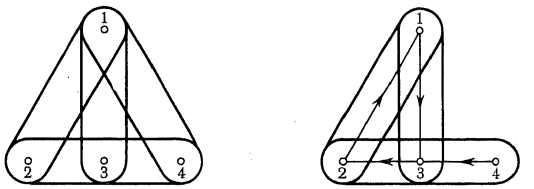

Fig. 1. An example of a non-admissible family and its admissible subfamily

We say that a family $\mathfrak{M}$ is *segmental* (*finally segmental*) over $\langle X, S\rangle$ if all $M \in \mathfrak{M}$ are segments (final segments) in $\langle X, S\rangle$. Our basic problem is that of the existence and construction, for a given family $\mathfrak{M}$, of an f-graph $\langle X, S\rangle$ such that $\mathfrak{M}$ is segmental over it. If $\mathfrak{M}$ admits such an f-graph then it is called *admissible*.

EXAMPLE 9.1. Let $X = \{1, 2, 3, 4\}$, $\mathfrak{M} = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3, 4\}\}$. It is easy to see that $\mathfrak{M}$ is not admissible and that each of its proper subfamilies, e.g. $\mathfrak{M} \setminus \{\{1, 4\}\}$, is admissible (see Fig. 1).

We may impose restrictions, in the definition of an admissible family, both on the type of the f-graph and on the type of segmentality. We then obtain different classes of families instead of the class of admissible families. These classes will be referred to as *classes of admissibility*.

DEFINITION 9.2. Let $\mathscr{E} \subseteq \mathscr{F}(X)$ be a class of f-graphs (e.g. $\mathscr{F}(X)$, $\mathscr{LF}(X)$, $\mathscr{CF}(X)$ or $\mathscr{AF}(X)$), and let $\mathscr{R} \subseteq \mathscr{P}(X) \times \mathscr{F}(X)$ be a relation, e.g. one of the relations $\mathscr{S}$, $\mathscr{S}^*$ defined as follows:

$$\langle M, G\rangle \in \mathscr{S} \Leftrightarrow M \text{ is a segment in } G,$$

$$\langle M, G\rangle \in \mathscr{S}^* \Leftrightarrow M \text{ is a final segment in } G.$$

We define a class $\mathfrak{c}(\mathscr{E}, \mathscr{R})$ of families of subsets of X as follows:

$$\mathfrak{M} \in \mathfrak{c}(\mathscr{E}, \mathscr{R}) \Leftrightarrow (\exists G \in \mathscr{E})\,(\forall M \in \mathfrak{M})\,\langle M, G\rangle \in \mathscr{R}.$$

The following classes of admissibility will play an important role in further considerations:

$\mathscr{A}dm(X) = \mathfrak{c}(\mathscr{F}(X), \mathscr{S})$ — the class of *admissible* families,

$\mathscr{L}(X) = \mathfrak{c}(\mathscr{LF}(X), \mathscr{S})$ — the class of *linear* families,

$\mathscr{L}^*(X) = \mathfrak{c}(\mathscr{LF}(X), \mathscr{S}^*)$ — the class of *finally linear* families,

$\mathscr{C}(X) = \mathfrak{c}(\mathscr{CF}(X), \mathscr{S})$ — the class of *cyclic* families,

$\mathscr{A}(X) = \mathfrak{c}(\mathscr{AF}(X), \mathscr{S})$ — the class of *acyclic* families,

$\mathscr{A}^*(X) = \mathfrak{c}(\mathscr{AF}(X), \mathscr{S}^*)$ — the class of *finally acyclic* families.

We say that $\langle X, S\rangle$ *realizes* the admissibility (linearity, etc.) of $\mathfrak{M}$ if $\langle X, S\rangle$ is an f-graph (linear f-graph, etc.) such that $\mathfrak{M}$ is segmental over it.

Let us observe that the class $\mathscr{L}(X)$ is related to the so-called $(0,1)$-*matrices with the consecutive* 1's *property* (see Fulkerson and Gross [3]). With every family $\mathfrak{M} = \{M_1, M_2, \ldots, M_n\} \subseteq \mathscr{P}(\{x_1, x_2, \ldots, x_m\})$ we may associate a $(0,1)$-matrix $\mathbf{A} = [a_{ij}]$ with m rows and n columns, called its *incidence matrix* and defined as follows:

$$a_{ij} = \begin{cases} 1 & \text{if} \quad x_i \in M_j, \\ 0 & \text{if} \quad x_i \notin M_j. \end{cases}$$

The matrix $\mathbf{A}$ is said to have the consecutive 1's property if there exists a permutation of its rows such that the permuted matrix has consecutive 1's in each column (see an example in Fig. 7 and Fig. 8, Section 13). It is easy to see that $\mathfrak{M} \in \mathscr{L}(X)$ iff the incidence matrix of $\mathfrak{M}$ has the consecutive 1's property.

There are some simple relations between different classes of admissibility:

LEMMA 9.3. (a) $\mathscr{A}dm(X)$ *contains each of the other classes of admissibility.*

(b) $\mathscr{L}(X) \subseteq \mathscr{C}(X) \cap \mathscr{A}(X)$; *the equality, in general, does not hold.*

(c) $\mathscr{L}^*(X) = \{\mathfrak{M} \subseteq \mathscr{P}(X)\colon (\forall M, N \in \mathfrak{M})\ M \subseteq N \vee N \subseteq M\}$, *i.e.,* $\mathscr{L}^*(X)$ *contains exactly the families which are linearly ordered by inclusion (nested).*

Proof. We prove (b). $\mathscr{L}(X) \subseteq \mathscr{A}(X)$ since each linear f-graph is acyclic. $\mathscr{L}(X) \subseteq \mathscr{C}(X)$ since we may always add the edge $\langle e(X), h(X)\rangle$ to an f-graph $\langle X, S\rangle$ realizing the linearity of $\mathfrak{M}$, obtaining an f-graph realizing the cyclicity of $\mathfrak{M}$. An example (due to M. Wąsowska) of a family which is cyclic and acyclic but not linear is shown in Fig. 2.

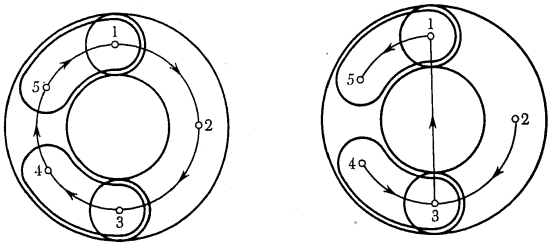

Fig. 2. An example of a family $\mathfrak{M} \in (\mathscr{C}(X) \cap \mathscr{A}(X)) \setminus \mathscr{L}(X)$ $(X = \{1, 2, 3, 4, 5\}$, $\mathfrak{M} = \{\{1, 2, 3\}, \{1, 3, 4, 5\}, \{1, 5\}, \{3, 4\}\})$

DEFINITION 9.4. (a) Let $\mathfrak{M} \subseteq \mathscr{P}(X)$ and let $C \subseteq X$. The *trace* of $\mathfrak{M}$ on C (in symbols $\mathfrak{M}|_C$) is a family of subsets of C defined as follows:

$$\mathfrak{M}|_C = \{M \cap C\colon M \in \mathfrak{M}\}.$$

(b) Let $\langle X, S\rangle$ be an f-graph and let $C \subseteq X$. The *contraction* of $\langle X, S\rangle$ to C (in symbols $\langle X, S\rangle|_C$) is an f-graph on C defined as follows:

$$\langle X, S\rangle|_C = \langle C, S_C\rangle,$$

$$\mathscr{D}S_C = \{y \in C\colon (\exists k > 0)\, S^k(y) \in C \wedge S^k(y) \neq y\},$$

$$S_C(x) = S^{k_x}(x) \quad \text{for each } x \in \mathscr{D}S_C, \text{ where } k_x = \min\{k\colon k > 0 \wedge S^k(x) \in C\}.$$

Roughly speaking, we take as $S_C(x)$ the first vertex belonging to C (different from x) encountered on the path beginning at x. The operations of trace and contraction are used to describe the situation which arises when some objects are deleted from an i.s.r. system. The adequacy of these operations for our purposes is confirmed by the following

THEOREM 9.5. *Let $\langle X, S\rangle$ realize the admissibility of a family $\mathfrak{M} \subseteq \mathscr{P}(X)$ and let $C \subseteq X$. Then $\langle X, S\rangle|_C$ realizes the admissibility of $\mathfrak{M}|_C$. The same holds for the classes $\mathscr{L}(X)$, $\mathscr{C}(X)$, $\mathscr{A}(X)$, $\mathscr{L}^*(X)$, $\mathscr{A}^*(X)$ taken instead of $\mathscr{A}dm(X)$.*

Proof. It follows from the fact, which is easy to establish, that the contraction changes neither the type of an f-graph nor the type of segmentality. ■

Thus admissibility (linearity, etc.) of a family is a property inherited by its traces on arbitrary sets. Using the operation of trace we can formulate the following "local characterization" of finally acyclic families:

THEOREM 9.10. *Let $\mathfrak{M} \subseteq \mathscr{P}(X)$. Then $\mathfrak{M} \in \mathscr{A}^*(X)$ iff for each $M \in \mathfrak{M}$, $\mathfrak{M}|_M \in \mathscr{L}^*(M)$.*

We leave the proof to the reader. ■

10. Admissibility and components

In this section we shall show that the admissibility (linearity, etc.) of a family $\mathfrak{M}$ does not depend on the cardinalities of the components of $\mathfrak{M}$, it only depends on which components are nonempty. This observation enables us to restrict ourselves to the families with each nonempty component consisting of one element.

DEFINITION 10.1. (a) Let $\mathfrak{M} = \{M_1, M_2, \ldots, M_n\} \subseteq \mathscr{P}(X)$. We define, for $i = 1, 2, \ldots, n$,

$$Z(M_i) = \{\langle \varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\rangle \in \{0,1\}^n\colon \varepsilon_i = 1 \wedge S_{\mathfrak{M}}(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n) \neq \emptyset\},$$

$$\mathscr{Z}(\mathfrak{M}) = \{Z(M_1), Z(M_2), \ldots, Z(M_n)\}.$$

(b) Two families, $\mathfrak{M} \subseteq \mathscr{P}(X)$ and $\mathfrak{N} \subseteq \mathscr{P}(Y)$, are *isomorphic* if there is a bijection $\varphi\colon X \to Y$ such that

$$(\forall M \in \mathfrak{M})\, \varphi(M) \in \mathfrak{N} \wedge (\forall N \in \mathfrak{N})\, \varphi^{-1}(N) \in \mathfrak{M}.$$

(c) Two families $\mathfrak{M}$ and $\mathfrak{N}$ are *similar* if the families $\mathscr{Z}(\mathfrak{M})$ and $\mathscr{Z}(\mathfrak{N})$ are isomorphic.

Notice that $Z(M_i)$ can be treated as a set of rows of the incidence matrix of $\mathfrak{M}$. It is easy to see that each nonempty component of $\mathscr{Z}(\mathfrak{M})$ consists of one element and that $\mathfrak{M}$ and $\mathscr{Z}(\mathfrak{M})$ are similar (since $\mathscr{Z}(\mathfrak{M}) = \mathscr{Z}(\mathscr{Z}(\mathfrak{M}))$).

THEOREM 10.2. *Let the families $\mathfrak{M} \subseteq \mathscr{P}(X)$ and $\mathfrak{N} \subseteq \mathscr{P}(Y)$ be similar. Then*

$$\mathfrak{M} \in \mathscr{K}(X) \Leftrightarrow \mathfrak{N} \in \mathscr{K}(Y)$$

where $\mathscr{K}(X)$ is one of the classes $\mathscr{A}dm(X)$, $\mathscr{L}(X)$, $\mathscr{C}(X)$, $\mathscr{A}(X)$, $\mathscr{L}^*(X)$, $\mathscr{A}^*(X)$.

Proof. The theorem is obvious if we replace similarity by isomorphism. Thus it is sufficient to prove the theorem for $\mathfrak{N} = \mathscr{Z}(\mathfrak{M})$. To this end let us take a set $C \subseteq X$ such that for each $S \in \mathscr{S}(\mathfrak{M})$, $|C \cap S| = 1$ (recall that $\mathscr{S}(\mathfrak{M})$ is the family of all nonempty components of $\mathfrak{M}$; see Def. 7.2). It is easy to see that $\mathfrak{M}|_C$ and $\mathscr{Z}(\mathfrak{M})$ are isomorphic. Thus, if $\mathfrak{M} \in \mathscr{K}(X)$ then, by Theorem 9.5, $\mathfrak{M}|_C \in \mathscr{K}(C)$ and consequently $\mathscr{Z}(\mathfrak{M}) \in \mathscr{K}(Y)$. Conversely, if $\mathscr{Z}(\mathfrak{M}) \in \mathscr{K}(Y)$ then we apply the following procedure to an f-graph realizing the admissibility (of class $\mathscr{K}(Y)$) of $\mathscr{Z}(\mathfrak{M})$: we replace its vertices by the corresponding components of $\mathfrak{M}$, as shown in Fig. 3. ■

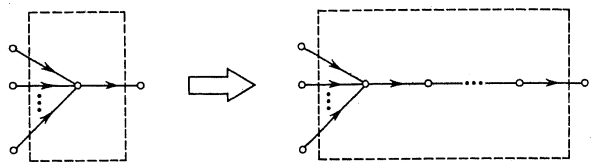

Fig. 3. The procedure of "pushing up" vertices used in the proof of Theorem 10.2

Theorem 10.2 has one important consequence. Let us notice that if $\mathscr{Z}(\mathfrak{M})$ is admissible (linear, cyclic or acyclic) then $\mathscr{Z}(\mathfrak{M}) \cup \mathscr{S}(\mathscr{Z}(\mathfrak{M}))$ is in the same class of admissibility, since each component of $\mathscr{Z}(\mathfrak{M})$ consists of one element. But $\mathscr{Z}(\mathfrak{M}) \cup \mathscr{S}(\mathscr{Z}(\mathfrak{M})) = \mathscr{Z}(\mathfrak{M} \cup \mathscr{S}(\mathfrak{M}))$, thus we have the following

THEOREM 10.3. *Let $\mathscr{K}(X)$ be one of the classes $\mathscr{A}dm(X)$, $\mathscr{L}(X)$, $\mathscr{C}(X)$, $\mathscr{A}(X)$. Then*

$$\mathfrak{M} \in \mathscr{K}(X) \Leftrightarrow \mathfrak{M} \cup \mathscr{S}(\mathfrak{M}) \in \mathscr{K}(X). \blacksquare$$

In the case of $\mathscr{L}^*(X)$ and $\mathscr{A}^*(X)$ we may only require that the components be segments, not necessarily final. Usually we shall consider only the "regular" f-graphs realizing the admissibility (linearity, etc.) of a family, i.e., f-graphs in which all the components are segments.

Admissibility of a family imposes severe restrictions on the number of its nonempty components. An arbitrary family $\mathfrak{M}$ can have 2^n ($n = |\mathfrak{M}|$) nonempty components, but we have the following theorem:

THEOREM 10.4. *Let $\mathfrak{M} \subseteq \mathscr{P}(X)$ and let $|\mathfrak{M}| = n$.*

(a) *If $\mathfrak{M} \in \mathscr{L}^*(X)$ then $|\mathscr{S}(\mathfrak{M})| \leqslant n+1$.*

(b) *If $\mathfrak{M} \in \mathscr{L}(X)$ then $|\mathscr{S}(\mathfrak{M})| \leqslant 2n$.*

(c) *If $\mathfrak{M} \in \mathscr{C}(X)$ then $|\mathscr{S}(\mathfrak{M})| \leqslant 2n$.*

(d) *If $\mathfrak{M} \in \mathscr{A}^*(X)$ then $|\mathscr{S}(\mathfrak{M})| \leqslant 2n$.*

(e) *If $\mathfrak{M} \in \mathscr{A}(X)$ then $|\mathscr{S}(\mathfrak{M})| \leqslant 3n-2$, for $n > 1$.*

(f) *If $\mathfrak{M} \in \mathscr{A}dm(X)$ then $|\mathscr{S}(\mathfrak{M})| \leqslant 3n+1$, for $n > 3$,*

and $|\mathscr{S}(\mathfrak{M})| \leqslant 8$ for $n = 3$. ■

All these bounds are reached by some families. For the proof the reader is referred to [11]. Here we show only, as an example, an admissible family with

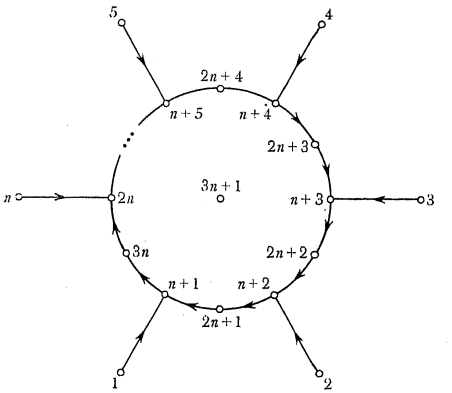

Fig. 4. An admissible family with $3n+1$ nonempty components

$3n+1$ nonempty components (see Fig. 4): $X = \{1, 2, \ldots, 3n+1\}$, $\mathfrak{M} = \{M_1, M_2, \ldots, M_n\}$ and for each i, $|M_i| = 6$ and $h(M_i) = i$; each component consists of one element.

It should be emphasized that Theorem 10.4 gives only necessary conditions for a family to be in certain classes of admissibility, these conditions not being sufficient.

11. Families of three sets

In this section we study the admissibility of "small" families. In virtue of Theorem 10.2 we may assume that all the families $\mathfrak{M} \subseteq \mathscr{P}(X)$ under consideration satisfy the following conditions:

(i) Each component of $\mathfrak{M}$ either consists of one element or is empty (for other families we "push up" the components).

(ii) The set of nonempty components of $\mathfrak{M}$ is the maximal possible for a class of admissibility under consideration (for other families we obtain appropriate f-graphs by contraction).

For convenience we assume also that $\bigcup \mathfrak{M} = X$. For a given $\mathfrak{M} \subseteq \mathscr{P}(X)$, two f-graphs $\langle X, S_1 \rangle$ and $\langle X, S_2 \rangle$ are said to be *essentially different* if there is no bijection $\varphi: X \to X$ satisfying the following two conditions:

(i) $(\forall x, y \in X)\big(S_1(x) = y \Leftrightarrow S_2(\varphi(x)) = \varphi(y)\big)$,

(ii) $(\forall M \in \mathfrak{M})\, \varphi(M) \in \mathfrak{M}$.

It is easy to see that each family consisting of two sets is linear and finally acyclic. The case $\mathfrak{M} = \{M_1, M_2, M_3\}$ is less trivial.

THEOREM 11.1. (See [17].) *Each family $\mathfrak{M}$ consisting of three sets is admissible The unique four essentially different f-graphs realizing the admissibility of $\mathfrak{M}$ are depicted in Fig. 5.* ■

For other classes of admissibility we have the following theorems:

THEOREM 11.2. *A family $\{M_1, M_2, M_3\}$ is acyclic iff*

$$\bar{M}_1 \cap M_2 \cap M_3 = \emptyset \vee M_1 \cap \bar{M}_2 \cap M_3 = \emptyset \vee M_1 \cap M_2 \cap \bar{M}_3 = \emptyset$$

($\bar{M}_i$ denotes $X \setminus M_i$), i.e., iff one of the sets M_1, M_2, M_3 contains the intersection of the other two. ■

THEOREM 11.3. *A family $\{M_1, M_2, M_3\}$ is cyclic iff at least one of the following two conditions is satisfied:*

(i) $M_1 \cap M_2 \cap M_3 = \emptyset$.

(ii) $M_1 \cap \bar{M}_2 \cap \bar{M}_3 = \emptyset \vee \bar{M}_1 \cap M_2 \cap \bar{M}_3 = \emptyset \vee \bar{M}_1 \cap \bar{M}_2 \cap M_3 = \emptyset$. ■

Notice that (ii) is equivalent to the fact that one of the sets M_1, M_2, M_3 is contained in the union of the other two.

THEOREM 11.4. *A family $\{M_1, M_2, M_3\}$ is linear iff at least one of the following three conditions is satisfied:*

(i) $M_i \subseteq M_j$ *for some* $i, j \in \{1, 2, 3\}$, $i \neq j$.

(ii) $M_i \cap M_j = \emptyset$ *for some* $i, j \in \{1, 2, 3\}$, $i \neq j$.

(iii) $M_i \cap M_j \subseteq M_k \subseteq M_i \cup M_j$ *for some* $i, j, k \in \{1, 2, 3\}$, $i \neq j, j \neq k, k \neq i$. ■

More details and the proofs of the above theorems can be found in [14].

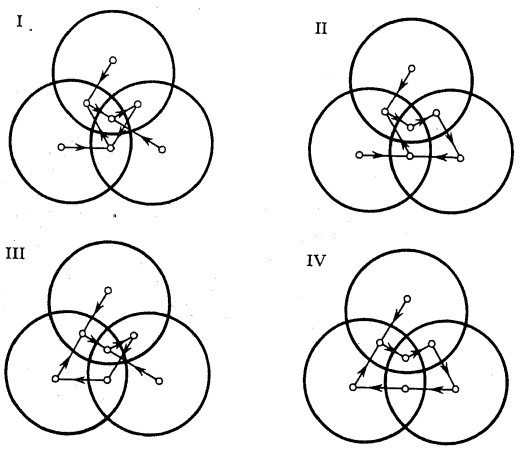

Fig. 5. The f-graphs realizing the admissibility of a family of three sets

12. Equivalent families

Let us recall that our basic problem is to construct, for a given $\mathfrak{M} \subseteq \mathscr{P}(X)$, an f-graph realizing the admissibility (of a specified class) of $\mathfrak{M}$. If the structure of $\mathfrak{M}$ is complicated then our problem becomes very difficult. We can then apply the following method. We try to transform $\mathfrak{M}$ into a certain family $\mathfrak{M}'$ such that $\mathfrak{M}'$ has a simple structure allowing an easy solution of our problem, and $\mathfrak{M}'$ is "equivalent" to $\mathfrak{M}$ in the sense that each f-graph realizing the admissibility of $\mathfrak{M}'$ realizes the admissibility of $\mathfrak{M}$ and conversely. This method will be exploited in the cases of linear and cyclic families. Now we shall describe it more precisely. For the sake of simplicity we shall restrict ourselves to the class $\mathscr{L}(X)$.

Let us define for any $\mathfrak{M} \subseteq \mathscr{P}(X)$ and $\mathscr{G} \subseteq \mathscr{LF}(X)$,

$$\mathfrak{M}^* = \{G \in \mathscr{LF}(X) \colon (\forall M \in \mathfrak{M}) \langle M, G \rangle \in \mathscr{S}\},$$
$$\mathscr{G}^* = \{M \in \mathscr{P}(X) \colon (\forall G \in \mathscr{G}) \langle M, G \rangle \in \mathscr{S}\}.$$

$\mathfrak{M}^*$ is the set of all f-graphs realizing the linearity of $\mathfrak{M}$, and $\mathscr{G}^*$ is the greatest family segmental over each $G \in \mathscr{G}$. The mappings $\mathfrak{M} \mapsto \mathfrak{M}^*$ and $\mathscr{G} \mapsto \mathscr{G}^*$ have the following properties:

(i) $\mathfrak{M} \subseteq \mathfrak{N} \Rightarrow \mathfrak{M}^* \supseteq \mathfrak{N}^*$, $\quad \mathscr{G} \subseteq \mathscr{H} \Rightarrow \mathscr{G}^* \supseteq \mathscr{H}^*$,

(ii) $\mathfrak{M} \subseteq \mathfrak{M}^{**}$, $\quad \mathscr{G} \subseteq \mathscr{G}^{**}$,

(iii) $\mathfrak{M}^{***} = \mathfrak{M}^*$, $\quad \mathscr{G}^{***} = \mathscr{G}^*$,

being an example of a *Galois connection* (see Cohn [2]).

DEFINITION 12.1. (a) We define an operator $\boldsymbol{D}_{\mathscr{L}} \colon \mathscr{P}(\mathscr{P}(X)) \to \mathscr{P}(\mathscr{P}(X))$ as follows:

$$\boldsymbol{D}_{\mathscr{L}}(\mathfrak{M}) = \mathfrak{M}^{**} \quad \text{for each } \mathfrak{M} \subseteq \mathscr{P}(X).$$

(b) M is *$\mathscr{L}$-dependent* on $\mathfrak{M}$ if $M \in \boldsymbol{D}_{\mathscr{L}}(\mathfrak{M})$.

(c) $\mathfrak{M}$ and $\mathfrak{N}$ are *$\mathscr{L}$-equivalent* (in symbols $\mathfrak{M} \sim_{\mathscr{L}} \mathfrak{N}$) if $\mathfrak{M}^* = \mathfrak{N}^*$.

Now let us examine the interpretation of the above notions. $\boldsymbol{D}_{\mathscr{L}}(\mathfrak{M})$ contains exactly these subsets of X which are "forced" to be segments in each f-graph realizing the linearity of $\mathfrak{M}$. M is $\mathscr{L}$-dependent on $\mathfrak{M}$ iff M is a segment in each f-graph realizing the linearity of $\mathfrak{M}$. For instance, if $M, N \in \mathfrak{M}$ and $M \cap N \neq \emptyset$, then $M \cup N$ is $\mathscr{L}$-dependent on $\mathfrak{M}$. $\mathfrak{M} \sim_{\mathscr{L}} \mathfrak{N}$ iff the same f-graphs realize the linearity of $\mathfrak{M}$ and $\mathfrak{N}$.

It is clear that the same considerations can be carried out if any other class of admissibility $\mathscr{K}(X)$ is taken instead of $\mathscr{L}(X)$. We then obtain the notions of $\mathscr{K}$-dependence, $\mathscr{K}$-equivalence, and the operator $\boldsymbol{D}_{\mathscr{K}}$.

The following simple lemmas result from the general properties of Galois connections (see Cohn [2]).

LEMMA 12.2. *$\boldsymbol{D}_{\mathscr{K}}$ is a closure operator, i.e., for each $\mathfrak{M}, \mathfrak{N} \subseteq \mathscr{P}(X)$*

(i) $\mathfrak{M} \subseteq \mathfrak{N} \Rightarrow \boldsymbol{D}_{\mathscr{K}}(\mathfrak{M}) \subseteq \boldsymbol{D}_{\mathscr{K}}(\mathfrak{N})$.

(ii) $\mathfrak{M} \subseteq \boldsymbol{D}_{\mathscr{K}}(\mathfrak{M})$.

(iii) $\boldsymbol{D}_{\mathscr{K}}(\boldsymbol{D}_{\mathscr{K}}(\mathfrak{M})) = \boldsymbol{D}_{\mathscr{K}}(\mathfrak{M})$. ■

Notice that in general $D_{\mathscr{K}}(\emptyset) \neq \emptyset$ and $D_{\mathscr{K}}(\mathfrak{M} \cup \mathfrak{N}) \neq D_{\mathscr{K}}(\mathfrak{M}) \cup D_{\mathscr{K}}(\mathfrak{N})$.

LEMMA 12.3. *The following conditions are equivalent*:

(i) $\mathfrak{M} \sim_{\mathscr{K}} \mathfrak{N}$.

(ii) $D_{\mathscr{K}}(\mathfrak{M}) = D_{\mathscr{K}}(\mathfrak{N})$.

(iii) *Each* $M \in \mathfrak{M}$ *is* $\mathscr{K}$*-dependent on* $\mathfrak{N}$ *and each* $N \in \mathfrak{N}$ *is* $\mathscr{K}$*-dependent on* $\mathfrak{M}$. ■

DEFINITION 12.4. Two sets M, N *overlap* (in symbols $M \between N$) if

$$M \cap N \neq \emptyset \wedge M \setminus N \neq \emptyset \wedge N \setminus M \neq \emptyset.$$

The following simple lemma enables us to find some (in general not all) sets dependent on a family.

LEMMA 12.5. *Let* $M, N \in \mathfrak{M} \subseteq \mathscr{P}(X)$ *and let* $x \in X$. *Then*

(i) $\emptyset$ *and* $\{x\}$ *are* $\mathscr{A}dm$*-dependent on* $\mathfrak{M}$.

(ii) $\emptyset$, $\{x\}$ *and* X *are* $\mathscr{L}$*-dependent on* $\mathfrak{M}$.

If $M \between N$ *then* $M \cup N$, $M \cap N$, $M \setminus N$, $N \setminus M$ *are* $\mathscr{L}$*-dependent on* $\mathfrak{M}$.

(iii) $\emptyset$, $\{x\}$, X *and* $X \setminus \{x\}$ *are* $\mathscr{C}$*-dependent on* $\mathfrak{M}$. $X \setminus M$ *is* $\mathscr{C}$*-dependent on* $\mathfrak{M}$.

If $M \between N \wedge M \cup N \neq X$ *then* $M \cup N$, $M \cap N$, $M \setminus N$, $N \setminus M$ *are* $\mathscr{C}$*-dependent on* $\mathfrak{M}$.

(iv) $\emptyset$ *and* $\{x\}$ *are* $\mathscr{A}$*-dependent on* $\mathfrak{M}$. $M \cap N$ *is* $\mathscr{A}$*-dependent on* $\mathfrak{M}$. ■

As an example of an application of the above lemma we shall consider the following theorem.

THEOREM 12.6. *Let* $\mathfrak{M} \subseteq \mathscr{P}(X)$ *and let* x_0 *be a fixed element of* X. *Define*

$$\mathfrak{M}' = \{M^{\varepsilon(M)} \colon M \in \mathfrak{M}\}$$

where

$$\varepsilon(M) = \begin{cases} 1 & \text{if} \quad x_0 \notin M, \\ 0 & \text{if} \quad x_0 \in M \end{cases}$$

(*recall that* $M^1 = M$, $M^0 = X \setminus M$). *Then the families* $\mathfrak{M}$ *and* $\mathfrak{M}'$ *are* $\mathscr{C}$*-equivalent. Moreover,* $\mathfrak{M}$ *is cyclic iff* $\mathfrak{M}'$ *is linear.*

Proof. Every $M \in \mathfrak{M}'$ is $\mathscr{C}$-dependent on $\mathfrak{M}$ since $M \in \mathfrak{M}$ or $X \setminus M \in \mathfrak{M}$. Similarly, every $N \in \mathfrak{M}$ is $\mathscr{C}$-dependent on $\mathfrak{M}'$. By Lemma 12.3, $\mathfrak{M}' \sim_{\mathscr{C}} \mathfrak{M}$. Suppose that $\mathfrak{M}$ is cyclic. Then $\mathfrak{M}'$ is cyclic, and consequently linear. Indeed, $x_0 \notin \bigcup \mathfrak{M}$

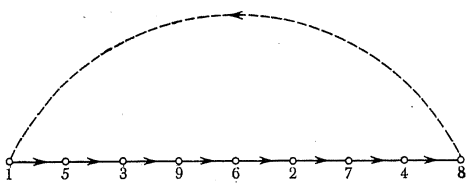

Fig. 6. Constructing an f-graph for a given family of sets

and the edge $\langle x_0, S(x_0)\rangle$ can be deleted from any f-graph $\langle X, S\rangle$ realizing the cyclicity of $\mathfrak{M}'$. Conversely, if $\mathfrak{M}'$ is linear, then it is cyclic and, since $\mathfrak{M} \sim_{\mathscr{C}} \mathfrak{M}'$, $\mathfrak{M}$ is cyclic. ■

The theorem above provides a simple method of reducing the problem of finding an f-graph realizing the cyclicity of a family to an analogous problem for the linear case.

EXAMPLE 12.7. Let $X = \{1, 2, \ldots, 9\}$ and let $\mathfrak{M} = \{\{1, 4, 5, 8\}, \{2, 3, 5, 6, 9\}, \{2, 4, 7\}, \{2, 4, 6, 7, 8\}\}$. We take $x_0 = 8$ and we obtain $\mathfrak{M}' = \{\{2, 3, 6, 7, 9\}, \{2, 3, 5, 6, 9\}, \{2, 4, 7\}, \{1, 3, 5, 9\}\}$. Then we construct an f-graph $\langle X, S\rangle$ realizing the linearity of $\mathfrak{M}'$ (for a method of construction of such an f-graph see the next section) and we add the edge $\langle e(X), h(X)\rangle = \langle 8, 1\rangle$ (see Fig. 6). The resulting f-graph realizes the cyclicity of $\mathfrak{M}$.

13. Algorithms for constructing an f-graph for a given family of sets

In this section we give algorithms for constructing f-graphs for families of classes $\mathscr{L}^*(X)$, $\mathscr{L}(X)$, $\mathscr{C}(X)$ and $\mathscr{A}^*(X)$.

The case $\mathfrak{M} \in \mathscr{L}^*(X)$ is trivial. Indeed, $\mathfrak{M}$ is then linearly ordered by $\subseteq$. We find a minimal (with respect to $\subseteq$) set in $\mathfrak{M}$, say M_1, and we order it linearly (by a successor function). Then we find a minimal set in $\mathfrak{M} \setminus \{M_1\}$, say M_2, and we add $M_2 \setminus M_1$ at the beginning of our ordering, and so on. After a finite number of steps we obtain a linear f-graph such that $\mathfrak{M}$ is finally segmental over it (or, if $M_i \nsubseteq M_{i+1}$ for some i, we find out that $\mathfrak{M} \notin \mathscr{L}^*(X)$).

In order to deal with the case $\mathfrak{M} \in \mathscr{L}(X)$, we shall need a decomposition theorem, which is based on an idea of Fulkerson and Gross [3]. First we shall give some auxiliary definitions.

DEFINITION 13.1. Let $\mathfrak{M} \subseteq \mathscr{P}(X)$.

(a) The *overlap graph* $\mathscr{O}(\mathfrak{M})$ is a nondirected graph with $\mathfrak{M}$ as the set of vertices, two vertices joined by an edge iff $M \between N$.

(b) A family $\mathscr{B} \subseteq \mathfrak{M}$ is a *block* of $\mathfrak{M}$ if $\mathscr{B}$ is the set of vertices of a component of $\mathscr{O}(\mathfrak{M})$. (For graph-theoretical notions see Harary [7].)

(c) We define a partial ordering $\leqslant$ on the set of blocks of $\mathfrak{M}$ as follows:

$$\mathscr{B}_1 \leqslant \mathscr{B}_2 \Leftrightarrow (\exists M_1 \in \mathscr{B}_1)(\exists M_2 \in \mathscr{B}_2)\, M_1 \subseteq M_2.$$

(d) A block $\mathscr{B}$ of $\mathfrak{M}$ is *maximal* if it is a maximal element in the ordering $\leqslant$, i.e., there is no block $\mathscr{B}' > \mathscr{B}$.

THEOREM 13.2 (*Decomposition Theorem*). (a) *A family* $\mathfrak{M}$ *is linear iff all its blocks are linear.*

(b) *A family* $\mathfrak{M} \subseteq \mathscr{P}(X)$, *such that* $X \notin \mathfrak{M}$, *is cyclic in exactly two cases*:

(i) *All the blocks of* $\mathfrak{M}$ *are linear* (*and consequently* $\mathfrak{M}$ *is linear*).

(ii) *There is exactly one maximal block of* $\mathfrak{M}$, *and it is cyclic. The other blocks are linear.*

(c) *A family* $\mathfrak{M}$ *is acyclic iff all its maximal blocks are acyclic and all the other blocks are linear.*

Proof. The necessity of the above conditions is obvious. For the sufficiency let us notice that if $\mathscr{B}_1 < \mathscr{B}_2$ then $\bigcup \mathscr{B}_1$ is contained in a component of $\mathscr{B}_2$. The

ordering of elements within a component is, as we noticed in Section 10, immaterial. Thus we can proceed as follows. We take an f-graph realizing the linearity (cyclicity, acyclicity) of the maximal blocks. Then we modify it to render the blocks of "depth" two (i.e., the immediate $\leqslant$-predecessors of the maximal blocks) segmental. Then, if necessary, we modify the ordering within the components of blocks of "depth" two, in such a way that the blocks of "depth" three are segmental, and so on. ■

Now we are going to describe an algorithm for constructing an f-graph realizing the linearity of a family of sets. In virtue of the Decomposition Theorem we may restrict ourselves to the case where our family, $\mathfrak{M} \subseteq \mathscr{P}(X)$, consists of one block. Without loss of generality we may also assume that $\bigcup \mathfrak{M} = X$. The algorithm constructs a family $\mathfrak{N} \in \mathscr{L}^*(X)$ such that each f-graph realizing the final linearity of $\mathfrak{N}$ realizes the linearity of $\mathfrak{M}$ and conversely (for more details see [11]).

Step 1. We find a minimal "connected covering" $\mathfrak{M}_0 \subseteq \mathfrak{M}$, i.e., a $\subseteq$-minimal $\mathfrak{M}' \subseteq \mathfrak{M}$ such that

(i) $\bigcup \mathfrak{M}' = X$,

(ii) $\mathscr{O}(\mathfrak{M}')$ (see Def. 13.1(a)) is connected.

To this end we check, for each $M \in \mathfrak{M}$, whether $\mathfrak{M} \setminus \{M\}$ satisfies (i) and (ii) and, if so, delete M from $\mathfrak{M}$. The resulting family $\mathfrak{M}_0$ is a minimal "connected covering" and $\mathscr{O}(\mathfrak{M}_0)$ has the form of an elementary path (provided that $\mathfrak{M} \in \mathscr{L}(X)$). Let M_0 be one of its endpoints. It is easy to see that M_0 is a final or initial segment in each f-graph realizing the linearity of $\mathfrak{M}$.

Step 2. Given the set M_0, we can produce other final segments (we can restrict ourselves to the case where they are final, since $\langle X, S\rangle$ can be replaced, if necessary, by $\langle X, S^{-1}\rangle$). More specifically, we construct a family $\mathfrak{N}$—the least family $\mathfrak{N}' \subseteq \mathscr{P}(X)$ such that

(i) $M_0 \in \mathfrak{N}'$;

(ii) if $M \in \mathfrak{N}' \wedge N \in \mathfrak{M} \wedge M \cap N \neq \emptyset \wedge N \setminus M \neq \emptyset$ then $M \cup N$, $M \setminus N \in \mathfrak{N}'$.

It is easy to see that $\mathfrak{N} \in \mathscr{L}^*(X)$ (if $\mathfrak{M} \in \mathscr{L}(X)$) and that for each $M \in \mathfrak{M}$ there are two sets $N_1, N_2 \in \mathfrak{N}$ such that $M = N_2 \setminus N_1$ (recall that $\mathfrak{M}$ is a block). It is easily seen that each f-graph realizing the final linearity of $\mathfrak{N}$ realizes the linearity of $\mathfrak{M}$, and conversely.

Step 3. We construct an f-graph realizing the final linearity of $\mathfrak{N}$.

EXAMPLE 13.3. Let $X = \{1, 2, \ldots, 10\}$, $\mathfrak{M} = \{\{1, 2, 4\}, \{1, 2, 3, 4, 7, 9\}, \{2, 5, 9, 10\}, \{9, 10\}, \{1, 7\}, \{3, 6, 8\}, \{1, 3, 4, 6, 7, 8\}\}$. Step 1 gives $\mathfrak{M}_0 = \{\{1, 2, 3, 4, 7, 9\}, \{2, 5, 9, 10\}, \{3, 6, 8\}\}$. We take $M_0 = \{2, 5, 9, 10\}$. The whole construction is shown in Fig. 7.

The algorithm was implemented in PL/1 and was run on an IBM/370 (Model 145) computer. For the example shown in Fig. 8 and Fig. 9 ($|X| = 50$, $|\mathfrak{M}| = 100$) the execution time was 19.25 s. The output of the program consists of two incidence matrices of $\mathfrak{M}$, the second one with rows permuted in such a way that it has consecutive 1's in each column (see Section 9).

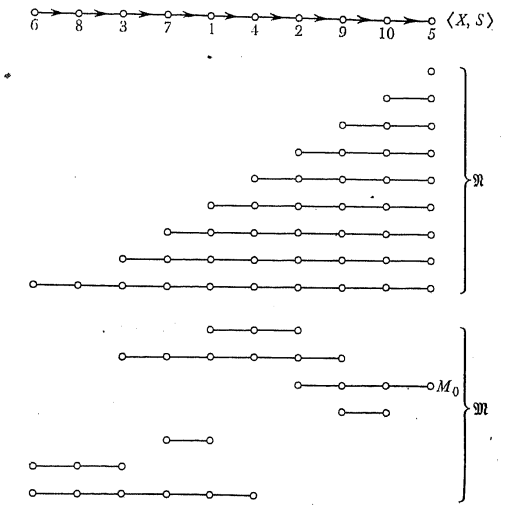

Fig. 7. Constructing an f-graph for a linear family of sets (see Example 13.3)

The program is also capable of constructing an f-graph realizing the cyclicity of a family of sets. The method described in Section 12 (see Theorem 12.6) is then used.

For the case $\mathfrak{M} \in \mathscr{A}^*(X)$ the construction of the appropriate f-graph is as follows. Let $\mathfrak{M} = \{M_1, M_2, \ldots, M_n\}$. We construct the families

$$\mathfrak{M}|_{M_1}, \mathfrak{M}|_{M_2 \setminus M_1}, \mathfrak{M}|_{M_3 \setminus (M_1 \cup M_2)}, \ldots, \mathfrak{M}|_{M_n \setminus (M_1 \cup M_2 \cup \cdots \cup M_{n-1})}.$$

These families are finally linear and have disjoint unions. We construct for each of them an f-graph realizing its final linearity. Then we glue them together by adding appropriate edges.

14. Augmentation of a family

The considerations of this section are motivated by problems arising in updating i.s.r. systems. There are two kinds of updating: one can change either the set of singled out terms (queries), or the set of objects. This leads to the following question: How should the appropriate f-graph be modified to remain good for the modified family of sets? All the theorems of this section are constructive, that is, they provide an algorithm for performing such a modification of an f-graph.

The situation is simple when a set of our family is deleted (we need not change the f-graph), or when an object $x \in X$ is deleted (we take the contraction $\langle X, S\rangle|_{X \setminus \{x\}}$ which is good for $\mathfrak{M}|_{X \setminus \{x\}}$, see Theorem 9.5).

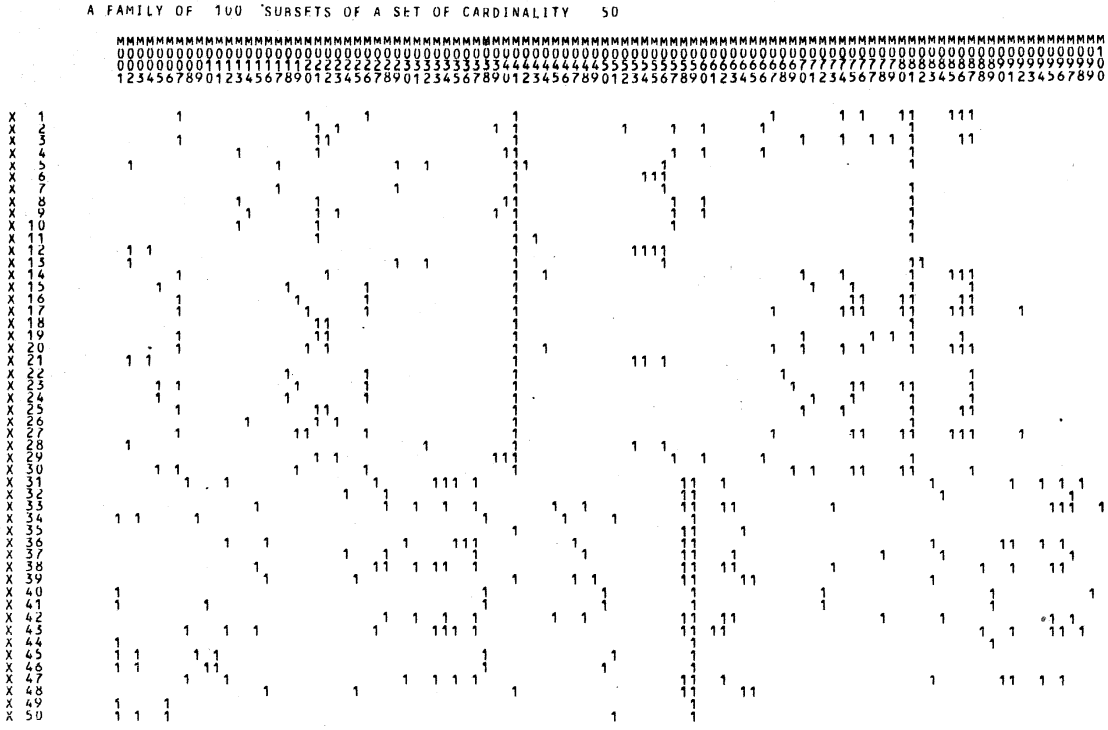

Fig. 8. The incidence matrix of a linear family of sets (0's are omitted)

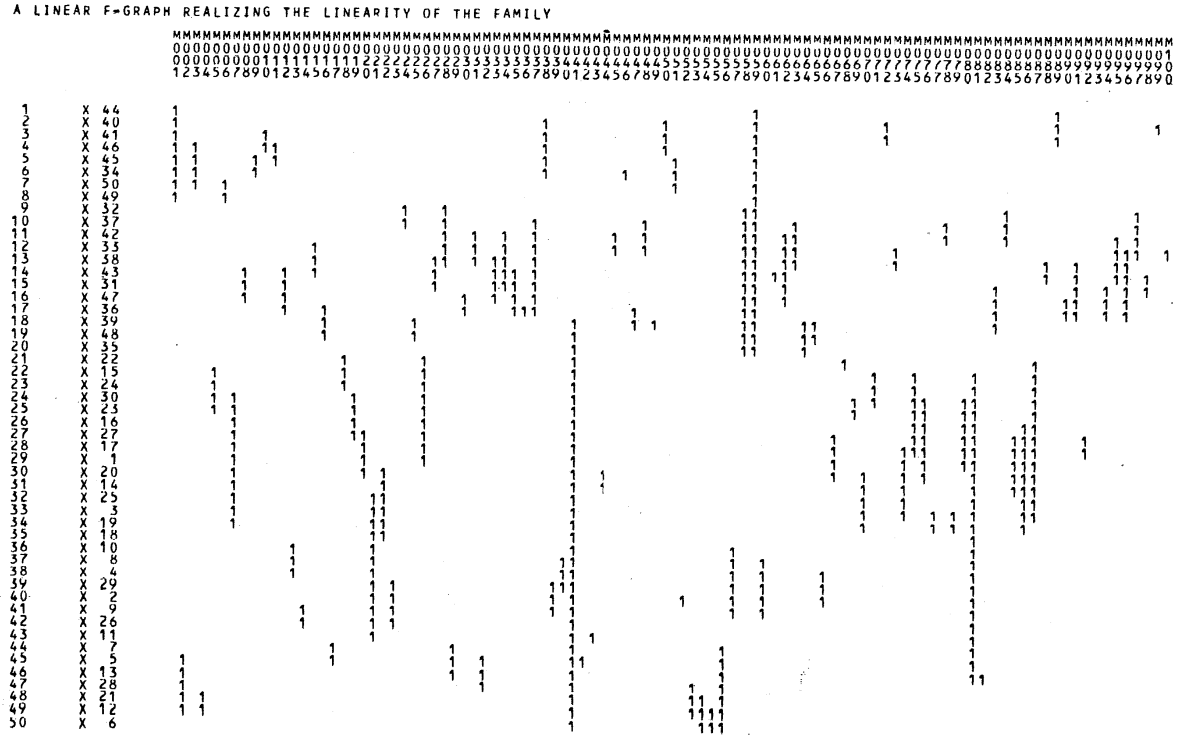

Fig. 9. The incidence matrix from Fig. 8 with permuted rows

Now we shall study the possibility of the addition of a new set to a family. By Theorem 13.2, if we add to a linear family $\mathfrak{M} \subseteq \mathscr{P}(X)$ a set $M \subseteq X$ which does not overlap any set of $\mathfrak{M}$ then $\mathfrak{M} \cup \{M\}$ remains linear ($\{M\}$ is then a block). In particular, M may be contained in a component of $\mathfrak{M}$ or $M \supseteq \bigcup \mathfrak{M}$.

THEOREM 14.1. *Let $\mathfrak{M} \in \mathscr{A}^*(X)$ and let $M \subseteq X$ be a set such that $\mathfrak{M}|_M \in \mathscr{L}^*(M)$. Then $\mathfrak{M} \cup \{M\} \in \mathscr{A}(X)$.*

Proof. Let $\mathfrak{M}$ be finally segmental over an acyclic f-graph $\langle X, S\rangle$ and let $\mathfrak{M}|_M$ be finally segmental over a linear f-graph $\langle M, S_1\rangle$. Let the f-graph $\langle X, S\rangle|_{X \setminus M}$ have k components. For each i, $1 \leqslant i \leqslant k$, there is a unique vertex x_i of the ith component such that $x_i \notin \mathscr{D}S_{X \setminus M}$. Let x_0 be the end of M in $\langle M, S_1\rangle$. Then the f-graph $\langle X, \bar{S}\rangle$, where

$$\bar{S} = S_{X \setminus M} \cup S_1^{-1} \cup \{\langle x_i, x_0\rangle : 1 \leqslant i \leqslant k\},$$

realizes the acyclicity of $\mathfrak{M} \cup \{M\}$. ■

Notice that the condition $\mathfrak{M}|_M \in \mathscr{L}^*(M)$ in the above theorem can be replaced by

$$(\exists N \in \mathfrak{M})\ M \cap \bigcup \mathfrak{M} \subseteq N.$$

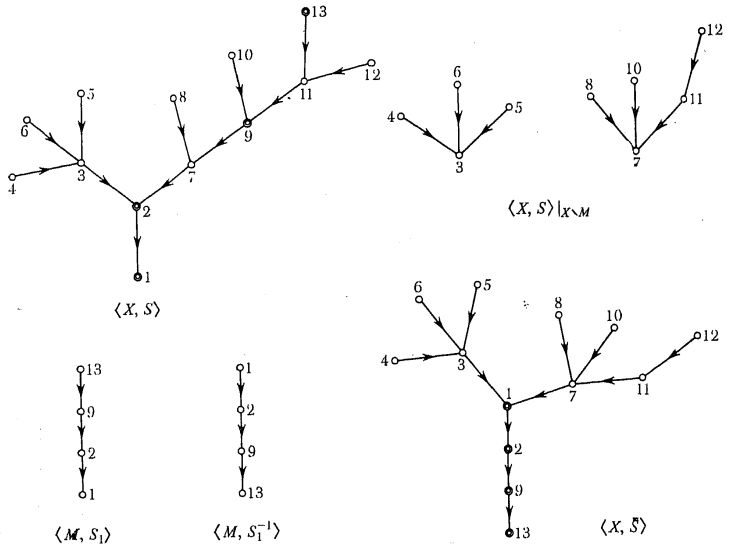

Fig. 10. Constructing the f-graph $\langle X, \bar{S}\rangle$ (see Example 14.2)

EXAMPLE 14.2. Let $\langle X, S\rangle$ be the f-graph shown in Fig. 10, let $\mathfrak{M}$ be the family of all final segments in $\langle X, S\rangle$ and let $M = \{1, 2, 9, 13\}$. The construction of $\langle X, \bar{S}\rangle$ is shown in Fig. 10. If the f-graph $\langle X, S\rangle$ is linear then $\langle X, \bar{S}\rangle$ is also linear. Hence we obtain

THEOREM 14.3 (Ghosh [5]). *If $\mathfrak{M} \in \mathscr{L}^*(X)$ and $M \subseteq X$ then $\mathfrak{M} \cup \{M\} \in \mathscr{L}(X)$.*

Proof. If $\mathfrak{M} \in \mathscr{L}^*(X)$ then $\mathfrak{M}|_M \in \mathscr{L}^*(M)$ for every $M \subseteq X$. ■

Let us notice that M is a final segment in the f-graph $\langle X, \bar{S}\rangle$ constructed in the proof of Theorem 14.2. From this fact one can easily deduce the following

THEOREM 14.4. *Let $\mathfrak{M}_1, \mathfrak{M}_2, \mathscr{L}(X)$, $\bigcup \mathfrak{M}_1 \cap \bigcup \mathfrak{M}_2 = \emptyset$ and let $M \subseteq X$. Then $\mathfrak{M}_1 \cup \mathfrak{M}_2 \cup \{M\} \in \mathscr{L}(X)$.* ■

Another theorem of this type, which can easily be proved, is the following

THEOREM 14.5. *Let $\mathfrak{M} = \mathfrak{M}_1 \cup \mathfrak{M}_2$, where $\mathfrak{M}_1 \cup \{X \setminus N : N \in \mathfrak{M}_2\} \in \mathscr{L}^*(X)$, and let $M \subseteq X$. Then $\mathfrak{M} \cup \{M\} \in \mathscr{C}(X)$.* ■

Now we shall study the possibility of adding a new element to the set X, that is, for a given $\mathfrak{M} \subseteq \mathscr{P}(X)$ we shall consider families $\overline{\mathfrak{M}} \subseteq \mathscr{P}(X \cup \{x\})$ such that $\overline{\mathfrak{M}}|_X = \mathfrak{M}$. We shall call $\overline{\mathfrak{M}}$ an *extension* of $\mathfrak{M}$. To each $M \in \mathfrak{M}$ there corresponds an $\bar{M} \in \overline{\mathfrak{M}}$ equal to M or $M \cup \{x\}$. It is easy to see that $\mathfrak{M}$ and $\overline{\mathfrak{M}}$ are in the same class of admissibility if the addition of x does not create any new nonempty component, i.e. if there exists a $y \in X$ such that for each $M \in \mathfrak{M}$, $y \in M$ iff $x \in \bar{M}$. In this case the necessary modification consists of adding the edge $\langle y, x\rangle$ and, if $y \in \mathscr{D}S$, replacing $\langle y, S(y)\rangle$ by $\langle x, S(y)\rangle$.

THEOREM 14.6. *Let $|X| \geqslant 2$ and let $\overline{\mathfrak{M}} \subseteq \mathscr{P}(X \cup \{x\})$ be an extension of $\mathfrak{M} \subseteq \mathscr{P}(X)$.*

(a) *$\overline{\mathfrak{M}}$ is cyclic iff there exist $y, z \in X$ and an f-graph $\langle X, S\rangle$ realizing the cyclicity of $\mathfrak{M}$ such that $S(y) = z$ and*

$$\{M \in \mathfrak{M}_0 : y, z \in M\} \subseteq \{M \in \mathfrak{M}_0 : x \in \bar{M}\} \subseteq \{M \in \mathfrak{M}_0 : y \in M \vee z \in M\}$$

where $\mathfrak{M}_0 = \{M \in \mathfrak{M} : M \neq \emptyset\}$;

(b) *$\overline{\mathfrak{M}}$ is linear iff there exist $y, z \in X$ and an f-graph $\langle X, S\rangle$ realizing the linearity of $\mathfrak{M}$ such that either $S(y) = z$ and*

$$\{M \in \mathfrak{M}_0 : y, z \in M\} \subseteq \{M \in \mathfrak{M}_0 : x \in \bar{M}\} \subseteq \{M \in \mathfrak{M}_0 : y \in M \vee z \in M\}$$

or $z \notin \mathscr{D}S$ and

$$\{M \in \mathfrak{M}_0 : x \in \bar{M}\} \subseteq \{M \in \mathfrak{M}_0 : z \in M\}.$$

Proof. To prove the "if" part of the theorem we replace in $\langle X, S\rangle$ the edge $\langle y, z\rangle$ by the edges $\langle y, x\rangle$, $\langle x, z\rangle$, or, in the second case in (b), we add the edge $\langle z, x\rangle$. To prove the "only if" part we take the contraction to X of an f-graph realizing the cyclicity (linearity) of $\overline{\mathfrak{M}}$. ■

15. Decomposition into linear and acyclic subfamilies

In practice we often encounter a situation in which a family of sets to be stored in the memory of a computer is not admissible. What we can do then is to decompose it into admissible subfamilies, and store each of them separately. Such a solution

leads to redundancy: some objects are stored several times. The best decomposition $\mathfrak{M} = \mathfrak{M}_1 \cup \mathfrak{M}_2 \cup \ldots \cup \mathfrak{M}_k$ is the one which minimizes this redundancy, expressed by the *package coefficient*

$$p = \sum_{i=1}^{k} |\bigcup \mathfrak{M}_i|/L \quad \text{where} \quad L = \sum_{M \in \mathfrak{M}} |M|.$$

The problem of finding such an optimal decomposition is very difficult. However, certain simple decomposition algorithms producing, in general, suboptimal decompositions can be proposed (see [17]). For instance, we can exploit the fact that each family consisting of two sets is linear, and decompose $\mathfrak{M}$ into such subfamilies. Suppose that $|\mathfrak{M}| = 2k$. The package coefficient of a decomposition

$$\mathfrak{M} = \{M_1, M_2\} \cup \{M_3, M_4\} \cup \ldots \cup \{M_{2k-1}, M_{2k}\}$$

is

$$p = \sum_{i=1}^{k} |M_{2i-1} \cup M_{2i}|/L = 1 - \sum_{i=1}^{k} |M_{2i-1} \cap M_{2i}|/L.$$

Thus, instead of minimizing

$$\sum_{i=1}^{k} |M_{2i-1} \cup M_{2i}|$$

we can maximize

$$\sum_{i=1}^{k} |M_{2i-1} \cap M_{2i}|.$$

The algorithm is as follows:

Step 1. We find a pair of distinct sets $M_1, M_2 \in \mathfrak{M}$ with the greatest intersection $|M_1 \cap M_2|$ and put $\mathfrak{M}_1 = \{M_1, M_2\}$, then we find a pair $M_3, M_4 \in \mathfrak{M} \setminus \mathfrak{M}_1$ with the greatest intersection $|M_3 \cap M_4|$. The process is continued until we obtain a decomposition

$$\mathfrak{M} = \mathfrak{M}_1 \cup \mathfrak{M}_2 \cup \ldots \cup \mathfrak{M}_k$$

such that for $i = 1, 2, \ldots, k$, $\mathfrak{M}_i = \{M_{2i-1}, M_{2i}\}$ and the sets M_{2i-1}, M_{2i} have the greatest intersection among all pairs of sets of $\mathfrak{M} \setminus (\mathfrak{M}_1 \cup \mathfrak{M}_2 \cup \ldots \cup \mathfrak{M}_{i-1})$.

Step 2 (*optional*). We find a pair of subfamilies $\mathfrak{M}_r = \{A, B\}$, $\mathfrak{M}_s = \{C, D\}$ $(r \neq s)$ with the greatest value of

$$\delta(r, s) = \max(|A \cap C| + |B \cap D|, |A \cap D| + |B \cap C|) - (|A \cap B| + |C \cap D|).$$

Then, if $\delta(r, s) > 0$, we interchange sets between $\mathfrak{M}_r$ and $\mathfrak{M}_s$, putting either

$$\mathfrak{M}_r = \{A, C\}, \quad \mathfrak{M}_s = \{B, D\} \quad (\text{if } |A \cap C| + |B \cap D| \geqslant |A \cap B| + |C \cap D|)$$

or

$$\mathfrak{M}_r = \{A, D\}, \quad \mathfrak{M}_s = \{B, C\} \quad (\text{if } |A \cap C| + |B \cap D| < |A \cap B| + |C \cap D|).$$

We repeat the above procedure until further improvement is impossible (i.e., until $\delta(r, s) \leqslant 0$).

This algorithm was implemented in PL/1 and was tested on an IBM/370 computer. For instance, for the family of sets

$$\{U(\alpha), U(\beta), U(a), U(b), U(c), U(d), U(A), U(B), U(C)\}$$

from the example given in Section 1, the decomposition was

$$\{U(\beta), U(A)\} \cup \{U(\alpha), U(d)\} \cup \{U(c), U(C)\} \cup \{U(b), U(B)\} \cup \{U(a)\}$$

with the package coefficient 0.747 (without Step 2 the package coefficient was 0.753). The execution time was 3.81 s.

Similarly, Theorem 11.1 can serve as a basis for decomposition into admissible subfamilies consisting of three sets. Suppose that $|\mathfrak{M}| = 3k$. The package coefficient of a decomposition

$$\mathfrak{M} = \{M_1, M_2, M_3\} \cup \{M_4, M_5, M_6\} \cup \ldots \cup \{M_{3k-2}, M_{3k-1}, M_{3k}\}$$

can be written as

$$p = \sum_{i=1}^{k} |M_{3i-2} \cup M_{3i-1} \cup M_{3i}|/L = 1 - \sum_{i=1}^{k} \varrho(M_{3i-2}, M_{3i-1}, M_{3i})/L$$

where we write $\varrho(A, B, C) = |A \cap B| + |B \cap C| + |C \cap A| - |A \cap B \cap C|$.

Thus, instead of minimizing

$$\sum_{i=1}^{k} |M_{3i-2} \cup M_{3i-1} \cup M_{3i}|$$

we can maximize

$$\sum_{i=1}^{k} \varrho(M_{3i-2}, M_{3i-1}, M_{3i}).$$

The algorithm is similar to the previous one. Instead of finding pairs M_{2i-1}, M_{2i} maximizing $|M_{2i-1} \cap M_{2i}|$ we now find triples $M_{3i-2}, M_{3i-1}, M_{3i}$ maximizing $\varrho(M_{3i-2}, M_{3i-1}, M_{3i})$. The algorithm was also implemented. For the same family of sets as before the decomposition was

$$\{U(\beta), U(b), U(A)\} \cup \{U(\alpha), U(c), U(C)\} \cup \{U(a), U(d), U(B)\}$$

with the package coefficient 0.680.

Thus, when we use the method of inverted files to implement the i.s.r. system from Section 1, the required storage space can be decreased by exploiting the above algorithms by 25.3% and 32%, respectively.

If the information on the values of descriptors, i.e. the sets $U(a)$, $a \in A$, is not available one should group together pairs (triples) of descriptors with a similar meaning, since the intersection of their values (or ϱ in the case of triples) is then expected to be relatively large.

For more details and other decomposition algorithms the reader is referred to [17].

16. Admissibility over i.s.r. systems

As we remarked in Section 9, to every family of terms $\mathscr{H} \subseteq \mathscr{T}$ there corresponds, in a fixed i.s.r. system $\mathscr{S}$, a family of sets $\mathfrak{M} = \{||t||_{\mathscr{S}} : t \in \mathscr{H}\}$. This enables us to speak about admissible families of terms:

DEFINITION 16.1. (a) A family $\mathscr{H} \subseteq \mathscr{T}$ is *admissible* over an i.s.r. system $\mathscr{S}$ if the family of sets $\mathfrak{M} = \{||t||_{\mathscr{S}} : t \in \mathscr{H}\}$ is admissible.

(b) A family $\mathscr{H} \subseteq \mathscr{T}$ is *absolutely admissible* if it is admissible over every i.s.r. system (recall that A and R_I are fixed).

In all the definitions and theorems of this section admissibility can be replaced by linearity, cyclicity, etc.

THEOREM 16.2. *Let $\mathscr{S}_1, \mathscr{S}_2$ be two* i.s.r. *systems and let $\mathscr{S}_1 \equiv \mathscr{S}_2$* (see Section 5). *Then a family $\mathscr{H} \subseteq \mathscr{T}$ is admissible over $\mathscr{S}_1$ iff it is admissible over $\mathscr{S}_2$. In particular, we can take as $\mathscr{S}_2$ the unique (up to isomorphism) selective system equivalent to $\mathscr{S}_1$.*

Proof. If $\mathscr{S}_1 \equiv \mathscr{S}_2$ then the families $\{||t||_{\mathscr{S}_1}: t \in \mathscr{H}\}$ and $\{||t||_{\mathscr{S}_2}: t \in \mathscr{H}\}$ are easily seen to be similar (see Def. 10.1(c)). Thus our theorem follows from Theorem 10.2. ■

THEOREM 16.3. *Let $\mathscr{S}_1 \subseteq \mathscr{S}_2$* (see Def. 6.1) *and let $\mathscr{H}$ be a family of terms in the language of $\mathscr{S}_1$. If $\mathscr{H}$ is admissible over $\mathscr{S}_2$ then it is admissible over $\mathscr{S}_1$.*

Proof. Let $\mathfrak{M}_2 = \{||t||_{\mathscr{S}_2}: t \in \mathscr{H}\}$ be admissible. Then $\mathfrak{M}_1 = \{||t||_{\mathscr{S}_1}: t \in \mathscr{H}\}$ is also admissible, since $\mathfrak{M}_1 = \mathfrak{M}_2|_{X_1}$ where X_1 is the set of objects of $\mathscr{S}_1$ (see Theorem 9.5). ■

THEOREM 16.4. *Let $\mathscr{S}_1$ and $\mathscr{S}_2$ be two* i.s.r. *systems and let $\mathscr{H} \subseteq \mathscr{T}$. Assume that for every simple term $t \in \mathscr{T}_0$, $||t||_{\mathscr{S}_2} = \emptyset \Rightarrow ||t||_{\mathscr{S}_1} = \emptyset$. Then, if $\mathscr{H}$ is admissible over $\mathscr{S}_2$, it is also admissible over $\mathscr{S}_1$.*

Proof. We use the preceding two theorems and the fact that $\mathscr{S}_1$ is equivalent to a subsystem of $\mathscr{S}_2$. ■

If we take $\mathscr{S}_2 = \mathscr{S}_{\max}$ (see Section 5), we obtain the following

COROLLARY 16.5. *A family of terms is absolutely admissible iff it is admissible over $\mathscr{S}_{\max}$.* ■

In practice, updating a system causes the set of objects and the function U to vary in time. Consequently we are interested in families of terms which are absolutely admissible rather than admissible over a fixed system. However, there is one exception: Sometimes we know some relations between descriptors of different attributes. This is equivalent to the possibility of predicting *a priori* that certain components will always be empty. In such a case we may consider families of terms admissible over a certain proper restriction of $\mathscr{S}_{\max}$.

From the fact that the admissibility of a family depends only on which of its components are nonempty, we can easily deduce the following

THEOREM 16.6. *For every $\mathscr{H} \subseteq \mathscr{T}$ there exists a formula Φ of the language $\mathscr{L}_A$, such that for every* i.s.r. *system $\mathscr{S}$,*

$$\mathscr{H} \text{ is admissible over } \mathscr{S} \Leftrightarrow \mathscr{S} \models \Phi. \blacksquare$$

Thus the admissibility of a family of terms is expressible within the language $\mathscr{L}_A$. For instance, if $\mathscr{H} = \{r, s, t\}$ then the sentence "$\mathscr{H}$ is acyclic" is equivalent to the formula $r \cdot s \cdot \sim t = F \vee r \cdot \sim s \cdot t = F \vee \sim r \cdot s \cdot t = F$ (see Theorem 11.2). The above formula is positive, and this is a general fact. It can be shown that Φ is always equivalent to a positive formula, this fact being a special case of the following, more general theorem:

THEOREM 16.7. *Let a formula Φ have the following property: for every system $\mathscr{S} = \langle X, A, R_I, U\rangle$,*

$$\mathscr{S} \models \Phi \Rightarrow (\forall Y \subseteq X)\, \mathscr{S}|_Y \models \Phi.$$

Then Φ is equivalent to a positive formula. ■

The formula Φ from Theorem 16.6 is usually very complicated, and we do not know any reasonable way of constructing it. This is the reason why this theorem is usually of no practical use.

17. Related combinatorial problems

In this section some combinatorial problems related to admissible families of sets are briefly described.

DEFINITION 17.1.(a) The *intersection graph* $\mathscr{G}(\mathfrak{M})$ of a family $\mathfrak{M}$ is a nondirected graph with $\mathfrak{M}$ as the set of vertices, two different vertices M, N joined by an edge iff $M \cap N \neq \emptyset$.

(b) A nondirected graph G is *representable* in a class of admissibility $\mathscr{K}(X)$ if there exists $\mathfrak{M} \in \mathscr{K}(X)$ such that G and $\mathscr{G}(\mathfrak{M})$ are isomorphic.

Boland and Lekkerkerker [1] give the following characterization of representable in $\mathscr{L}(X)$ graphs (in a slightly different form):

THEOREM 17.2. *A nondirected graph is representable in $\mathscr{L}(X)$ for some X, iff it is a non-asteroidal rigid circuit graph.*

(A graph is *asteroidal* if there exist vertices v_1, v_2, v_3 and paths P_1, P_2, P_3 such that for $i = 1, 2, 3$, P_i joins the two vertices v_j, $j \neq i$, and there is no edge joining v_i with any vertex of P_i. A graph is a *rigid circuit graph* if for each of its elementary cycles of length > 3 there exists an edge joining two vertices of that cycle, not joined by any edge of the cycle.) ■

Simple examples show that, in general, there are many families, both linear and non-linear, with the same intersection graph. Nevertheless, the following theorem holds:

THEOREM 17.3 (see [11]). *A family $\mathfrak{M}$ is linear iff $\mathscr{G}(\mathfrak{M} \cup \mathscr{S}(\mathfrak{M}))$ is a non-asteroidal rigid circuit graph.* ■

Since there is a strong restriction on the number of non-empty components of a linear family (see Theorem 10.4(b)), the above theorem can be of some use in testing a family against linearity. Some methods of testing whether a given graph is asteroidal or a rigid circuit graph can be found in [1] and [3].

Another criterion of linearity is due to Nakano ([22], [23]):

THEOREM 17.4. *A family $\mathfrak{M} \subseteq \mathscr{P}(X)$ is linear if* cl($\mathfrak{M}$), *the least family $\mathfrak{M}' \supseteq \mathfrak{M}$ such that*

$$(\forall M, N \in \mathfrak{M}')\ (M \cap N \neq \emptyset \Rightarrow M \cup N \in \mathfrak{M}'),$$

contains no triangles.

(A *triangle* is a subfamily $\{M_1, M_2, M_3\}$ such that $M_1 \cap M_2 \cap \bar{M}_3 \neq \emptyset \wedge M_1 \cap \bar{M}_2 \cap M_3 \neq \emptyset \wedge \bar{M}_1 \cap M_2 \cap M_3 \neq \emptyset$, where $\bar{M}_i = X \setminus M_i$.) ■

An interesting combinatorial problem is that of reconstructing a family from its intersection pattern. The *intersection pattern* of a family $\mathfrak{M} = \{M_1, M_2, \ldots, M_n\}$ is the $n \times n$ matrix $[a_{ij}]$, $a_{ij} = |M_i \cap M_j|$. In general the intersection pattern of a family does not determine that family up to isomorphism. But if two families $\mathfrak{M}$, $\mathfrak{N}$ have the same intersection pattern and $\mathfrak{M}$ is cyclic, then $\mathfrak{M}$ and $\mathfrak{N}$ are isomorphic (see [11]). Thus the intersection pattern of a family contains the information on whether this family is cyclic or not. Similarly for the case of linear families.

Other combinatorial problems arise in the so called two-dimensional file organization, proposed, in the "linear" case, by Ghosh [6]. For the details the reader is referred to [6] and [14].

18. Perspectives

Between the time when the Semester ended and the time of writing there have been new developments in the subject. Most of them have been worked out in the seminars continuing the work of the group formed at the Semester. We give here a short report on the progress obtained, in order to show further perspectives of the subject.

A. *Quantifiers*

One of the problems not considered in the formalism is that of the power of the value of a term. This may be treated with the help of so called *quantifiers*, both unary and having more arguments. We show several of them.

First, we extend our language by adding one or more symbols Q_i, and to each Q_i we adjoin a natural number n_i (the *valency* of quantifier Q_i). We extend the notion of a formula by stipulating that if $t_0, \ldots, t_{n_i-1}$ are terms then $Q_i\, t_0 \ldots t_{n_i-1}$ is a formula.

(i) Power quantifiers

$$Q^n t \quad \text{is interpreted as} \quad |||t||| \geqslant n.$$

(ii) Frequency quantifiers

$$Q^r t \quad \text{is interpreted as} \quad \frac{|||t|||}{|X|} \geqslant r \qquad (0 \leqslant r \leqslant 1).$$

(iii) "More than" quantifier

$$Qt_1 t_2 \quad \text{is interpreted as} \quad |||t_1||| > |||t_2|||.$$

(iv) "As many" quantifier

$$Qt_1 t_2 \quad \text{is interpreted as} \quad |||t_1||| = |||t_2|||.$$

There are examples of other quantifiers.

Notice that the simplest quantifiers "There exist" and "For all" are definable in our system:

$$\exists t \Leftrightarrow t \neq F,$$
$$\forall t \Leftrightarrow t = T.$$

Results analogous to those concerning the existence of the system $\mathscr{S}_{\max}$, have been also obtained for the systems with quantifiers.

B. *Cardinality function*

There is another, slightly more complicated approach to the problem of expressing the power of the value of a term. Namely, we add a symbol ϱ to the language and extend the set of terms by stipulating that if t is a term not containing ϱ then ϱt is also a term. The domain of interpretation of such a language consists of a Boolean algebra of subsets of a set X, the truth values $\lfloor\!\!\diagup$ and $\diagup\!\!\rfloor$, and in addition the set N of natural numbers. Terms not containing ϱ and formulas are interpreted as before: as subsets of X and truth values, respectively. ϱt is interpreted as a natural number as follows:

$$||\varrho t|| = |||t||| \in N,$$

i.e. the function symbol ϱ is interpreted as the function $|\cdot|$.

In the case of quantifiers queries could be of the type "Does $||t||$ consist of at least 7 objects?" whereas now we can ask "How many objects does $||t||$ consist of?", the latter question not being expressible by means of quantifiers.

Constants $\mathbf{1}, \mathbf{2}, \ldots$ for the natural numbers ($\mathbf{n}$ being always interpreted as n), as well as relation symbols $\leqslant$, $<$, etc., can also be introduced into the language. Then we can form formulas of the type $\varrho t > 7$, $\varrho t \leqslant \varrho s$, $\varrho t = \varrho s$, etc., and express most of the quantifiers.

C. *Modifiers*

The problem of a hierarchical approach has been solved in the formalism. It seems that from the point of view of implementation procedures it was not the most suitable choice. This problem is tackled with the help of so called modified i.s.r. systems.

Apart from the basic sets of descriptors we have the set of so called *modifiers*. Let H be a set of modifiers, $H^* = \bigcup_{n \in N} H^n$, i.e., a set of sequences of modifiers. $F\colon A \to \mathscr{P}(H^*)$ is a *context function* iff it satisfies the following condition:

$$\vec{h} \in F(a) \Rightarrow (\forall n < \mathrm{lh}(\vec{h}))\, \vec{h} \restriction n \in F(a)$$

where $\mathrm{lh}(\vec{h})$ is the length of $\vec{h}$ and $\vec{h} \restriction n$ is the sequence of the last n modifiers of $\vec{h}$. (Just consider the example: red, strongly red, almost red, etc.)

The definition of the set of terms is changed by adding the following clause:

$$\text{If } a \in A \wedge \vec{h} \in F(a) \text{ then } \vec{h}a \in \mathscr{T}.$$

The definition of the set of formulas is not changed.

The definition of semantics is slightly changed to allow the terms $\vec{h}a$. In this way a number of results on hierarchical systems are more readily expressed. Indeed,

a suitable choice of modifiers allows the introduction of hierarchy in the set of descriptors. The subject is treated in detail in a forthcoming paper by Jaegermann, Marek and Sobolewski [9], to which we refer the reader.

D. *Systems with incomplete information*

In [21] we promised to consider the problem of incomplete information. This may be treated in various ways. One of them was thoroughly investigated by Jaegermann [8], who considered systems where one knew only "approximate" descriptions of objects.

Let us consider a new sort of descriptors: a_{ij} for every subset A_{ij} of A_i. The intuition is that the value of a_{ij} is the set of objects which have at least one of the properties $a \in A_{ij}$. This corresponds to the following situation: We know what objects are "blue or green" but we are not able to classify them completely. This leads to consideration closely related to the intuitionistic logic and so called pseudo-Boolean algebras. The subject is treated in detail in a paper by Jaegermann [8].

E. *Covering systems*

Let $\mathscr{S}_1 = \langle X, A, R_I, U\rangle$ be an i.s.r. system. A system $\mathscr{S}_2 = \langle X, B, R_J, V\rangle$ is said to *cover* $\mathscr{S}_1$ iff the following conditions are satisfied:

(i) $J \subseteq I$.

(ii) There is a mapping $C: \bigcup_{j\in J} A_j \to B$ such that

(a) $a \in A_j \Rightarrow C(a) \in B_j$, for each $j \in J$.

(b) $V(b) = \bigcup_{a\in C^{-1}(b)} U(a)$.

The intuition which is connected with this is the following: $\mathscr{S}_2$ is a "presystem" classifying the objects in X according to some of the attributes from I, but not necessarily in the same way as in $\mathscr{S}_1$.

We have the following fact:

The Boolean algebra of subsets of X describable in $\mathscr{S}_2$ *(i.e.* $\mathscr{B}(\mathscr{S}_2)$*) is a subalgebra of* $\mathscr{B}(\mathscr{S}_1)$ *and the generators of* $\mathscr{B}(\mathscr{S}_2)$ *(i.e., values of simple terms in* $\mathscr{S}_2$*) are sums of values of simple terms in* $\mathscr{S}_1$.

This suggests a special implementational algorithm, suitable in some cases. Namely, we consider the components (i.e. the generators) of $\mathscr{B}(\mathscr{S}_2)$ as autonomous i.s.r. systems.

Let us illustrate this by an example.

Assume that we classify documents by the name of the author, the subject, and the year and month of appearance. We can form a presystem (i.e., a covering system) where we classify documents according to only the first letter of the name and the year of appearance. Then the storage area where the appropriate component is stored is organized as a smaller i.s.r. system where we search for documents only according to the other letters of the name, month of appearance and subject.

A detailed presentation of this idea is contained in a forthcoming paper by the second author.

F. *Many-sorted systems*

Apart from systems where there are only unary descriptors we may consider also systems where we speak about binary, ternary, etc. relations over a given set of objects.

Let us consider, as an example, a system classifying human beings where there is a binary descriptor "married". Then apart unary relations, i.e. sets, we have binary relations describable within the system. Consider for instance the relation consisiting of married couples where the husband is a doctor and the wife a nurse, say. Let us observe that this is nothing else but the intersection of the following two relations: one consisting of pairs where the first element is an object belonging to $\downarrow||doctor||$ and the second element is an object belonging to $||nurse||$ (i.e., their Cartesian product), with the relation $||married||$.

Now apart from Boolean operations we need other operations. For the sake of simplicity let us restrict ourselves to the case where only binary descriptors are allowed. Apart from terms which we now call unary terms we have binary terms defined as follows. If t is a binary descriptor then it is a binary term, if t_1, t_2 are binary terms then $t_1, t_1 \cdot t_2$, $t_1 + t_2$ are binary terms. If t is a binary term then t^{-1} (corresponding to inverse relation) is a binary term. Finally, if t_1, t_2 are unary terms then $t_1 \times t_2$ is a binary term.

Thus, in our example we have formed $(doctor \times nurse) \cdot married$.

Let us notice that the binary relations—through projections—generate unary relations (e.g. *married* generates *husband* and *wife*). Thus the class of unary terms is also extended.

All this leads to a nice but much more complex theory of many-sorted i.s.r. systems. The subject was discussed by the second author in his talk at the Jadwisin Conference closing the Semester.

G. *If-then-else construction*

A natural extension of our language is the if-then-else construction. Let us consider, as an example, the term **if** $s \cdot t = s$ **then** s **else** F (the semantics of the if-then-else construction is defined in the natural way). If a term t contains the if-then-else construction, then for each system $\mathscr{S}$ there is a term t_1 in the basic language (without the if-then-else construction) such that $||t_1||_{\mathscr{S}} = ||t||_{\mathscr{S}}$. However, this construction cannot be eliminated uniformly for all i.s.r. systems. To define precisely the set of terms and formulas in such an extended language we use simultaneous induction, as is done in ALGOL 60 to define arithmetic and Boolean expressions.

H. *Implementational methods*

A new method of organizing an i.s.r. system has been proposed by the first author in [13]. It exploits a normal form, other than the standard one. This form, called *minimal regular* form, is the sum of all the so called *prime regular implicants* of a given term. To define the latter, let us suppose that $I = \{1, 2, \ldots, k\}$, $A_i = \{a_2^{(i)}, a_1^{(i)}, \ldots, a_{n_i-1}^{(i)}\}$. A positive primitive term s is a prime regular implicant of t iff s is of the form $a_{b_1}^{(1)} \cdot a_{b_2}^{(2)} \cdot \ldots \cdot a_{b_m}^{(m)}$, $m \leqslant k$ and $s \leqslant t$.

If the components in our system are arranged in a linearly ordered memory according to the lexicographic ordering defined in Section 8, then the value of each prime regular implicant is a segment in this ordering. Thus the minimal regular form of a term t defines a decomposition of $||t||$ into a set of disjoint segments. The addresses of the head and the end of each such segment can easily be computed. For the details the reader is referred to [13].

References

[1] Boland, J. Ch., C. C. Lekkekerker, *Representation of a finite graph by a set of intervals on the real line*, Fund. Math. 51 (1962), pp. 45–64.

[2] Cohn, P. M., *Universal Algebra*, Harper and Row, New York 1965.

[3] Fulkerson, D. R., O. A. Gross, *Incidence matrices and interval graphs*, Pacif. J. Math. 15 (1965), pp. 835–855.

[4] Ghosh, S. P., *File organization: the consecutive retrieval property*, Comm. ACM. 15 (1972) pp. 802–808.

[5] —, *On the theory of consecutive storage of relevant records*, Inform. Sci. 6 (1973), pp. 1–9.

[6] —, *File organization: consecutive storage of relevant records on drum-type storage*, Inform. Control 25 (1974), pp. 145–165.

[7] Harary, F., *Graph Theory*, Addison and Wesley, Reading 1969.

[8] Jaegermann, M., *Information storage and retrieval systems—mathematical foundations IV. Systems with incomplete information*, CC PAS Reports, No. 215, Warsaw 1975.

[9] —, W. Marek, M. Sobolewski, *Information storage and retrieval systems—mathematical foundations III. Tree-structured-attribute systems*, CC PAS Reports, No. 214, Warsaw 1975.

[10] Kuratowski, K., A. Mostowski, *Set Theory*, North-Holland, Amsterdam 1967.

[11] Lipski, W., *Information storage and retrieval systems—mathematical foundations II*, CC PAS Reports No. 153, Warsaw 1974.

[12] —, *Combinatorial aspects of information storage and retrieval*, Proceedings of the Third Symposium on Mathematical Foundations of Computer Science, Jadwisin 1974, Springer-Verlag, Berlin 1975, pp. 313–326.

[13] —, *Kombinatoryczne aspekty teorii wyszukiwania informacji* (in Polish), Doctoral Dissertation, Warsaw 1975.

[14] —, *Variants of file organization for a family of three sets*, CC PAS Reports No. 175, Warsaw 1975.

[15] —, *An efficient method of information retrieval*, CC PAS Reports No. 194, Warsaw 1975.

[16] —, W. Marek, *An application of graph theory to information retrieval*, Bull. Acad. Polon. Sci., Sér. Sci. Math. Astronom. Phys. 22 (1974), pp. 691–695.

[17] —, —, *File organization, an application of graph theory*, Automata, Languages and Programming (Proc. Colloq. Saarbrücken 1974), Springer-Verlag, Berlin–Heidelberg–New York 1974, pp. 270–279.

[18] Lyndon, R., *Notes on Logic*, Van Nostrand, Princeton 1966.

[19] Marek, W., Z. Pawlak, *Mathematical foundations of information storage and retrieval, Parts* 1, 2, 3, CC PAS Reports No. No. 135, 136, 137, Warsaw 1973.

[20] —, —, *On the foundations of information retrieval*, Bull. Acad. Polon. Sci., Sér. Sci. Math. Astronom. Phys. 22 (1974), pp. 447–452.

[21] —, —, *Information storage and retrieval systems—mathematical foundations I*, CC PAS Reports No. 149, Warsaw 1974. Also published in Theoretical Computer Science 1 (1976), pp. 331–354.

[22] Nakano, T., *A characterisation of intervals; the consecutive one's or retrieval property*, Comment. Math. Univ. Sancti Pauli 22 (1973), pp. 49–59.

[23] Nakano, T., *A remark on the consecutivity of incidence matrices*, Comment. Math. Univ. Sancti Pauli 22 (1973), pp. 61–62.

[24] Pawlak, Z., *Mathematical foundations of information retrieval*, CC PAS Reports No. 101, Warsaw 1973.

[25] Shoenfield, J. R., *Mathematical Logic*, Addison and Wesley, Reading 1967.

[26] Wong, E., T. C. Chiang, *Canonical structure in attribute based file organization*, Comm. ACM 14 (1971), pp. 593–597.

Note added in proof. We list below some recent publications concerning the subject treated in the paper.

[27] Ehrich, H.-D., W. Lipski, *On the storage space requirement of consecutive retrieval with redundancy*, Inform. Proc. Lett. 4 (1976), pp. 101–104.

[28] Konikowska, B., *Continuous information systems*, CC PAS Reports No. 273, Warsaw 1977.

[29] Lipski, W., *Three queries file organizations*, Fundamenta Informaticae 1(1977), to appear.

[30] —, *Informational systems with incomplete information*, Automata, Languages and Programming (Proc. Colloq. Edinburgh 1976), Edinburgh University Press, Edinburgh 1976, pp. 120–130.

[31] —, *Informational systems: semantic issues related to incomplete information, Part I*, CC PAS Reports No. 275, Warsaw 1977.

[32] —, *Information storage and retrieval—mathematical foundations II (Combinatorial problems)*, Theoretical Computer Science, to appear.

[33] Marek, W., *Some normal forms codings and their applications to information retrieval*, CC PAS Reports No. 265, Warsaw 1973.

[34] Marek, W., I. Rode-Babczenko, *A decomposition of informational systems*, CC PAS Reports No. 212, Warsaw 1975.

[35] Marek, W., T. Traczyk, *Stochastic informational systems I*, CC PAS Reports No. 247 Warsaw 1976. To appear in Fundamenta Informaticae 1(1977).

[36] Rajkow-Krzywicki, J., *An application of structural numbers to information retrieval*, CC PAS Reports No. 254, Warsaw 1976.

[37] Traczyk, T., Jr., *On consecutive retrieval from drum-type storage*, CC PAS Reports, to appear.