Inform. Systems Vol. 6, No. 3, pp. 205–218, 1981
Printed in Great Britain.

0306–4379/81/030205–14$02.00/0

INFORMATION SYSTEMS THEORETICAL FOUNDATIONS

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

(*Received* 14 *March* 1980; *in revised form* 9 *December* 1980)

Abstract—Some basic concepts concerning information systems are defined and investigated. With every information system a query language is associated and its syntax and semantics is formally defined. Some elementary properties of the query language are stated. The presented approach leads to a new information systems organization. The presented idea was implemented and the implementation shows many advantages compared with other methods.

INTRODUCTION

This paper reports part of the activities of the Information Systems Group in Warsaw.

We proposed and investigated in this Group a certain mathematical model for attribute based information systems. This model was first published by Pawlak[1] and extended by Marek and Pawlak[2]. In this report we use somewhat a new formulation of the discussed model and state some new problems.

The idea of an information system investigated in this report is slightly related to that of Codd[3], Salton[4] and Wang and Chiang[5], however there are essential differences between them. In our approach in contrary to [4] and [5] the query language is formally introduced and extensively investigated. The language plays also an essential role in the approach of Cherniavsky and Schneider[6], where not the data model but a data information language (extended first order language) is the departure point of an information systems study and implementation.

In our approach we try link both mentioned views together. Formal definition of syntax and semantics of a query language is introduced in this approach—which offers deeper insight and understanding of phenomena involved in information processing, and provide facilities for using standard logical methods in this area.

For example in the relational model, the query language is not defined precisely, there is no formal definition of the semantics what causes that some basic notions are obscure in this approach.

The functional dependency of attributes is another good example to trace the differences between our approach and relational model. In both models data are arranged into tables. The columns are marked by attributes, and each column contains values of an attribute marking that column. Each such a table defines also all functional dependencies between attributes and this fact is a departure point of our definition of functional dependency, whereas in relational model some initial functional dependencies are assumed to be valid—independently from the database—on the basis of the knowledge about the real world. This may, however, lead to contradiction, that is to say, preassumed dependencies may be not consistant with those already "existing" in the data base.

To this end the main difference between relational model and the model of an information system discussed in this paper is that we are concerned mainly with subsets of object having some properties expressable in the query language, whereas in the relational model the relations between data are of primary conçern.

In fact our principal aim is to precise some basic notions concerning information systems as a "Uniform theory", and we believe that the obtained results may be regarded as the first step in this direction.

The proposed model of an information system has been implemented in 1978 by E. Margański for an agriculture library, with *ca.* 50,000 documents, on Polish computer ODRA 1305 (compatible with ICL 1900).

Detailed description of this implementation and practical results one can find in Margański[7]. Short version of the paper will be published in Information System.

The model of an information system considered in this paper has been extended in various directions.

Information systems with incomplete information are investigated by Jaegermann[8], Lipski[9] and Orłowska[10].

Stochastic information systems are introduced by Konikowska and Traczyk[11].

Time varying information systems are considered by Orłowska[12] and Wakulicz-Deja[13].

1. INFORMATION SYSTEMS

In this paragraph we give the basic notions of the paper, which will be discussed in details in the rest of the paper.

The main notion is that of *information system*. The basic component of an information system is a finite set of objects X, for example human beings, books, etc. The objects are classified by means of a finite set A of *attributes*. With every attribute $a \in A$, there is associated a non-empty set V_a of *values* of an attribute a; V_a will be also referred to as *domain* of attribute a. For instance if a is "sex" then $V_a = \{\text{male}, \text{female}\}$, if a is colour, then for example $V_a = \{\text{red}, \text{green}, \text{blue}\}$. Naturally some attributes can share the set of values, for example domain of attribute "length" and "height" is the same and it is the set of nonnegative reals.

In order to "define" some properties of objects we introduce a function ρ from $X \times A$ into V ($V = \bigcup_{a \in A} V_a$), such that $\rho(x, a) \in V_a$ for every $x \in X$ and $a \in A$.

This is to mean that by means of the function ρ we associate with each object its description—a set of attribute values.

Now we can give formal definition of an information system (see Pawlak[10]). By an information system we shall mean a 4-tuple

$$S = \langle X, A, V, \rho \rangle,$$

where X is a finite set of *objects*, A in a finite set of *attributes*, $V = \bigcup_{a \in A} V_a$, where V_a is the set of *values of attribute a*, and $\text{card}(V_a) > 1$, ρ is a function from $X \times A$ into V.

If the function ρ is total then system will be called *complete*; otherwise the system is *incomplete*. We shall consider here complete systems only.

Example 1

Let us consider very simple information system defined as follows:

$$X = \{x_1, x_2, x_3, x_4, x_5\},$$
$$A = \{\text{sex}, \text{salary}, \text{age}\},$$
$$V = \{V_{\text{sex}} \cup V_{\text{sal}} \cup V_{\text{age}}\},$$

where $V_{\text{sex}} = \{\text{male, female}\}$, $V_{\text{sal}} = \{\text{low, medium, high}\}$, and $V_{\text{age}} = \{\text{young, middle, old}\}$.

The salary "low" is less than \$6000 a year; "medium"—between \$6000 and \$24,000; "high"—more than \$24,000 a year.

The age "young" is to mean less than 21; "middle"—between 21 and 40, "old" more than 40.

The function ρ in our example is defined by the following table:

X	SEX	SALARY	AGE
x_1	male	low	young
x_2	male	high	middle
x_3	female	low	young
x_4	male	medium	old
x_5	female	low	middle

We shall also use the notion of a *descriptor* of an attribute a.

By a descriptor we shall mean any element of the set $\{a\} \times V_a$. That is to mean that descriptors are pairs of the form (a, v), where $v \in V_a$. For instance in Example 1 the following are descriptors: (AGE, young), (SALARY, low), (SEX, male). Instead of (AGE, young) we shall write (AGE = young) etc., as it is assumed in programming praxis.

For every $x \in X$ we define the function ρ_x from A into V such that $\rho_x(a) = \rho(x, a)$. We shall call this function *information* (or *data*) *about x in S*.

For instance in Example 1 information about x_2 is the following function:

$$\rho_{x_2} = \begin{array}{ccc} \text{SEX} & \text{SALARY} & \text{AGE} \\ \text{male} & \text{high} & \text{middle}. \end{array}$$

In other words information about x in S is simply a set of descriptors corresponding to all attributes in the system. Thus we may write information about x_2 in the form:

$$\{(\text{SEX} = \text{male}), (\text{SALARY} = \text{high}), (\text{AGE} = \text{middle})\}.$$

Let us notice that our information about objects is *exhaustive* and *exclusive*, i.e. values of each attribute exhaust all possibilities, and only one attribute value can be associated with each object.

Because we deal in this paper only with finite systems, that is systems having finite number of objects, finite number of attributes and finite domains of attributes we may identify the notion of an information systems with the finite table defining the function ρ. The columns of the table, labelled with attributes, are composed of values of corresponding attributes and rows of the table, labelled with objects, are informations about corresponding objects. Of course we admit occurrence of the same rows in the table. Naturally the order of columns and rows in the table is insignificant.

2. PROPERTIES OF INFORMATION SYSTEMS

In the paragraph we shall give some more details about information systems which will give better insight in the considered notion.

Any function φ from A into V such that for every a, $\varphi(a) \in V_a$ will be called *information* in S. The set of all informations in S will be denoted by $\text{Inf}(S)$. There are evidently

$$\prod_{a \in A} \text{card}(V_a)$$

informations (different) in the system S.

For instance in the example given in the previous paragraph we have

$$\text{card}(V_{\text{sex}}) \cdot \text{card}(V_{\text{sal}}) \cdot \text{card}(V_{\text{age}}) = 2 \cdot 3 \cdot 3 = 18 \text{ informations.}$$

For every $\varphi \in \text{Inf}(S)$, we define $X_\varphi = \{x \in X : \rho_x = \varphi\}$. We can interpret X_φ as a set of all objects $x \in X$ whose information in S is identical with φ. This is to mean that objects belonging to the set X_φ are indistinguishable in the system S.

An information φ is said to be *empty* iff $X_\varphi = \emptyset$. Otherwise it is said to be nonempty.

An information φ is said to be *selective* if $\text{card}(X_\varphi) = 1$. System S is said to be *selective* iff every nonempty information in S is selective. A system S is said to be *maximal* iff every information in S is non-empty.

Example 2

Let $S = \langle X, A, V, \rho\rangle$ be an information system defined by the table

X	a	b	c
x_1	p_1	q_2	r_1
x_2	p_2	q_3	r_2
x_3	p_1	q_2	r_1
x_4	p_1	q_1	r_3.

The function φ such that $\varphi(a) = p_1$, $\varphi(b) = q_2$, $\varphi(c) = r_1$ is an information in S and $X = \{x_1, x_3\}$, because

$$\begin{aligned} X_\varphi &= \{x \in X: \varphi_x = \varphi\} \\ &= \{x \in X: \bigwedge_{a \in A} \rho_x(a) = \varphi(a)\} \\ &= \bigcap_{a \in A} \{x \in X: \rho(x(a) = \varphi(a))\} \\ &= \{x \in X: \rho(x, a) = p_1\} \cap \{x \in X: \rho(x, b) = q_2\} \\ &\quad \cap \{x \in X: \rho(x, c) = r_1\} \\ &= \{x_1, x_3, x_4\} \cap \{x_1, x_3\} \cap \{x_1, x_3\} \\ &= \{x_1, x_3\}. \end{aligned}$$

So the system is neither selective nor complete, because card $X_\varphi = 2$ and there are empty informations in the system, for example $\varphi'(a) = p_1$, $\varphi'(b) = q_1$, $\varphi'(c) = r_1$.

Let $S = \langle X, A, V, \rho\rangle$ be an information system. We define two binary relations $\tilde{a}$ $(a \in A)$, and $\tilde{S}$ on S in the following way:

$$x \tilde{a} y \quad \text{iff} \quad \rho(x, a) = \rho(y, a),$$
$$x \tilde{S} y \quad \text{iff} \quad \rho_x = \rho_y.$$

Two objects are in the relation $\tilde{a}$ iff they are undistinguishable with respect to the attribute a; and similarly, two objects are in the relation $\tilde{S}$ if they have the same information in S (i.e. they are undistinguishable with respect to every attribute $a \in A$).

In the recent example $x_1 \tilde{a} x_4$ (x_1, x_4 are undistinguishable with respect to the attribute a because $\rho_{x_1}(a) = \rho_{x_4}(a)$) and objects x_1, x_3 are undistinguishable with respect to every attribute in A; i.e. $x_1 \tilde{S} x_3$, because $\rho_{x_1} = \rho_{x_3}$.

It is easy to check that:

For every information system $S = \langle X, A, V, \rho\rangle$, $\tilde{a}$, $\tilde{S}$, are equivalence relations on X and

$$\tilde{S} = \bigcap_{a \in A} \tilde{a}.$$

In particular if $B \subset A$ then $\tilde{B} = \bigcap_{b \in B} \tilde{b}$.

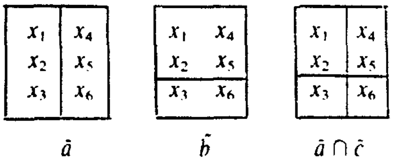
The equivalence classes of the relation $\tilde{S}$ will be called ***elementary*** (***atomic***) ***sets*** in S or when X is fixed, elementary (atomic) sets. The family of all elementary sets in S will be denoted by E_S.

Example 3

Let $S = \langle X, A, V, \rho\rangle$ be an information system defined as follows

X	a	b
x_1	p_1	q_1
x_2	p_1	q_1
x_3	p_1	q_2
x_4	p_2	q_1
x_5	p_2	q_1
x_6	p_2	q_2.

The partitions generated by the attributes are depicted below



Thus the partition $\tilde{a}$ consists of two equivalence classes

$$\{x_1, x_2, x_3\}, \{x_4, x_5, x_6\};$$

Partition $\tilde{b}$ gives also two equivalence classes

$$\{x_1, x_2, x_4, x_5\}, \{x_3, x_6\}.$$

and the product partition $\tilde{S} = \tilde{a} \cap \tilde{b}$ consists of four elementary sets

$$\{x_1, x_2\}, \{x_4, x_5\}, \{x_3\}, \{x_6\}.$$

That is to say if we classified objects of a given set by means of all attributes and their values (descriptors) we automatically introduce a partition of the set of all objects. In each equivalence class (elementary set) of this partition there are objects which are undistinguishable in the system. In general each elementary set contains more than one element. (The system is not selective.) That is to mean the "description power" of a chosen set of attributes and its values is not strong enough to describe every single member of the set X.

Let us observe that if φ, ψ are different informations in the system S, then

$$X_\varphi \cap X_\psi = \phi,$$

$$\bigcup_{\varphi \in \text{Inf}(S)} X_\varphi = X,$$

and if φ is not empty information then X_φ is an elementary set in S. In other words all informations generate a partition of the set X, and this is exactly the partition generated by the relation $\tilde{S}$.

Thus with every elementary set in S we can associate exactly one information in S, and conversely, with every information in S we can associate exactly one elementary set in S (possibly an empty set).

Let $S = \langle X, A, V, \rho\rangle$ be an information system. We shall define a new information system $S^* = \langle E_S, A, V, \rho^*\rangle$, called the *representation* of the system S, where

$$\rho^*: E_S \times A \to V$$

and

$$\rho^*(e, a) = V, \quad e \in E_S, \quad a \in A$$

if and only if

$$\rho(x, a) = V$$

for all $x \in e$.

In other words if we remove all duplicate rows in the table S and replace objects by elementary sets containing this objects in the table S, so we obtain representation of the system S.

For example if the system S is given by the table

X	a	b	c
x_1	u_1	v_1	w_2
x_2	u_2	v_3	w_1
x_3	u_2	v_2	w_3
x_4	u_1	v_1	w_2
x_5	u_1	v_1	w_2
x_6	u_2	v_2	w_3

then the representation of S is the system.

E_s	a	b	c
$\{x_1, x_4, x_5\}$	u_1	v_1	w_2
$\{x_2\}$	u_2	v_3	w_1
$\{x_3, x_6\}$	u_2	v_2	w_3.

Thus representation of any system is selective, i.e. each row in the representation occurs only once.

3. DEPENDENCY OF ATTRIBUTES

Often value of some attribute can be derived from values of another attribute.

For example if the value of an attribute AGE is "two years", then the value of the attribute EDUCATION will be "no education", if both attributes are concerning the same person. The problem of dependency of attributes has been studied in relational model (see Aho *et al.*[1]), but we shall define it in somewhat different way.

The formal definition of this relation is the following one.

Let $a, b \in A$ be two attributes in an information system $S = \langle X, A, V, \rho\rangle$.

(a) Attribute b is said to be *dependent* on $a (a \to b)$ in S iff $\tilde{a} \subset \tilde{b}$,

(b) Attributes a, b are called *independent* in S iff neither $\tilde{a} \subset \tilde{b}$ nor $\tilde{a} \supset \tilde{b}$,

(c) Attributes a, b are said to be *equivalent* in S $(a \sim b)$ iff $\tilde{a} = \tilde{b}$.†

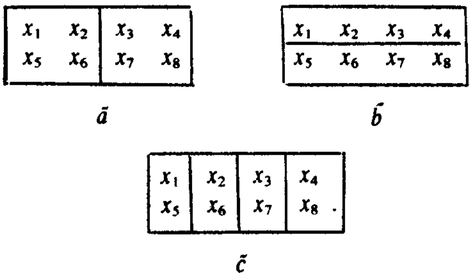
Example 4

Let $S = \langle X, A, V, \rho\rangle$ be an information system defined by the table

X	a	b	c
x_1	p_1	q_1	r_1
x_2	p_1	q_1	r_2
x_3	p_2	q_1	r_3
x_4	p_2	q_1	r_4
x_5	p_1	q_2	r_1
x_6	p_1	q_2	r_2
x_7	p_2	q_2	r_3
x_8	p_2	q_2	r_4.

It is easy to see that $c \to a$, but a, b and c, b are pair-wise independent.

The situation may be depicted as shown below


x_1 x_2 | x_3 x_4
x_5 x_6 | x_7 x_8
$\tilde{a}$
x_1 x_2 x_3 x_4
x_5 x_6 x_7 x_8
$\tilde{b}$
x_1 | x_2 | x_3 | x_4
x_5 | x_6 | x_7 | x_8 .
$\tilde{c}$

Similarly we introduce the relations $B \to a$, $a \to B$, $B \to C$, where B, C are subsets of A.

Attribute a is said to be dependent on the set B of attributes, $B \subset A$, iff $\tilde{B} \subset \tilde{a}$, similarly $a \to B$ iff $\tilde{a} \subset \tilde{B}$. In general we may write $B \to C$ iff $\tilde{B} \subset \tilde{C}$.

Sets of attributes B, C are equivalent $(B \sim C)$ iff $\tilde{B} = \tilde{C}$.

Let us notice that $B \to C$ iff $B \to c_1$, and $B \to c_2$ and $B \to c_k$, and if $b_1 \to C$ or $b_2 \to C$ or $b_l \to C$ then $B \to C$, where $B = \{b_1, b_2, \ldots, b_l\}$ and $C = \{c_1, c_2, \ldots, c_k\}$.

The meaning of the "dependency" relation $B \to C$ is obvious. It simply means that values of the left-hand side attributes determine values of the r.h.s. attributes.

That is to say, if $B \to C$, then there exists one function f (*dependency function*)

$$f: \underset{b \in B}{P} V_b \to \underset{c \in C}{P} V_c$$

such that

$$\rho(x, c)_{c \in C} = f(\rho(x, b)_{b \in B}), \quad \text{for all } x \in X.$$

(P denotes cartesian product). In other words there exists

† If system S is fixed we shall write in short, "b is dependent on a", etc.

one set of functions $(f_c)_{c \in C}$ such that

$$(\bigwedge_{c \in C})(\rho(x, c) = f_c(\rho(x, b)_{b \in B}),$$

and

$$\rho(x, c) = f_c(\rho(x, b)_{b \in B})$$

if

$$X_{c,\rho(x,c)} \supset X_{b_1,\rho(x,b_1)} \cap X_{b_2,\rho(x,b_2)} \cap \cdots \cap X_{b_k,\rho(x,b_k)}$$

for all $x \in X$, where $X_{c,v} = \{x \in X: \rho_x(c) = v\}$.

Example 5

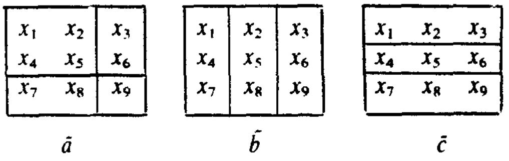
Let $S = \langle X, A, V, \rho \rangle$ be an information system, such that

$$X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\},$$
$$A = \{a, b, c\},$$
$$V_a = \{p_1, p_2, p_3, p_4\},$$
$$V_b = \{q_1, q_2, q_3\},$$
$$V_c = \{r_1, r_2, r_3\}.$$

Assume that the attributes generate the following partitions on X:

$$X_{a,p_1} = \{x_1, x_2, x_4, x_5\},$$
$$X_{a,p_2} = \{x_3, x_6\},$$
$$X_{a,p_3} = \{x_7, x_8\},$$
$$X_{a,p_4} = \{x_9\},$$
$$X_{b,q_1} = \{x_1, x_4, x_7\},$$
$$X_{b,q_2} = \{x_2, x_5, x_8\},$$
$$X_{b,q_3} = \{x_3, x_6, x_9\},$$
$$X_{c,r_1} = \{x_1, x_2, x_3\},$$
$$X_{c,r_2} = \{x_4, x_5, x_6\},$$
$$X_{c,r_3} = \{x_7, x_8, x_9\}.$$

The partitions are shown below



In this system all attributes are pairwise independent but $\{b, c\} \to a$, because $\tilde{b} \cap \tilde{c} \subset \tilde{a}$.

The corresponding "dependency" function f is given by the table below:

V_b	V_c	V_a
q_1	r_1	p_1
q_1	r_2	p_1
q_1	r_3	p_3
q_2	r_1	p_1
q_2	r_2	p_1
q_2	r_3	p_3
q_3	r_1	p_2
q_3	r_2	p_2
q_3	r_3	p_4.

Thus for instance $f(q_2, r_3) = p_3$ and $f(q_3, r_2) = p_2$.

Thus knowing values of attributes b and c we may compute by means of dependency function f value of the attribute a.

It is obvious that if $B \to C$ in S then also $B \to C$ in S^*. So instead of checking whether or not $B \to C$ in S we check the dependency in S^*, which is much simpler, because the table of S^* is much simpler than the table of S.

The question arises whether the dependency $B \to C$ could be deduced from some other known dependencies in S by means of logical inference rules and not by checking the table, of S or S^*. Similar problems have been investigated in relational model of database (see for example Aho *et al.*[14]), but we shall assume here another solution (see Orłowska[15], Jaegermann and Marek[16]).

4. REDUCED SYSTEMS

As we have stated in the previous paragraph some attributes in the information system may be superfluous in this sense that their values can be "derived" from the values of other attributes in the system. We shall consider this question in this paragraph in some details.

Let us first introduce basic definitions.

A subset $B \subsetneq A$ is said to be *independent in S* iff, for every $B' \subset B$, $\tilde{B} \neq \tilde{B}'$.

A subset $B \subsetneq A$ is said to be *dependent* in S iff there exists a $B' \subset B$ such that $\tilde{B}' = \tilde{B}$.

The set B is said to be *derivable from C* in S iff $B, C \subset A$, $C \subset B$ and $\tilde{B} = \tilde{C}$.

One can easily verify the following properties:

(a) If $B \subsetneq A$ is the greatest independent set in S then for every $a \in A - B$, $B \to a$.

(b) If B is dependent in S then there exists $B' \subset B$ independent in S for such that every $a \in B - B'$, $B \to a$.

(c) If $B \subset A$, then $A \to B$.

Example 6

Let $S = \langle X, A, V, \rho \rangle$ be an information system such that $X = \{x_1, x_2, x_3, x_4, x_5\}$ and $A = \{a, b, c, d\}$.

Assume that the attributes generate the following partitions on X:

$\tilde{a} = \{x_1, x_2, x_5\}, \{x_3, x_4\}$, $\tilde{c} = \{x_1, x_2, x_3, x_4\}, \{x_5\}$,

$\tilde{b} = \{x_1\}, \{x_2, x_3, x_4, x_5\}$, $\tilde{d} = \{x_1\}, \{x_3, x_4\}, \{x_2, x_5\}$.

Of course, the partition generated by attribute a is the set of all equivalence classes of the relation $\tilde{a}$, i.e. the partition on X defined by the relation $\tilde{a}$.

It is easy to see that the whole set of attributes A determine the partition $\{x_1\}, \{x_2\}, \{x_3, x_4\}, \{x_5\}$. Now, we have the following relationship between the attributes: $d \to b$ and $d \to a$ (because $\tilde{d} \subset \tilde{b}$ and $\tilde{d} \subset \tilde{a}$).

Also $\{a, b, c\} \to d$ because $\tilde{a} \cap \tilde{b} \cap \tilde{c} \subset \tilde{d}$; and $\{c, d\} \to \{a, b\}$, because $\{c, d\} \to a$, and $\{c, d\} \to b$, i.e. $\tilde{c} \cap \tilde{d} \subset \tilde{a}$ and $\tilde{c} \cap \tilde{d} \subset \tilde{b}$.

The set A is dependent in S because there exists $B \subset A$, $B = \{a, b, c\}$ such that $\tilde{B} = \tilde{A}$. There are also other subsets $C, D \subset A$ of attributes $C = \{a, c, d\}$, $D = \{c, d\}$, with the same property, e.g. $\tilde{C} = \tilde{D} = \tilde{A}$.

Sets B and D are independent in S whereas C is not because $\tilde{C} = \tilde{D}$.

As we have seen from previous consideration some attributes may be sometimes eliminated from the system, and one can derive their values from the remaining set of attributes. This is to mean that they are superfluous in the system. This leads to the following definition.

Let $S = \langle X, A, V, \rho\rangle$ be an information system. A set $A' \subset A$ will be called a ***reduct of** A* if $\tilde{A} = \tilde{A}'$, and there does not exist a proper subset B of A' such that $\tilde{B} = \tilde{A}'$. The corresponding system $S' = \langle X, A', V, \rho'\rangle$ is called *reduced system*. (ρ' is the restriction of the function ρ to the set $X \times A'$.)

It is clear that a system can turn out to have more than one reduct. In Example 6, we have two reducts of A, namely B and D.

It is easy to prove the following properties:

(a) If an information system is maximal then it is also reduced (the converse implication does not hold (see Example 7)).

(b) If an information system is reduced then all its different attributes are pairwise independent. (The converse implication does not hold (see Example 8).)

(c) Two information systems S, S' with the same set of objects X, are said to be equivalent iff $\tilde{S} = \tilde{S}'$.

For every information system S there exists a reduced system S' equivalent to S.

Let us also notice, that if S is reduced, then also S^* is reduced.

Example 7

Let $S = \langle X, A, V, \rho\rangle$ be an information system

X	a	b
x_1	p_1	q_1
x_2	p_2	q_1
x_3	p_1	q_2.

The system is reduced but is not complete since for a, b such that $\varphi(a) = p_2$ and $\varphi(b) = p_2$, $X_\varphi = \phi$.

Example 8

Let $S = \langle X, A, V, \rho\rangle$, where $X = \{x_1, x_2, x_3, x_4\}$, $A = \{a, b, c\}$ and the attributes determine the following partition of the set X:

$$\tilde{a} = \{x_1, x_2\}, \{x_3, x_4\},$$

$$\tilde{b} = \{x_1\}, \{x_2, x_3, x_4\},$$

$$\tilde{c} = \{x_2\}, \{x_1, x_3, x_4\}.$$

The attributes a, b, c are pairwise independent, while $\{a, b\}$, $\{a, c\}$ and $\{b, c\}$ are reducts of A.

The idea of reduction of an attribute set in a system is of great practical importance, because it shows that one can get sometimes the same information from the system with smaller set of attributes. This may have special meaning in the case when attributes are symptoms of some illness but in order to get the proper diagnosis it is not necessary to investigate all symptoms, but try to find only those which are really necessary. In fact there can be more than one set of such minimal symptoms (see example 6).

The problem arises how to find effectively reducts of a given information system. Because all sets in the system are finite such an algorithm always exists, however it may be not very efficient in general.

Some considerations concerning this subject one can find in Łoś[17], Truszczyński[18], Grzymała-Busse[19].

5. SUBSYSTEMS

In this section we shall introduce and discuss the notion of *subsystem* of a given information system.

Let $S = \langle X, A, V, \rho\rangle$ and $S' = \langle X', A', V', \rho'\rangle$ be two information systems. We say that S' is a subsystem of S if $X' \subset X$, $A' \subset A$, $V' \subset V$ and $\rho' = \rho/X' \times A'$.

If S' is a subsystem of S, then we shall write $S' < S$ or $S' \underset{X', A'}{<} S$, or $S' = S/X' \times A'$.

In other words if we remove from the table S some columns or rows then the remaining table is the subsystem of the system S.

For example if in the system

X	a	b	c
x_1	v_1	u_2	w_1
x_2	v_2	u_1	w_2
x_3	v_1	u_2	w_2
x_4	v_1	u_2	w_1,

we drop the column b and the row x_3 then we obtain a subsystem of S

X'	a	c
x_1	v_1	w_1
x_2	v_2	w_2
x_4	v_1	w_1.

We shall introduce two kinds of subsystems.

If $S' < S$ and $X' = X$, then we shall say that S' is an *attribute restricted* subsystem of S, in symbols $S' \underset{A'}{<} S$ or $S' = S/A$.

If $S' < S$ and $A' = A$, then we shall say that S' is an *object restricted* subsystem of S, in symbols $S' \underset{X'}{<} S$ or $S' = S/X'$.

Thus if S is an information system and we drop some column from it, then the obtained system is an attribute restricted subsystem of S, and if we remove some rows from the system S—we obtain object restricted subsystem of S.

For example if in the system

X	a	b	c
x_1	v_1	u_2	w_1
x_2	v_2	u_1	w_2
x_3	v_1	u_2	w_2
x_4	v_1	u_1	w_1

we remove column b we obtain attribute restricted subsystem of S

X	a	c
x_1	v_1	w_1
x_2	v_2	w_2
x_3	v_1	w_2
x_4	v_1	w_1

and if we remove from S the row x_3 we obtain object restricted subsystem of S

X'	a	b	c
x_1	v_1	u_1	w_1
x_2	v_2	u_1	w_2
x_4	v_1	u_2	w_1.

Now we shall give some elementary properties of subsystems.

If $S' = S/X'$ then $\tilde{S}' = \tilde{S} \cap (X')^2$.

If $S' = S/A'$, then $\tilde{S}' \supset \tilde{S}$.

If $S' < S$ and S is reduced then S' is also reduced.

If $S' = S/X'$ and S is maximal then S' may be not maximal.

If $S' = S/A'$ and S is maximal then S' is maximal.

If $S' = S/A'$ and S is selective then S' may not be selective.

If $S' = S/X'$ and S is selective then S' is selective.

If $S' = S/X'$ then

$$(S')^* = S^*/X'.$$

If $S' = S/A'$, then

$$(S')^* \neq S^*/A'.$$

If $S' = S/X', A'$ then there exist exactly one system $S_1 = S/X'$ and $S_2 = S/A'$ such that $S' = S_1/A' = S_2/X'$.

6. CONNECTION OF INFORMATION SYSTEMS

Very often we face the following problem. We are given some information systems $S_1, S_2, \ldots, S_k$ and we want to have one "common" information system S combining all systems $S_1, S_2, \ldots, S_k$ into one. The system S will be called *connection* of systems S_i, $i = 1, 2, \ldots, k$, and will be denoted as $S = \bigcup_{i=1}^{k} S_i$.

Let $S = \langle X, A, V, \rho \rangle$ and $S_i = \langle X_i, A_i, V_i, \rho_i \rangle$, $i = 1, \ldots, k$.

The system S is a connection of system S_i if the following conditions are satisfied:

$$X = \bigcup_{i=1}^{k} X_i,$$

$$A = \bigcup_{i=1}^{k} A_i,$$

$$V = \bigcup_{i=1}^{k} V_i,$$

$$\rho / X_i \times A_i = \rho_i, \quad i = 1, \ldots, k,$$

$$\rho_x = \bigcup_{i=1}^{k} \rho_{i_x}, \quad x \in X.$$

Connection $S = \bigcup_{i=1}^{k} S_i$ is well defined if the following two conditions are valid:

(1) If $(X_i \cap X_j) \neq \phi$ and $(A_i \cap A_j) \neq \phi$ then

$$\rho_i/(X_i \cap X_j) \times (A_i \cap A_j) = \rho_j/(X_i \cap X_j) \times (A_i \cap A_j),$$

for all $i, j = 1, \ldots, k$, and

(2) $\rho_x = \bigcup_{i=1}^{k} \rho_{i_x}$ is defined for all $x \in X$ and $a \in A$.

Of course systems S_i are subsystems of S. The first condition is obvious and the second needs some explanation.

Let S_1 be a system with only one attribute, say colour, and S_2 a system also with one attribute, for example, length, and assume that $X_1 \cap X_2 = \phi$. The second condition says that we are not allowed to define connection S of S_1 and S_2 because we do not have any information about lengths of objects in S_1 or about colours of objects in S_2. Thus we are unable to define for all $x \in X$ the information about colour and length of x. In other words, we are not able to define the function ρ_x for the connection $S = S_1 \cup S_2$.

This seems to have natural justification in real life systems. If we have two information systems, say first concerning insurance and the second medical care with different sets of population, for example, one in London and the second in Warsaw, then combining those two systems into one connected system is justified only in the case when we have insurance data in the medical system and conversely. Otherwise we are unable to define for all $x \in X$ the information ρ_x about insurance and medical care and, consequently, according to our definition, the connection of these two systems is not an information system.

Let us consider very simple formal example depicting above situation more clearly.

The connection of the two following systems

X	a	b	c
x_1	v_1	u_1	w_2
x_2	v_1	u_2	w_1
x_3	v_2	u_1	w_2
x_4	v_2	u_1	w_2

Y	c	d	e
x_3	w_2	p_1	q_1
x_4	w_2	p_2	q_1
y_1	w_1	p_3	q_1
y_2	w_2	p_1	q_2

is the table

$X \cup Y$	a	b	c	d	e
x_1	v_1	u_1	w_2	—	—
x_2	v_1	u_2	w_1	—	—
x_3	v_2	u_1	w_2	p_1	q_1
x_4	v_2	u_1	w_2	p_2	q_1
y_1	—	—	w_1	p_3	q_1
y_2	—	—	w_2	p_1	q_2

which is not an information system according to our definition because some values of attributes are undefined in the table. That is to say function defined by the table is not total but partial, which is not allowed in our definition of an information system.

This property leads to a definition of two special kinds of connections of information systems.

If $S = \cup S_i$ and $S_i = S/A_i$ then S will be called *attribute connected system.*

If $S = \cup S_i$ and $S_i = S/X_i$, then S will be called *object connected system.*

These two kinds of connections are depicted by the following example.

Example 9

Let S_1, S_2 be two information systems with the same set of objects and different sets of attributes as shown below:

X	a	b	c
x_1	u_1	v_1	w_2
x_2	u_1	v_2	w_1
x_3	u_2	v_1	w_2
x_4	u_1	v_1	w_2

X	a	d	e
x_1	u_1	p_1	q_2
x_2	u_1	p_2	q_1
x_3	u_2	p_1	q_1
x_4	u_1	p_1	q_2.

Connection of S_1 and S_2 is given below

X	a	b	c	d	e
x_1	u_1	v_1	w_2	p_1	q_2
x_2	u_1	v_2	w_1	p_2	q_1
x_3	u_2	v_1	w_2	p_1	q_1
x_4	u_1	v_1	w_2	p_1	q_2.

Let S_3, S_4 be two information systems with different sets of objects but the same set of attributes as shown in the tables

X	a	b	c
x_1	u_1	v_1	w_2
x_2	u_2	v_2	w_1
x_3	u_1	v_2	w_1
x_4	u_1	v_1	w_1

Y	a	b	c
x_3	u_1	v_2	w_1
x_4	u_1	v_1	w_1
y_1	u_2	v_2	w_2
y_2	u_1	v_2	w_1.

Connection of S_3 and S_4 is the system

$X \cup Y$	a	b	c
x_1	u_1	v_1	w_2
x_2	u_2	v_2	w_1
x_3	u_1	v_2	w_1
x_4	u_1	v_1	w_1
y_1	u_2	v_2	w_2
y_2	u_1	v_2	w_1.

Attribute connected system corresponds to the situation when all constituent information systems have the same set of objects but different set of attributes. For example if we have in the same town different information systems owned by insurance company, medical care service, bank office, police etc. then we may combine them into one information system. The set of objects in those systems are the same (all inhabitants of the town) but the set of attributes in all systems are different.

Object connected information system refers to the situation when all constituent systems have the same set of attributes but different sets of objects. For example if the same company, say insurance company, own information systems in different districts. Thus we have the case when the set of attributes in each system is the same but the objects (inhabitants of the districts) are different. So we can consider all these systems as an attribute connected system.

Now we shall give some elementary properties of the "connection" operation.

Let $S = \langle X, A, V, \rho \rangle$ and $S_i = \langle X_i, A_i, V_i, \rho_i \rangle$, $i = 1, \ldots, k$ be information systems and let $S = \bigcup_{i=1}^{k} S_i$.

If $S = \cup S_i$, $S_i = S/A_i$ and each S_i is reduced, then S may not be reduced.

If $S = \cup S_i$, $S_i = S/X_i$ and each S_i is reduced, then S is also reduced.

If$S = \cup S_i$, $S_i = S/A'$, then $\tilde{S} = \bigcap_{i=1}^{k} \tilde{S}_i$.

If $S = \cup S_i$, $S_i = S/X_i$, then $X_\varphi = \bigcup_{i=1}^{k} X_{i,\varphi}$ for all $\varphi \in \operatorname{Inf}(S)$.

If $S = \cup S_i$, $S_i = S/A_i$ and each S_i is reduced then S may not be reduced.

If $S = \cup S_i$, $S_i = S/X_i$, and each S_i is reduced, then S is also reduced.

If $S = \cup S_i$, $S_i = S/X_i$, the $S^* \neq \cup S_i^*$.

If $S = \cup S_i$, $S_i = S/A_i$, then $S^* = \cup S_i^*$.

If $S = \cup S_i$, $S_i = S/A_i$ and each S_i is selective, then S is also selective.

If $S = \cup S_i$, $S_i = S/X_i$ and each S_i is selective then S may not be selective.

If $S = \cup S_i$ and each S_i is maximal then S is also maximal.

There are systems S_i such that

$$(\cup S_i)^* \neq \cup (S_i)^*.$$

7. THE LANGUAGE OF AN INFORMATION SYSTEM (QUERY LANGUAGE)

With each information system S we shall associate a query language L_S, which will be used for asking queries about informations contained in the system S. A query submitted to the system can be either a term or a formula. Terms are interpreted as subsets of the set of objects in the system (documents or records), whereas formulas are interpreted as truth values (truth and falsity). In the first case the response to the query is the subset of objects relevant to the query, and in the second case the response is "yes" or "not".

In the paragraph we define a language tailored to meet the requirements mentioned above.

First the syntax and the semantics of the language will be introduced and then some properties of the language are stated.

Let us first define the set of terms T_S of the query language L_S. Terms are built up from constants 0, 1 and descriptors combined by means of symbols for boolean operations $\sim, +, \cdot$.

More exactly, the set of terms of the query language L_S is the least set satisfying the conditions:

(1) 0, 1 and all descriptors of S are terms in L_S.

(2) If t, s are terms in L_S then so are $\sim t$, $t + s$, $t \cdot s$. Parentheses are used, if necessary, in the obvious way.

The following expressions are terms in some query language:

(NAME = Smith)
(AGE = middle) + (SEX = female)
((PROFESSION = Clerk) (AGE = young))
~ ((SALARY = high) + 1)

Now we shall define the set of formulas F_S of the query language L_S. Formulas are built up from simple formulas of the form $t = s$, where t, s were terms in L_S, and symbols T, F (truth, falsity) by means of logical connectives $-, \vee, \wedge$.

More exactly the set of formulas of the query language L_S is the least set satisfying the conditions:

(1) T, F are formulas; if t, s are terms in L_S, then $t = s$ is a formula in L_S.

(2) If ϕ, ψ are formulas in L_S, then $-\phi$, $\phi \vee \psi$, $\phi \wedge \psi$ are formulas in L_S.

Parentheses are used if necessary in the same way as in the case of terms.

The following are examples of formulas in some query language:

(NAME = Smith) = (AGE = middle);
(AGE = old) + (PROFESSION = clerk) = 1;
((SALARY = low) = (AGE = middle))

~ (PROFESSION = farmer).

Now we shall define the semantics of the language L_S, which assigns a subset of objects to each term and a truth value to each formula. We shall define the semantics in two steps, first for terms and second for the formulas.

Semantics of terms is a function σ_S (or σ when S is fixed) from terms into subsets of objects, i.e. $\sigma_S: T_S \to p(X)$, defined as follows:

(1) $\sigma(0) = \phi$, $\sigma(1) = X$,

(2) $\sigma(a, v) = \{x \in X: \rho_x(a) = v\}$,

(3) $\sigma(\sim t) = X - \sigma(t)$,
$\sigma(t + s) = \sigma(t) \cup \sigma(s)$,
$\sigma(t \cdot s) = \sigma(t) \cap \sigma(s)$.

It is clear from the above definition that the answer to each query which is a term is some subset of the set of all objects having the property stated by the term.

For example the answer to the query

(AGE = middle) · (SEX = female)

in some information system S is the set of all middle age women being registrated in the information system S. Of course the answer to the same query in another information system may be different.

Thus by means of the definition of the semantics function we are able to compute the answer to any query term in every information system.

Similarly we can define how to compute answers to queries which are formulas. Semantics of formulas we shall also denote by σ_S (or simple σ when the system S is fixed).

Semantics of formulas is a function σ_S assigning to each formula its truth value T, F, i.e. σ is a function from F_S into $\{T, F\}$, such that:

(1) $\sigma(T) = T$, $\sigma(F) = F$,

(2) $\sigma(t = s) = \begin{cases} T, & \text{if } \sigma(t) = \sigma(s), \\ F, & \text{otherwise.} \end{cases}$

(3) $\sigma(-\phi) = \begin{cases} T, & \text{if } \sigma(\phi) = F, \\ F, & \text{if } \sigma(\phi) = T. \end{cases}$

(4) $\sigma(\phi \vee \psi) = \sigma(\phi) \vee \sigma(\psi)$,

(5) $\sigma(\phi \wedge \psi) = \sigma(\phi) \wedge \sigma(\psi)$.

It is easily seen from the definition of the semantics of formulas that the answer to the query which is a formula is truth or falsity (yes or no). For instance if the query is of the form: (AGE = middle) = (SALARY = low), then the answer to this query is "truth" if each middle age person has a low salary, otherwise the answer is "falsity".

Of course the answer in this case is again related to same information system S. The answer to the same query in another information system may be different.

So by means of rules (1)–(5) we are able to compute the value of any formula in every information system S.

We can also compute answer to the query using the "normal form" property of terms and formulas. This is often much simpler than the method based directly on the definition of semantics of terms and formulas.

First we shall discuss the problem of transforming terms into normal form.

Let us first introduce some notions.

Let $A = \{a_1, a_2, \ldots, a_n\}$. A term t is said to be *elementary* if it is of the form: $(a_1, v_1) \cdot (a_2, v_2) \cdot \ldots \cdot (a_n, v_n)$ where $v_i \in V_{a_i}$, for $i = 1, \ldots, n$.

The following are examples of elementary terms in some information system:

(SEX = male) · (SALARY = high) · (AGE = young),
(SEX = female) · (SALARY = low) · (AGE = old).

A term t is said to be *normal* if it is 0, 1 or of the form: $t_1 + t_2 + \cdots + t_k$, where $k \geq 1$ and each t_i is elementary.

The set $Y \subset X$ $(Y \neq \phi)$ is said to be *elementary* iff there exists an elementary term t such that $\sigma(t) = Y$.

Let us observe that each elementary set is an equivalence class of the relation $\tilde{S}$.

So elementary terms are linguistic representatives of informations in our system. Of course the following is true:

If t, s are different elementary terms, then

$$\sigma(t) \cap \sigma(s) = \phi$$

$$\bigcup_{t \in T_{el}} \sigma(t) = X,$$

where T_{el} is the set of all elementary terms in S.

Let $t, s \in T_S$. We say that t and s are *equivalent* in S iff $\sigma(t) = \sigma(s)$.

For every $t \in T_S$, there exists a term s in normal form which is equivalent to t in S.

(Let us note that negation does not occur in normal form term. This can be done because of the finiteness of the sets of values of attributes, e.g. one can say instead of "not red", "green", or "blue", or "white", etc., exhausting all possible colours.)

This normal form property says that the answer to any query which is a term is simply the union of some elementary sets in S.

The normal form property also says that if the system is not selective we are unable to describe by means of terms every subset of objects in the system, but only those subsets of X which are unions of elementary sets.

This leads to a definition of a *describable* set in L_S. The subset $Y \subset X$ is called *describable* in L_S iff there exists term t in L_S such that $\sigma(t) = Y$.

Describable sets are only possible answers in the system S. Thus the "description power" of the query language of the system is limited, because we are unable to express in the language of the system in general case any property of objects, i.e. describe any subset of objects—and it does not matter how the system is implemented.

The notion of a describable set can be used to define the *accuracy* of the query language L_S as follows:

$$\lambda_S = \frac{2^k}{2^{\mathrm{card}(X)}},$$

where k is the number of elementary sets in the system S and X is the set of objects in the system S.

Thus the coefficient λ expresses the ratio of the number of all describable sets in the systems to the number of all subsets of objects in S. In other words, the "accuracy" coefficient λ, expresses the ratio of all properties of objects in the system, which are expressable in the language of this system to all possible properties. (We identify the notion of a property with that of a subset.) Let us remark that $\lambda \leq 1$, and $\lambda = 1$ only for selective systems, and this is the greatest possible accuracy.

We can also introduce the notion of *efficiency* of the language of an information system. The efficiency coefficient will be defined as

$$\mu_S = \frac{k}{\prod_{a \in A} \mathrm{card}(V_a)} = \frac{k}{\mathrm{card}(\mathrm{Inf}\,(S))},$$

where k is the number of elementary sets in the system L_S. Thus the efficiency coefficient of the language L_S is the ratio of all elementary sets in the system S (or nonempty informations, or nonempty elementary terms) to all informations (elementary terms) in the language L_S. Of course $0 \leq \mu \leq 1$, and $\mu = 1$ for selective systems.

Thus effectiveness of the language is due to the fact what part of the language has a meaning in the system. In other words, efficiency of the language L_S is:

$$\mu_S = \frac{\text{number of nonempty elementary terms in } L_S}{\text{number of all elementary terms in } L_S}.$$

Let us remark that if we know the number of elements in each elementary set in the system we can simply compute from the normal form of a query the number of elements in the answer, because

$$\mathrm{card}(\sigma(t)) = \mathrm{card}(\sigma(t_1)) + \mathrm{card}(\sigma(t_2)) + \cdots + \mathrm{card}(\sigma(t_k)),$$

where the normal form of t is $t_1 + t_2 + \cdots + t_k$.

We can also introduce another measure of the size of elementary sets,

$$p_x(t) = \frac{\mathrm{card}(\sigma(t))}{\mathrm{card}(X)},$$

where t is an elementary term.

This measure can be interpreted as a probability that an object $x \in X$ has the property t, i.e. belongs to the set $\sigma(t)$. Then the probability that an object $x \in X$ has the property t is

$$p_x(t) = \frac{\sum_{i=1}^{k} \mathrm{card}(\sigma(t_i))}{\mathrm{card}(X)},$$

where t has the normal form $t_1 + t_2 + \cdots + t_k$. So we can get the number of relevant objects to the query without retrieving them first from the memory of a computer.

Now let us return to the question how to transform terms to normal forms. In order to do this we need some transformation rules preserving equivalence of terms—which are in fact axioms of our query language.

As axioms for terms we assume substitutions of terms into the axioms of Boolean algebra (e.g. $\sim(\sim t)=t$, $t+0=0$, $t+s=s+t$ etc.) and the following specific axioms

(1) $(a, v)\cdot(a, v')=0$ if $v, v'\in V_a$ and $v\neq v'$,

(2) $\sum_{v'\in V_a}(a, v)=1$,

(3) $\sim(a, v)=\sum_{v'\in V_a}(av')$, $v\neq v'$,

(4) $\sum_{t_i\in T_{el}} t_i=1$.

Example 10

Let us consider information system S in which there are three attributes a, b, c, with the following domains

$$V_a=\{v_1, v_2\},\quad V_b=\{u_1, u_2\},\quad V_c=\{w_1, w_2, w_3\}.$$

The term

$$t=(a, v_1)\cdot(b, u_2)+\sim(c, w_2)$$

in the language L_S has the following normal form

$$\begin{aligned}t=&(a, v_1)\cdot(b, u_2)\cdot(c, w_1)+(a, v_1)\cdot(b, u_2)\cdot(c, w_2)\\&+(a, v_1)\cdot(b, u_2)\cdot(c, w_3)+(a, v_1)\cdot(b, u_1)\cdot(c, w_1)\\&+(a, v_1)\cdot(b, u_2)\cdot(c, w_1)+(a, v_2)\cdot(b, u_1)\cdot(c, w_1)\\&+(a, v_2)\cdot(b, u_2)\cdot(c, w_1)+(a, v_1)\cdot(b, u_1)\cdot(c, w_3)\\&+(a, v_1)\cdot(b, u_2)\cdot(c, w_3)+(a, v_2)\cdot(b, u_1)\cdot(c, w_3)\\&+(a, v_2)\cdot(b, u_2)\cdot(c, w_3).\end{aligned}$$

This is to mean that in order to get an answer to the query

$$t=(a, v_1)\cdot(b, u_2)+\sim(c, w_3)$$

we have to take union of all elementary sets corresponding to all elementary terms occuring in normal form of the query t. This is much simpler than computing the answer directly from the definition of the semantic function because we avoid taking the intersection, and complement operation on sets, which are very unefficient in computer implementation (they require access to files stored in slow memory). So by transforming the query to normal form we omit this inconvenience, and the transformation to normal form can be done very fast.

In real life information systems very many elementary terms are empty. In order to obtain normal form without empty terms, which are superfluous, we have to use in axiom (4) only nonempty elementary terms in the considered information system.

We can also give rules to transform formulas to normal forms, however there is no big difference in efficiency (in opposite to the case of terms) in computation the truth-values of a formula using directly the definition of semantic function and the normal form approach.

The only problem is how to compute the truth-values of elementary formulas, i.e. formulas of the form $t=s$. Checking equality of two sets is very unefficient operation again, but we have very simple property, which allow to avoid this operation, namely

$$t=s \quad \text{iff} \quad t_1=0\wedge t_2=0\wedge\cdots\wedge t_k=0$$

where $t_1, t_2, \ldots, t_k$ are elementary terms occuring in t or s but not in both.

8. REMARKS ON IMPLEMENTATION

The mathematical model of an information system presented in this paper leads to a new, simple and efficient method of information retrieval.

Let us suppose that data are clustered in computer memory in such a way that in each cluster there are data with the same information, so that each cluster forms an elementary set. Then in order to find an answer to a query it is enough to transfer the query to normal form and then fihd the proper clusters (elementary sets) associated with each elementary term.

There are however three practical problems when implementating this idea.

The first problem is due to the number of elementary sets in the system. If A is the set of attributes in the information system S, then there are at most

$$\prod_{a\in A}\operatorname{card}(V_a)$$

elementary sets in the system S.

For example for ten attributes and ten values of each attribute we have 10^{10} elementary sets.

So we can not use the method literally, because organizing data in this way is of course unrealistic. However we may assume as a basis for elementary set organization not all attributes occuring in the system but some of them only. In this way we obtain attribute restricted subsystem in which elementary sets are bigger as in the original system and consequently their number can be reduced to "reasonable" size. Reducing the number of attributes in the system we get not exact but approximate answers only. Thus in order to get the final answer we must add one step more in the retrieval process, in which, on the basis of all attributes, the proper answer is searched in the reduced memory space determined in the first step. The second step can be organized for example as a linear search.

We can also reduce the number of elementary sets introducing new attributes to the system. Let us consider fo example the attribute NAME. If we introduce attribute FIRST LETTER OF NAME, and organize elementary sets on the basis of the second attribute instead—the first one, we get similar effect on the elementary sets as in the previous case.

So we have two possibilities to reduce the number of elementary sets, and we may exploit both of them at the same time.

Next important practical problem is how to find effectively the elementary sets. Proposed algorithm is based on a proper enumeration of elementary sets, or what is the same—enumeration of elementary terms.

Special number system has been proposed, in which we index the attributes and values within each attribute by nonnegative integers, and treat then the elementary terms as numbers in thus obtained number system. (For detail see Marek and Pawlak[2].) Then by means of address table, to each elementary set (its number) an address of the corresponding elementary set in the storage is assigned.

Thus having the normal form of a query, through the enumeration and address table, we can find directly the locations where the answer to this query is stored.

The third important practical problem is due to the normal form transformation algorithm. We need an algorithm which produces nonempty elementary sets, only, and this is not a very difficult task, which can be solved by standard methods.

Detailed description of the information retrieval system based on this idea one can find in Margański[7].

9. DISTRIBUTED INFORMATION SYSTEMS

Very often we are interested in decomposing information systems into "smaller" ones or vice versa—combine some number of "small" systems into bigger ones. In both cases as a result we obtain distributed information system.

We shall deal in this paragraph with some logical questions connected with the problem how to find answers in distributed systems—on the basis of the presented approach to information systems theory.

Let us first consider the problem informaly.

We are given n "local" information systems $S_1, S_2, \ldots, S_n$. With each system S_i there is associated a query language L_{S_i} (or shortly L_i). If t is a query in L_i, then the answer to this query will be denoted by $\sigma_{S_i}(t)$ (or simply by $\sigma_i(t)$).

We may combine systems $S_1, S_2, \ldots, S_n$ in one system, which we shall denote by S, and call global information system. With the global system S we may associate a global language L_S (or in short L). The global language L may be viewed as a certain combination of the languages L_i (local languages).

The question arises whether we can obtain an answer to a query in global language (global query), by means of combining the local answers to local queries, that is whether the answer to the query t may be presented as a function of local answers, i.e.

$$\sigma_S(t) = h(\sigma_{S_1}(t_1), \ldots, \sigma_{S_2}(t_2), \ldots, \sigma_{S_n}(t_n)),$$

where t_i is a "projection" of the query t on the language L_i, and h is some function.

That is to mean that in order to answer a global query t we replace the query t by some queries $t_1, t_2, \ldots, t_n$, referring to corresponding local systems, and afterwards we form the global answer from the local answers obtained in this way. For example, let us assume that we have information systems owned by an insurance company, medical care service, bank office, police etc. In each such a system we may answer specific queries related to the need of the owner of the system, like "list all persons with the saving account greater than 10,000\$" or "list all persons which had a car accident in 1977" etc. Each such a query is related to specific information system, in which the relevant informations are stored. However, it may happen that some system user may be also interested in obtaining informations from several various systems, for example he may ask "whether there are persons who have caused a car accident while being treated with some drugs". This kind of queries cannot be answered by searching files in only one information system. This information is distributed at least in two systems: medical care system and insurance company system (or police system). Thus in order to get an answer to such a query we have to search for some information in more than one system.

The problem stated above is connected with another one: whether every local user is allowed to ask general queries or, in other words, whether any local user has access to informations stored in another system. If not, and this is widely used practice, the question is how to restrict access to protected informations against an unauthorised user. This problem will not be considered here, however we shall make some remarks concerning this subject.

In order to consider this problem formally let us first introduce the notion of an approximate answer to the query.

Let $S = \langle X, A, V, \rho \rangle$ and $S' = \langle X, A', V', \rho' \rangle$ be two information systems. Let S' be attribute restricted subsystem of S, and let Y and Z be describable set in S and S' respectively.

The set Z will be called the *least upper approximation* of Y in S' if Z is the least set including Y ($Y \subseteq Z$).

If Z is the least upper approximation of Y in S' we shall write

$$Z = LUA_{S'}(Y)$$

It is easy to see that

$$LUA_{S'}(Y) = \bigcup_{i=1}^{n} LUA_{S'}(Y_i),$$

where

$$Y = \bigcup_{i=1}^{n} Y_i$$

and Y_i are elementary sets in S.

Of course if Y is an elementary set in S then $LUA_{S'}(Y)$ is also an elementary set in S', including Y ($Y \subset LUA_{S'}(Y)$).

From the definition of elementary sets follows that each describable set in S has exactly one LUA in S'.

Let $S = \langle X, A, V, \rho \rangle$ and $S' = \langle X', A, V, \rho' \rangle$ be information systems. Let S' be an object restricted subsystem of S and let Y, Z be describable sets in S and S' respectively.

The set Z will be called the *greatest lower approximation* of Y in S' if Z is greatest set including Y ($Z \subseteq Y$).

If Z is the greatest lower approximation of Y in S' we shall write

$$Z = GLA_{S'}(Y).$$

Obviously

$$GLA_{S'}(Y) = \bigcup_{i=1}^{n} GLA_{S'}(Y_i),$$

where

$$Y = \bigcup_{i=1}^{n} Y_i,$$

and Y_i are elementary sets in S.

If Y is an elementary set in S then $GLA_{S'}(Y)$ is also an elementary set in S' included in Y $(GLA_{S'}(Y) \subseteq Y)$ (possibly an empty set).

From the definition of elementary sets follows that there is exactly one GLA in S' for every describable set in S.

Now let us come back to our original problem.

We shall consider distributed information systems consisting of local systems $S_1, S_2, \dots, S_n$ as a connection of local systems, according to the definition given in Section 6.

The solution of this problem in general case is rather somewhat difficult, therefore we shall discuss in details two extreme cases only, corresponding to attribute and object connected systems.

Let $Y = \bigcup_{j=1}^{n} Y_j$ be a describable set in $S = \bigcup_{i=1}^{k} S_i$, where Y_j are elementary sets in S.

It can be shown that

(1) If $S = \bigcup_{i=1}^{k} S_i$ is an attribute connected system then

$$Y = \bigcap_{i=1}^{k} LUA_{S_i}(Y) = \bigcup_{j=1}^{n} \bigcap_{i=1}^{k} LUA_{S_i}(Y_j).$$

(2) If $S = \bigcup_{i=1}^{k} S_i$ is an object connected system then

$$Y = \bigcup_{i=1}^{k} GLA_{S_i}(Y) = \bigcup_{j=1}^{n} \bigcup_{i=1}^{k} GLA_{S_i}(Y_j).$$

Let us recall that describable sets in a system are only possible answers in this system and elementary sets one may consider as "atoms", which each answer consists of.

Thus if the distributed system consist of local systems with the same set of objects but different set of attributes every answer in the system can be represented univocally as an intersection of the least upper approximations of this answer in local systems.

Because each answer (describable set) is union of some elementary sets we may first represent elementary sets in a connected system as union of the least upper approximations in local system and then combine this "elementary" answers together by taking union of them.

Similarly in the case when the distributed system consist of local systems with the same set of attributes but different set of objects each answer in the connected system can be presented as union of greatest lower approximation of this answer in all local systems. We may also first "approximate" the "elementary" answers in local systems and then take union of all of them.

Now let us express properties (1) and (2) in terms of a query language.

Let $S = \bigcup_{i=1}^{k} S_i$ be an attribute connected system. Then for every term t in the language L_S we have

$$(3)\ \sigma_S(t) = \bigcap_{i=1}^{k} \sigma_{S_i}(t/A_i) = \bigcup_{j=1}^{n} \bigcap_{i=1}^{k} \sigma_{S_i}(t_j/A_i).$$

where normal form of t is $t_1 + t_2 + \cdots + t_n$, and t/A_i is to mean the term obtained from the term t by deleting in it all descriptor which do not belong to the set A_i. (If after this removal no descriptors remain, then we assume that $t/A_i = 1$). Thus t/A_i is the term of a query language of the local system S_i.

This property means simply that in order to find an answer to the query t in the language L_S, we must first translate it into a normal form $t_1 + t_2 + \cdots + t_n$. Then for any $1 \leq i \leq k$ and any $1 \leq j \leq n$, we remove from each elementary term all attributes which do not belong to the language L_{S_i}, i.e. for each $1 \leq i \leq n$ we replace all elementary terms by terms of the form $t_j/A_i = t'_j$ belonging to the language L_{S_i}. Afterwards we have to find the least upper approximation for each term t_j in every subsystem S_i. Then the intersection of all approximations corresponding to a fixed elementary term t_j constitutes the proper answer to this term.

In order to obtain the whole answer to the query t we have to "add" the answers of all elementary terms occuring in the normal form of t.

The situation may be depicted as shown in Fig. 1.

We have in this case two kinds of users: local and global (central) ones. Local users are attached to local systems, use local languages L_i, and have access only to informations in local systems. The global user can ask queries in the global language L and he has access to the informations in all local systems.

This kind of organization of distributed system has one serious disadvantage. In order to find the answer to a global query, one has to search for the best upper approximations of elementary terms in local system and take the intersection of all approximations. The intersection operation is very unefficient, because it requires access to many disc memories in order to retrieve the least upper approximations of elementary terms, but only small part of thus obtained data may occur in the intersection of all approximations.

Thus another solution of this of systems seems to be more efficient. This solution is depicted in Fig. 2. There is only one central system and each user uses its own language L_i which can be any sublanguage of L_S. Because, in this case there is only one partition S generated by the global system S the answer to any query in global language, or any sublanguage of the global language, may be obtained directly from the central memory as a sum of some elementary sets in S. Thus in this case there is no intersection operation, which considerably slows down the retrieval process.

In the second case when $S = \bigcup_{i=1}^{k} S_i$ is an object con-

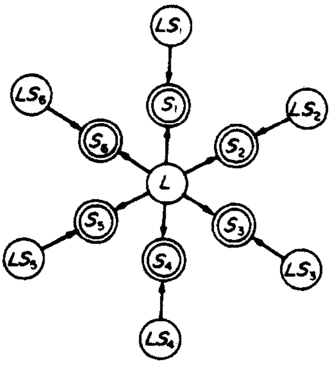


Fig. 1.

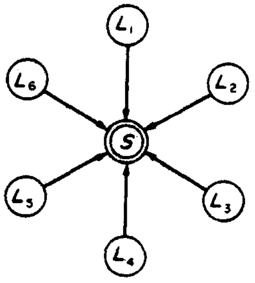


Fig. 2.

nected system, for every term t in the language L_S we have

$$(4)\ \sigma_S(t) = \bigcup_{i=1}^{k} \sigma_{S_i}(t) = \bigcup_{j=1}^{n} \bigcup_{i=1}^{k} \sigma_{S_i}(t_j),$$

where normal form of t is $t_1 + t_2 + \cdots + t_n$.

From this property it follows that in order to find the answer to the query t in the language L_S we have to find partial answers $\sigma_i(t)$ in all local systems S_i. This partial answers are unions of greatest lower approximations for t in all subsystems S_i of the system S. The answer to the query t is union of all partial answers. The operation is very simple to implement and is much faster than computing the intersection of upper approximations in the previous case.

We can also distinguish as in the previous case two kinds of memory organizations; local memory systems and control memory systems.

The implementation in both cases is similar and it is rather simple.

In both cases we may have central or local memory organization, that is to say that the information in the system may be physically centralized or distributed. From logical point of view however in both cases the system is distributed. So we could distinguish two kind of distributed systems: *language distributed* and *data distributed* systems. The first kind of distributions refers to some logical properties of the information systems (query processing philosophy), whereas the second kind—refers to physical distribution of data.

From our considerations follows also that in the attribute connected system every user may get some data about every object in the whole system, whereas in the object connected system every user may get every datȧ about same object. So in the first case we may easily restrict access to some data and in the second case we may easily restrict access to some object in the whole system.

Acknowledgements—Thanks are due to Dr. E. Konrad, Dr. W. Marek, Dr. E. Orłowska, Prof. H.-J. Schneider and the referees for their useful discussion and comments.

REFERENCES

[1] Z. Pawlak: Mathematical foundation of information retrieval. *CC PAS Reps*, No. 101 (1973).

[2] W. Marek and Z. Pawlak: Information storage and retrieval systems: mathematical foundations. *Theoretical Comput. Sci.* No. 1, 331–354 (1976).

[3] E. F. Codd: A relational model of data for large shared data banks. *Commun. ACM* **13**, 377–387 (1970).

[4] G. Salton: *Automatic Information Organization and Retrieval.* McGraw-Hill, New York (1968).

[5] E. Wong and T. C. Chiang: Canonical structure in attribute based file organization. *Commun. ACM* **14**, 593–597 (1971).

[6] V. Cherniavsky and H. J. Schneider: A data information language (a new approach to data base systems). *Rep. No.* 4/76.

[7] E. Margański: *Implementation of Retrieval System by the Method of Atomic Constituents* (in Polish), PWN (1979).

[8] K. M. Jaegerman: Information storage and retrieval systems with incomplete information, Part I. *Fundamenta Informaticae* **II.1**, 17–41 (1978); Part II, **II.2**, 141–166 (1979).

[9] W. Lipski: On semantic issues connected with incomplete information databases. *ACM Trans. Database Syst.* **4**(3), 262–298 (1979).

[10] M. Orłowska: *Algebraical and Topological Properties of Database Systems with Incomplete Information* (in Polish) PWN (1980).

[11] B. Konikowska and T. Traczyk: A query language of stochastic information systemş. *Fundamenta Informaticae* **II.3**, 351–363 (1978).

[12] E. Orɫowska: Dynamic information systems. *Fundamenta Informaticae* (in print).

[13] A. Wakulicz-Deja: Time varing information systems. Submitted to *Inform. Systems.*

[14] A. V. Aho and J. D. Ullman Beeri: The theory of joins in relational databases. *ACM Trans. Database Syst.* **4**(3), 297–314 (1979).

[15] E. Orłowska: On dependency of attributes in information systems. *JCS PAS Rep., No.* 425 (1980).

[16] M. Jaegermann and W. Marek: Dependencies of attributes in information systems. *JCS PAS Rep., No.* 428 (1981).

[17] J. Łoś: Characteristic sets of a system of equivalence relations. *Colloquium Mathematicum* **XLII**, 291–293 (1979).

[18] M. Truszczyński: Algorithmic aspects of the minimization of the set of attribute problem. *Fundamenta Informaticae* **4**(4) *Fundamenta Informaticae* (in print).

[19] J. Grzymała-Busse: manuscript, 1978.

[20] Z. Pawlak: Infomation systems. *ICS PAS Reps, No.* 338 (1978).

[21] W. Lipski and W. Marek: On queries involving cardinalities. *Inform. Syst.* **4**(3), 241–246 (1979).

[22] T. Traczyk: Common extension of Boolean information systems. *Fundamenta Informaticae* **II.1**, 63–70 (1978).