POLSKA AKADEMIA NAUK
INSTYTUT INFORMATYKI TEORETYCZNEJ I STOSOWANEJ

ARCHIWUM INFORMATYKI TEORETYCZNEJ I STOSOWANEJ

Tom 2 – Zeszyt 3-4

WARSZAWA 1990

WYDAWNICTWO NAUKOWE PWN

ARCHIWUM INFORMATYKI
TEORETYCZNEJ I STOSOWANEJ
Tom 2 (1990), z. 3-4
pp. 139-166

Information systems and decision tables a rough set perspective

ZDZISŁAW PAWLAK

Institute of Theoretical and Applied Informatics, Polish Academy of Sciences
ul. Baltycka 5, 44-100 Gliwice

(*Received 1989. 07. 15*)

Abstract. In this paper we are going to show how the concept of a rough set can be employed as a theoretical basis of information systems and decision tables. It turns out that many problems, in particular in AI, like machine learning, expert systems, pattern recognition decision support systems and others can be reduced to the proposed schemes. In fact the approach has found many real life applications in medicine [46, 47], cement kiln control algorithms [19], aircraft pilots performance evaluation [10] – and others.

I. Information systems

In this section we will give basic ideas concerning data tables, which will be called here information systems. These data tables contain some *explicit* facts about some objects, processes, phenomena etc., which represent our knowledge about some part of real or abstract world we are interested in. Our main task is to derive some *implicit* facts from the table. To this end the concept of a rough set will be used.

1. Introduction

Information system in our approach is in fact a data table, columns of which are labelled by attributes rows are labelled by objects (states, processes etc.) and each row represents an information about the corresponding object.

The data table can be obtained as a result of measurements, observations or represent knowledge of an agent or a group of agents. The origin of the data table is not important from our point of view and we shall be interested only in some the formal properties of such tables.

Informally speaking by an information system we mean a finite collection of data about some objects (states, processes etc.). We assume that objects are characterized by some features expressed as pairs (attribute, value). For example the following pairs (color, red), (height, tall), (sex, male), (age, young) are possible features of some objects.

Main problems we are going to deal with consist in discovering dependencies

among data, reducing data redundancies in the tables and generation decision rules. In other words we are interested in detecting attribute dependencies and reducing the set of attributes.

It turns out that many problems of fundamentally different nature can be reduced to the above mentioned data table analysis. Machine learning, pattern recognition, decision table or expert systems are exemplary applications problems where the proposed approach seems to give novel insight and algorithms.

2. Examples of information systems

In order to have better intuitive background, before we give formal definitions, let us first give with some examples of information systems.

Example 1. In this information system pathomorfological changes in cells organelles are listed (cf. [8]).

Table 3

State	Volume Density	Numerical Density	Surface Density
Normal	Normal	Normal	Normal
Proliferation	Normal	Increased	Increased
Hypertrophy	Increased	Normal	Normal
Hyperlasia	Increased	Increased	Increased
Hypoplasia	Decreased	Decreased	Decreased
Atrophy	Decreased	Normal	Decreased
Ageneration	Normal	Decreased	Decreased
Dysplasia	Increased	Decreased	Decreased
Dystrophy	Decreased	Increased	Increased

Objects in the system are states of cells of organelle systems. The organelles systems are characterized by attributes Volume Density, Numerical Density and Surface Density. ■

Example 2. Here an information about patients suffering from heart disease seen in one of the hospitals in Warsaw is given.

Table 4

	Gasom-etry	Dys-pnea	Cya-nosis	Pulmo-nary Stasis	Heart Rate	Hepato-megaly	Edema	Degree of Disease Advance
P1	37	1	1	1	62	0	0	1
P2	43	2	3	4	76	8	3	3
P3	42	1	2	1	71	1	0	1
P4	43	0	3	2	80	5	1	1
P5	48	1	3	3	92	6	3	3
P6	38	1	3	2	87	5	1	2
P7	54	0	0	0	95	1	0	2
P8	40	3	0	0	128	1	0	0
P9	40	1	0	0	111	1	0	1
P10	50	0	1	0	68	2	1	1

■

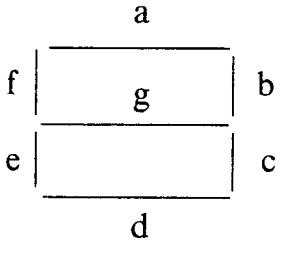
Example 3. In this example a characterization of various animals in terms of Size, Animality and Color is given [5].

Table 5

	Size	Animality	Colour
A1	small	bear	black
A2	medium	bear	black
A3	large	dog	brown
A4	small	cat	black
A5	medium	horse	black
A6	large	horse	black
A7	large	horse	brown

■

Example 4. The digit displays in calculators are composed out of seven elements as shown below.



Then structure of each digit is shown in the information system below.

Table 6

Digit	a	b	c	d	e	f	g
0	x	x	x	x	x	x	
1		x	x				
2	x	x		x	x		x
3	x	x	x	x			x
4		x	x			x	x
5	x		x	x		x	x
6	x		x	x	x	x	x
7	x	x	x				
8	x	x	x	x	x	x	x
9	x	x	x	x		x	x

Objects in the information system are digits 0, ..., 9 and attributes are elements a, b, c, d, e, f, g of the display. ■

Example 5. As is well known digital circuits can be described in a form of a truth tables. Each such a table can be considered as an information system. For example the following information system describes a binary full adder.

Table 7

	a_i	b_i	c_{i-1}	s_i	c_i
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

Objects in the system are states of the adder denoted 0, 1, 2, 3, 4, 5, 6 and 7, and attributes are the augments a_i, b_i, sum s_i, previous and current carry c_{i-1}, c_i. ■

It is easily seen from the examples that information system can represent the results of measurements, observations, or express knowledge of an agent (or group of agents).

In what follows we will ignore the semantic contents of the table, i.e. we will consider data table regardless what are the objects, attributes or their values, and we will treat information systems in entirely formal way.

Remark

At the end it is worthwhile to mention that the notion of an information system apparently looks like a relational table in the relational data base model [3].

There is, however, an essential difference between these two models. Most importantly, the relational model is not interested in the meaning of the information stored in the table. The emphasis is placed on efficient data structuring and manipulation. Consequently the objects about which information is contained in the table are not represented in the table. This is in contrarst with the primary assumption of the information system model presented here. In the information system all objects are explicitly represented and the attribute values i.e. the table entries have associated explicit meaning as features or properties of the objects. In addition to that the emphasis in the information system model is put mainly not on data structuring and manipulation but on analysis of actual dependencies existing in data, and data reduction, which is rather closer to statistical data model.

3. Formal definition

We begin this section with a formal definition of an information systems. An information system is a quadruple $S = (U, A, V, f)$ where

U – is a nonempty, finite set called the *universe*,

A – is a finite set of *attributes*

$V = \bigcup_{a \in A} V_a$; V_a – is called the *domain* of attribute a.

$f: U \times A \to V$ – is an *information function* such that

$f(x, a) \in V_a$ for every $a \in A$ and $x \in U$.

If f is a total function then S will be called *complete*; if f is a partial function, then S is referred to as *incomplete*. We shall be basically concerned with complete information systems unless stated otherwise.

In what follows we shall need the notion of information about an object – in the information system – which is defined below:

Let $x \in U$. The function $f_x: A \to V$ such that $f_x(a) = f(x, a)$ for every $a \in A$ will be called *information* on x in S.

Thus information on x is simply the set of values of an attributes assigned to an object x, or in other words – description of object x in terms of attributes available in the information system.

The example which follows will illustrate the definition.

Example 6. Let us consider the following information system.

Table 8

U	a	b	c	d	e
1	1	0	2	2	0
2	0	1	1	1	2
3	2	0	0	1	1
4	1	1	0	2	2
5	1	0	2	0	1
6	2	2	0	1	1
7	2	1	1	1	2
8	0	1	1	0	1

The universe U consist of 8 elements numbered 1, 2, 3, 4, 5, 6, 7 and 8, the set of attributes is $A = \{a, b, c, d, e\}$, whereas $V = V_a = V_b = V_c = V_d = V_e = \{0, 1, 2\}$. ■

4. Indiscernibility relation

It should be quite clear that some objects may have identical values of some attributes, i.e. they cannot be distinguished by attributes. This observation is fundamental one in our approach, and it is used to define the indiscernibility relation, which is the basis of rough set philosophy.

Let us express this more formally.

Let $S = (U, A, V, f)$ be an information system and let $P \subseteq A$. By $IND(P)$ we shall denote a binary relation over U defined as: $(x, y) \in IND(P)$ if and only if $f_x(a) = f_y(a)$ for every $a \in P$.

It is easily seen that $IND(P)$ is an equivalence relation for every P. Thus every subset of attributes generates the indiscernibility relation in the information system.

i.e. elements of U having the same values of attributes of P are indiscernible by the values of attributes P.

The family of an equivalence classes of the relation $IND(P)$ will be denoted by P^*, and elements of P^* are refereed to as blocks or indiscernibility classes of P^*. An equivalence class of the relation $IND(P)$ containing the element x is denoted by $[x]_P$.

An example of such indiscernibility relation, generated by the information system shown in Table 8 is given next.

Example 7. In the Table 8 elements 1, 4 and 5 of U are indiscernible by attribute a, elements 2, 7 and 8 are indiscernible by attributes b and c, and elements 2 and 7 are indiscernible by attributes d and e.

Exemplary partitions generated by attributes in this system are given below.

$a^* = \{\{2, 8\}, \{1, 4, 5\}, \{3, 6, 7\}\}$
$b^* = \{1, 3, 5\}, \{2, 4, 7, 8\}, \{6\}\}$
$\{c, d\}^* = \{\{1\}, \{2\}, \{3, 6\}, \{2, 7\}, \{4\}, \{5\}, \{8\}\}$
$\{a, b, c\}^* = \{\{1, 5\}, \{2, 8\}, \{3\}, \{4\}, \{6\}, \{7\}\}$ ■

The following are easy properties of indiscernibility relations:

Proposition 1.

(a) $IND(P) = \bigcap_{a \in P} IND(a)$, for every $P \subseteq A$

(b) $IND(P \cup Q) = IND(P) \cap IND(Q)$

(c) If $P \subseteq Q$ then $IND(Q) \subseteq IND(P)$

(d) $IND(\emptyset) = U \times U$

(e) $[x]_P = \bigcap_{a \in P} [x]_a$.

5. Approximations of sets

Having defined the indiscernibility relation, we are able to define now the basic concepts in our approach – the lower and the upper approximations.

Let us star first with some intuitive motivations. The main problem we are interested in is the description of some subsets of the universe by means of attributes available in the information system. It is clear that there are sets which can be exactly characterized by a given set of attributes, some sets however can not be defined in this way. Therefore we need the concept of approximate characterization of sets, and to this end we will the following definitions.

Let $P \subseteq A$ and $X \subseteq U$. The *P-lower approximation* of X, denoted $\underline{P}X$, and the *P-upper approximation* of X, denoted $\overline{P}X$ are, defined as below

$$\underline{P}X = U\{Y \in P^*: Y \subseteq X\}$$

$$\overline{P}X = U\{Y \in P^*: Y \cap X \neq \emptyset\}$$

The boundary of X is defined as $BN_P(X) = \overline{P}X - \underline{P}X$.

Set $\underline{P}X$ is the set of all elements of U which can with certainty classified as elements of X, employing the set of attributes P; $\overline{P}X$ is the set of elements of U which

can be possibly classified as elements of X, using the set of attributes P. The set $BN_P(X)$ is the set of elements which cannot be classified either to X or to $-X$ using the set of attributes P.

If $\overline{P}X = \underline{P}X$, set X will be called *exact* with respect to P; otherwise the set X is inexact or *rough* with respect to P.

Example 8. Consider the data Table 8, the set of attributes $C = \{a, b, c\}$ and the subset of the universe $X = \{1, 2, 3, 4, 5\}$. Then $\underline{C}X = \{1, 2, 3, 4, 5\}$, $\overline{C}X = \{1, 2, 3, 4, 5, 8\}$ and $BN_C(X) = \{8\}$.

Thus the set X is rough with respect to the attributes C, which is to say that we are unable to decide whether elements 2 and 8 are members of the set X or not. For the rest of the universe classification of elements, using the set C of attributes, is possible. ■

The following proposition exhibits basic properties of approximations:

Proposition 2.

(1) $\underline{P}X \subseteq X \subseteq \overline{P}X$

(2) $\underline{P}\emptyset = \overline{P}\emptyset = \emptyset$; $\underline{P}U = \overline{P}U = U$

(3) $\overline{P}(X \cup Y) = \overline{P}X \cup \overline{P}Y$

(4) $\underline{P}(X \cap Y) = \underline{P}X \cap \underline{P}Y$

(5) $X \subseteq Y$ implies $\underline{P}X \subseteq \underline{P}Y$ and $\overline{P}X \subseteq \overline{P}Y$

(6) $\underline{P}(X \cup Y) \supseteq \underline{P}X \cup \underline{P}Y$

(7) $\overline{P}(X \cap Y) \subseteq \overline{P}X \cap \overline{P}Y$

(8) $\underline{P}(-X) = -\overline{P}X$

(9) $\overline{P}(-X) = -\underline{P}X$

(10) $\underline{P}\,\underline{P}X = \overline{P}\,\underline{P}X = \underline{P}X$

(11) $\overline{P}\,\overline{P}X = \underline{P}\,\overline{P}X = \overline{P}X$

Proof.

(1a) If $x \in \underline{P}X$, then $[x] \subseteq X$, but $x \in [x]$ hence $x \in X$ and $\underline{P}X \subseteq X$.

(1b) If $x \in \overline{X}$, then $[x] \cap X \neq \emptyset$ (because $x \in [x] \cap X$) hence $x \in \underline{P}X$, and $X \subseteq \underline{P}X$.

(2a) From (1) $\underline{P}\emptyset \neq \emptyset$ and $\emptyset \subseteq \underline{P}\emptyset$ (because the empty set is included in every set) thus $\overline{P}\emptyset = \emptyset$.

(2b) Suppose $\overline{P}\emptyset \neq \emptyset$. Then there exist x such that $x \in \overline{P}\emptyset$. Hence $[x] \cap \emptyset \neq \emptyset$, but $[x] \cap \emptyset \neq \emptyset$, what contradicts the assumption, thus $\overline{P}\emptyset = \emptyset$.

(2c) From 1) $\underline{P}U \subseteq U$. In order to show that $U \subseteq \underline{P}U$ let us observe that if $x \in U$, then $[x] \subseteq U$, hence $x \in \underline{P}U$, thus $\underline{P}U = U$.

(2d) From (1) $\overline{P}U \supseteq U$, and obviously $\overline{P}U \subseteq U$, thus $\overline{P}U = U$.

(3) $x \in \overline{P}(X \cup Y)$ if $[x] \cap (X \cup Y) = \emptyset$ iff $[x] \cap X \cup [x] \cap Y \neq \emptyset$ iff $[x] \cap X \neq \neq \emptyset \vee [x] \cap Y \neq \emptyset$ if $x \in \overline{P}X \vee x \in \overline{P}Y$ iff $x \in \overline{P}X \cup \overline{P}Y$. Thus $\overline{P}(X \cup Y) = = \overline{P}X \cup \overline{P}Y$.

(4) $x \in \underline{P}(X \cap Y)$ if $[x] \subseteq X \cap Y$ if $[x] \subseteq X \wedge [x] \subseteq Y$ if $x \in \underline{P}X \cap \underline{P}Y$.

(5) Because $X \subseteq Y$ if $X \cap Y = X$ by virtue of (4) we have $\underline{P}(X \cap Y) = \underline{P}X$ if $\underline{P}X \cap \underline{P}Y = \underline{P}X$ which yields $\underline{P}X \subseteq \underline{P}Y$. Because $X \subseteq Y$ if $X \cup Y = Y$, hence $\overline{P}(X \cup Y) = \overline{P}Y$ and by virtue of (3) we have $\overline{P}X \cup \overline{P}Y = \overline{P}Y$ and hence $\overline{P}X \subseteq \overline{P}Y$.

(6) Because $X \subseteq X \cup Y$ and $Y \subseteq X \cup Y$, we have $\underline{P}X \subseteq \underline{P}(X \cup Y)$ and $\underline{P}Y \subseteq \subseteq \underline{P}(X \cup Y)$ which yields $\underline{P}X \cup \underline{P}Y \subseteq \underline{P}(X \cup Y)$.

(7) Because $X \cap Y \subseteq X$ and $X \cap Y \subseteq Y$, we have $\overline{P}(X \cap Y) \subseteq \overline{P}X$ and $\overline{P}(X \cap Y) \subseteq \overline{P}Y$ hence $\overline{P}(X \cap Y) \subseteq \overline{P}X \subseteq \overline{P}Y$.

(8) $x \in \underline{P}(x)$ if $[x] \subseteq X$ if $[x] \cap -X = \emptyset$ if $x \notin \overline{P}(-X)$ if $x \in -\overline{P}(-X)$, hence $\underline{P}(X) = -\overline{P}(-X)$.

(9) From substitution $-X$ for X in (9) we get $\overline{P}(X) = -\underline{P}(-X)$.

(10a) From (1) $\underline{P}\underline{P}X \subseteq \underline{P}X$, thus we have, to show that $\underline{P}X \subseteq \underline{P}\underline{P}X$. If $x \in \underline{P}X$ then $[x] \subseteq X$, hence $\underline{P}[x] \subseteq \underline{P}X$ but $\underline{P}[x] = [x]$, thus $[x] \subseteq \underline{P}X$ and $x \in \underline{P}\underline{P}X$, that is $\underline{P}X \subseteq \underline{P}\underline{P}X$.

(10b) From (1) $\underline{P}X \subseteq \overline{P}\underline{P}X$, thus it is enough to show that $\underline{P}X \supseteq \overline{P}\underline{P}X$. If $x \in \overline{P}\underline{P}X$, then $[x] \cap \underline{P}X \neq \emptyset$, i.e. there exists $y \in [x]$ such that $y \in \underline{P}X$, hence $[y] \subseteq X$ but $[x] = [y]$, thus $[x] \subseteq X$ and $x \in \underline{P}X$ which is to mean that $\underline{P}X \supseteq \overline{P}\underline{P}X$.

(11a) From (1) $\overline{P}X \subseteq \overline{P}\,\overline{P}X$. We have to show, that $\overline{P}X \supseteq \overline{P}\,\overline{P}X$. If $x \in \overline{P}\,\overline{P}X$, then $[x] \cap \overline{P}X \neq \emptyset$ and for some $y \in [x]$ $y \in \overline{P}X$, hence $[y] \cap X \neq \emptyset$ but $[x] = [y]$, thus $[x] \cap X \neq \emptyset$, i.e. $x \in \overline{P}X$, which yields $\overline{P}X \supseteq \overline{P}\,\overline{P}X$.

(11b) From (1) $\underline{P}\overline{P}X \subseteq \overline{P}X$. We have to show, that $\underline{P}\overline{P}X \supseteq \overline{P}X$. If $x \in \overline{P}X$ then $[x] \cap X \neq \emptyset$. Hence $[x] \subseteq \overline{P}X$ (because if $y \in [x]$, then $[y] \cap X = [x] \cap X \neq \neq \emptyset$, i.e. $y \in \overline{P}X$) and $x \in \underline{P}\overline{P}X$, which gives $\underline{P}\overline{P}X \supseteq X$. ■

The properties (6) and (7) show that approximations can not be always computed step-by-step, which is a serious drawback of the presented approach. Nevertheless these properties convey important message, that dividing data table into smaller parts leads in general to loss of information.

Also properties (8) and (9) indicate that the complement of approximation is not a "classical" operation.

II. Dependency of attributes

The dependency of attributes is another fundamental concept in the presented approach.

Intuitively speaking subset of attributes Q depends on subset of attributes A, if

values of attributes in A are uniquely determined by values of attributes in P, i.e. if there exists a function which assigns to each set of values of P set values of Q.

The idea of the dependency will be used to derive *implicit* facts from a data table, concerning dependencies among data.

1. Formal definition and some properties

Formally the dependency can be defined as shown below:

Let $S = (U, A, V, f)$ be an information system and let $P, Q \subseteq A$.

(1) The set of attributes Q *depends* on the set of attributes P in S denoted as $P \rightarrow_S Q$ (or in short $P \rightarrow Q$) if $IND(P) \subseteq IND(Q)$.

(2) Sets of attributes $P, Q \subseteq A$ are *equivalent*, denoted as $P \approx Q$, if $P \rightarrow Q$ and $Q \rightarrow P$.

(3) Sets of attributes $P, Q \subseteq A$ are *independent*, denoted as $P \approx Q$, if neither $P \rightarrow Q$ nor $Q \rightarrow P$.

If $P \rightarrow Q$ we will refer to P as *condition attributes* and to Q as *decision (or action) attributes.*

Obviously $P \approx Q$ if and only if $IND(P) = IND(Q)$.

The following example will demonstrate the definition of dependency.

Example 9. In the Table 8 we have the following dependency $\{a, b\} \rightarrow \{c\}$, because the indiscernibility relation $\{a, b\}$ has the following blocks $\{1, 5\}$, $\{2, 8\}$, $\{3\}$, $\{4\}$, $\{6\}$, $\{7\}$ and the indiscernibility relation $\{c\}$ has the blocks $\{1, 5\}$, $\{2, 7, 8\}$ and $\{3, 4, 6\}$, hence $IND(\{a, b\}) \subset IND(\{c\})$. ■

It is easy to show by simple computation the following properties.

Proposition 3. The following conditions are equivalent:

(1) $P \rightarrow Q$

(2) $IND(P \cup Q) = IND(P)$

(3) $POS_P(Q^*) = U$, where $POS_P(Q^*) = \bigcup_{X \in Q^*} \underline{P}X$

(4) $\underline{P}X = X$ for all $X \in Q^*$. ■

The Proposition 3 demonstrates that if the set Q depends on P, then the set Q is superfluous in the system in the sense that sets $P \cup Q$ and P provide the same characterization of objects. Thus this property can be used to reduce the set of attributes, which will be discussed in next section.

The properties given below characterize the dependency of attributes in more detail.

Proposition 4.

(1) If $P \rightarrow Q$, and $P' \supset P$, then $P' \rightarrow Q$

(2) If $P \rightarrow Q$, and $Q' \subset Q$, then $P \rightarrow Q'$

(3) $P \rightarrow Q$ and $Q \rightarrow R$ imply $P \rightarrow R$

(4) $P \rightarrow R$ and $Q \rightarrow R$ imply $P \cup Q \rightarrow R$

(5) $P \rightarrow R \cup Q$ imply $P \rightarrow R$ and $P \rightarrow Q$

(6) $P \to Q$ and $Q \cup R \to T$ imply $P \cup R \to T$
(7) $P \to Q$ and $R \to T$ imply $P \cup R \to Q \cup T$. ■

As we mentioned before the existence of dependency of attributes is based on the fact that values of the condition attributes determine values of the decision attribute. This dependency is functional in nature and will be referred to as dependency function. To further clarify this notion we will take a closer look at some properties of this function.

Let us first start with the simple case of dependency $\{p\} \to \{q\}$, or in short $p \to q$, where $p, q \in A$.

If $p \to q$ then there exists a unique dependency function

$$h\colon V_p \to V_q$$

such that

$$f(x, q) = h(f(x, p)).$$

To establish the relationship between partitions and dependency function let $X_{p,v} = \{x \in U : f(x, p) = v\}$. Of course $X_{p,v} \in P^*$, i.e. this is an equivalence class of the relation $IND(P)$.

Proposition 5. The following conditions are equivalent:
(1) $p \to q$ holds
(2) $X_{p,v} \subseteq X_{(q, h(v))}$ for all $v \in V_p$. ■
The above property is illustrated in the next examples.

Example 10. In the information system below

Table 9

U	a	b	c	d
1	0	0	0	0
2	0	1	0	2
3	1	1	0	1
4	1	1	0	1
5	0	1	1	2

there are following partitions:

$$a^* = \{\{1, 2, 5\}, \{3, 4\}\}$$
$$b^* = \{\{1\}, \{2, 3, 4, 5\}\}$$
$$c^* = \{\{1, 2, 3, 4\}, \{5\}\}$$
$$d^* = \{\{1\}, \{2, 5\}, \{3, 4\}\}$$

and hence the following dependencies are valid:

$$d \to a$$
$$d \to b.$$

The corresponding dependency functions are:

h_1	
$d \to a$	
0	0
2	0
1	1
2	0

h_2	
$d \to b$	
0	0
2	1
1	1

■

Example 11. In the example 1 we have may notice dependencies:
{Volume Density, Numerical Density} → Surface Density
{Volume Density, Surface Density} → Numerical Density.

■

In general case if the dependency $P \to Q$ holds then there exist functions h_q for $q \in Q$, such that

$$f(x, q) = h_q(f(x, p_1), \ldots, f(x, p_n))$$

for each $q \in Q$ where $P = \{p_1, \ldots, p_n\}$.

Proposition 6. The following conditions are equivalent:

(1) $P \to Q$

(2) $\bigcap_{p \in P} X_{p,v} \subseteq X_{(q, h_q(v))}$, for all $v \in V_q$ and $q \in Q$. ■

Let us also note that the dependency of attributes can be associated another very important interpretation. Under this interpretation whenever $P \to Q$ holds we can immediately conclude that any object can be correctly classified into a unique class of the partition Q^* based solely on the information expressed through values of attributes belonging to P. In other words values of attributes from P uniquely characterize classes of the partition Q^*. In fact this is the consequence of functional dependencies between Q and P.

2. Partial dependency of attributes

The above introduced notion of dependency of attributes is to strong for some real life applications and therefore in what follows we will define somewhat weaker notion of dependency of attributes.

Let $S = (U, A, V, f)$ be an information system and $P, Q \subseteq A$.

We say that the set of attributes Q *depends* in a degree k $(0 \leqslant k \leqslant 1)$ on the set of attributes P (in S), symbolically $P \to^k Q$, if

$$k = \gamma_P(Q^*) = \frac{card\, POS_P(Q^*)}{card\,(U)}$$

where card denotes cardinality of the set.

If $k = 1$ we will say that Q *depends totally* on P (or in short depends); if $0 < k < 1$, we say that Q *depends roughly* (*partially*) on P, and if $k = 0$ we say that Q is *totally independent* of P. If $P \to {}^1Q$ we shall also write $P \to Q$.

Let us notice that the dependency $P \to {}^kQ$, for $k = 1$ coincides with the notion of attribute dependency introduced at the beginning of the section. That is, if $k = 1$ then we have total functional dependency among corresponding attributes. If $0 < k < 1$ then the functional dependency is confined to some, but not all objects in the table. That kind of dependency can be also referred to as partial functional dependency. Finally, if $k = 0$ none of the values of the attributes in P are sufficient to determine corresponding values of attributes in Q. In this case there is entirely no functional dependency between P and Q. The idea of partial functional dependency can be also interpreted in terms of our ability to classify objects. More precisely, from the definition of dependency follows that if $P \to {}^kQ$ then the positive region of the partition Q^* induced by Q covers k^*100 percent of all objects represented in the table. On the other hand only those objects belonging to positive region of the partition can be uniquely classified. This means that k^*100 percent of objects can be classified into blocks of partition Q^* based on values of attributes belonging to P.

Thus the coefficient $\gamma_P(Q^*)$ can be understood as a degree of dependency between Q and P. In other words if we restrict the set of objects in the information system $S = (U, A, V, f)$ to the set POS_PQ^*, we would obtain the system $S' = (POS_PQ^*, A, V, f)$ in which $P \to Q$ is a total functional dependency.

Of course one could use another measure of rough dependency but the one assumed here seems to be very well suited to various applications and it is also easy to compute and interpret.

The measure k of dependency $P \to {}^kQ$ does not capture how actually this partial dependency is distributed among decision classes. For example some decision classes can be fully characterized by attributes in P whereas others may be characterized only partially. To this end we will need also a coefficient $\gamma_P(X) = card\, \underline{P}X/card\, X$, where $X \in Q^*$, which says how elements of each class of Q^* can be classified by employing only the set of attributes P.

Thus the two numbers $\gamma_P(Q^*)$ and $\gamma_P(X)$, $X \in Q^*$ give us full information about the "classification power" of the set at attributes P with respect to the classification Q^*.

Example 12. Let us compute the degree of dependency of attributes $D = \{d, e\}$ from the attributes $C = \{a, b, c\}$ in the Table 8. The partition D^* consists of the following blocks, $X_1 = \{1\}$, $X_2 = \{2, 7\}$, $X_3 = \{3, 6\}$, $X_4 = \{4\}$, $X_5 = \{5, 8\}$ and the partition C^* consists of blocks $Y_1 = \{1, 5\}$, $Y_2 = \{2, 8\}$, $Y_3 = \{3\}$, $Y_4 = \{4\}$, $Y_5 = \{6\}$ and $Y_6 = \{7\}$.

Because $\underline{C}X_1 = \emptyset$, $\underline{C}X_2 = Y_6$, $\underline{C}X_3 = Y_3 \cup Y_5$, $\underline{C}X_4 = Y_4$ and $\underline{C}X_5 = \emptyset$, thus $POS_C(D^*) = Y_3 \cup Y_4 \cup Y_5 \cup Y_6 = \{3, 4, 6, 7\}$. That is to say that only these elements can be classified into blocks of the partition D^* employing the set $C = \{a, b, c\}$

attributes. Hence the degree of dependency between Q and P is $\gamma_C(D^*) = 4/8 = 0.5$. ■

The following properties are counterparts of Proposition 4.

Proposition 7.

(1) If $R \to^k P$ and $Q \to^1 P$, then $R \cup Q \to^m P$, for some $m \geqslant \max(k, 1)$.
(2) If $R \cup P \to^k Q$, then $R \to^1 Q$ and $P \to^m Q$, for some 1, $m \leqslant k$.
(3) If $R \to^k Q$ and $R \to^1 P$, then $R \to^m Q \cup P$, for some $m \leqslant \min(k, 1)$.
(4) If $R \to^k Q \cup P$, then $R \to^1 Q$ and $R \to^m P$, for some 1, $m \geqslant k$.
(5) If $R \to^k P$ and $P \to^1 Q$, then $R \to^m Q$, for some $m \geqslant 1 + k - 1$. ■

III. Reduction of attributes

Our next most important notion is that of reduction of attributes.

As we have seen in the previous section some attributes can be abandoned in the information system without loose of information about objects. In this section we shall discuss this problem in some detail.

1. Formal definition and some properties

Let $S = (U, A, V, f)$ be an information system and let $P \subseteq A$.

We say that subset of attributes P is *independent* (in S), if for every $Q \subset P$ $IND(P) \neq IND(Q)$, otherwise subset P is *dependent* (in S).

Before we prove some properties of independence let us first observe that from Proposition 1 immediately follows the following fact:

LEMMA 1. Let P, Q, R be sets of attributes such that $P \subseteq Q \subseteq R$. If $IND(P) = IND(R)$, then $IND(P) = IND(Q) = IND(R)$. ■

Proposition 8. If P is independent and $Q \subseteq P$, then Q is also independent.

Proof. The proof is by contradiction. Suppose $Q \subset P$ and Q is dependent. Then there exists $R \subset Q$ such that $IND(R) = IND(Q)$, which implies $IND(R \cup (P-Q)) = IND(P)$ and $R \cup (P-Q) \subset P$. Hence P is dependent which is a contradiction. ■

Proposition 9. The following conditions are equivalent:

(1) P is independent
(2) $IND(P) \neq IND(P - \{a\})$, for every $a \in P$.

Proof. Obviously, (1) implies (2). The converse implication we shall prove by contradiction. Suppose P is dependent. Then there exists $Q \subset P$ such that $IND(P) = IND(Q)$. Then by Lemma 1 for any $p \in P-Q$, $Q \subseteq P - \{p\} \subseteq P$, $IND(P) = IND(P - \{p\})$, which is contradiction. ■

Based on the simple fact expressed in the Proposition 9 we can easily verify weather a subset of attributes is dependent or independent. This property is

important from computational point of view as it permits to replace the operation of of checking all subsets of P by simple removal of single attributes.

In many practical applications we are interested in reducing those attributes which are redundant with respect to a classification generated by the whole set of decision attributes. The classification of objects obtaining in the absence of such attributes is a good, in the sense of preserving the original classification, as the classification based on all attributes. To this end we introduce the concept of a reduct, that is, of a subset of attributes which is characterized by the following conditions:

1. It preservers the original classification,
2. None of the attributes can be removed from the reduct without destroying the property 1.

More precise definition of reduct is given below:

A subset $P \subseteq Q \subseteq A$ is a *reduct* of Q (in S), if P is independent subset of Q, and $IND(P) = IND(Q)$.

Example 13. The set C of attributes in the Table 8 has only one reduct $\{a, b\}$. A subset of attributes may have more then one reduct. The set of attributes $\{a, b, c\}$ in the Table 10.

Table 10

U	a	b	c	d
1	1	0	2	2
2	0	1	1	0
3	2	0	0	0
4	1	1	0	1
5	1	0	2	2
6	2	0	0	1
7	0	1	1	2
8	1	1	1	0
9	1	0	2	2
10	0	1	1	0

has three reducts $\{a, b\}$, $\{b, c\}$ and $\{a, c\}$ and consequently

$$\{a, b\} \to \{c, d\},\ \{b, c\} \to \{a, d\} \text{ and } \{a, c\} \to \{b, d\}. \quad \blacksquare$$

Example 14. In the information system given in Example 1 we have two reducts

{Volume Density, Numerical Density}
{Volume Density, Surface Density}.

That means that in order to characterize pathological states of a cell it is not necessary to use all of the three attributes, but it suffices to use only the attributes Volume Density and Numerical Density or Volume Density and Surface Density. ■

There are several interesting and important from practical perspective properties of the reduct. We summarize them in the following two propositions:

Proposition 10. If P is a reduct of Q, then $P \to Q-P$ and $IND(P) = IND(Q)$. ■

Proposition 11.

(a) If P is dependent, then there exists a subset $Q \subseteq P (Q \neq \emptyset)$ such that Q is a reduct of P.

(b) If $P \subseteq A$ is an independent subset of attributes, then all attributes in P are pair wise independent.

(c) *If* $P \subseteq A$ is an independent subset of attributes then every subset Q of P is independent.

(d) If $P \to^k Q$ and R is a reduct of P, then $R \to^k Q$. ■

2. Relative reduct of attributes

The definition of the reduct introduced in this section is based on the idea of preservation of classification. Briefly, we can say that the approximation space induced by the reduced set of attributes is the same as the space generated by the attributes before reduction. As we see from Proposition 11 d) the reduced set of attributes also preserves the dependency. That is, if a subset of decision attributes was determined in a degree k by a certain subset of condition attributes then the reduct of condition attributes also determines decision attributes with the same degree. It does not mean however that every subset of condition attributes which preserves the original dependency is a reduct. The question, often imposed in practical data reduction is how far we can reduce the original set of condition attributes without affecting dependency with decision attributes. Our investigation into that problem has led us to the generalized definition of the reduct, so-called relative reduct, which is based on the idea of dependency preserving between two groups of attributes, rather that of classification. Loosely speaking, assuming that we defined two groups of condition and decision attributes, the relative reduct is a subset of condition attributes such that:

1. It preserves the dependency with decision attributes.
2. None of the attributes can be removed from the reduct without affecting the dependency.

In what follows we introduce the definitions in more systematic and precise terms.

Let $S = (U, A, V, f)$ be an information system and let $P, Q \subseteq A$. We say that P *is independent with respect to* Q (or in short *Q-independent*), if for every $R \subset P\, POS_P(Q^*) \neq POS_R(Q^*)$; otherwise P is *dependent with respect* to Q (or in short *Q-dependent*).

Proposition 12. The following conditions are equivalent:

(1) P is independent with respect to Q,

(2) $POS_P(Q^*) \neq POS_{P-\{a\}}(Q^*)$ for every $a \in P$. ■

Let us notice that in particular if $P = Q$, we obtain the previously introduced concept of dependency of a set of attributes.

Set $R \subseteq P$ will be called a *relative reduct of P with respect to Q*, or in short Q-reduct of P, if R is independent subset of P with respect to Q, and $POS_R(Q^*) = POS_P(Q^*)$.

Again, if $P = Q$ the relative reduct of P with respect to Q, coincides with the reduct of P.

Directly from the definition we get the following properties:

Proposition 13.

(1) If $P \rightarrow^k Q$ and R is a relative reduct of P with respect to Q, then $R \rightarrow^k Q$.

(2) If P and Q are a reduct and a reduct with respect to R respectively, then $P \supseteq Q$. ■

In the next example we provide simple illustration of the notion of the relative reduct.

Example 15. Let us consider the Table 11, shown below where $B = \{a, b, c, d\}$ and $C = \{e, f\}$.

Table 11

U	a	b	c	d	e	f
1	1	0	0	1	1	2
2	1	0	0	0	1	2
3	1	1	0	1	1	2
4	0	0	0	0	0	2
5	0	0	0	1	0	2
6	1	0	0	2	1	1
7	1	1	0	2	1	1
8	2	1	0	2	1	1
9	2	1	1	2	1	0
10	2	2	1	2	1	0
11	2	2	1	1	1	0
12	2	1	1	1	1	0
13	2	0	1	1	1	0

It is easy to verify that the set B of attributes is independent. This set of attributes is also dependent with respect to the set C of attributes. The only C-reduct of attributes B is the set $\{a, c, d\}$. ■

We end this section with the following important remark.

The idea of the relative reduct can be modified to provide for further reduction of attributes. Let

$$Q^* = \{X_1, \ldots, X_n\}$$

be the partition (classification) generated by the set of attributes Q. We can apply the concept of a relative reduct of attributes to distinguish the class X_i from the remaining classes $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n$, thus obtaining the set of attributes characteristic to each class X_i. This procedure leads to n reducts (possibly different)

R_i each of which is associated with the class X_i. Of course in general case each class can have more then one reduct. This kind of reduct will be called a binary reduct. Binary reduct are particularly useful in decision rules generation and deep simplification of decision tables [52].

A detail discussion of this problem is left to the reader.

3. Core of attributes

In this section we are going to define a very useful concept in our approach, the notion of a core of attributes.

Intuitively speaking the core is the subset of the most important or significant attributes in the information system from the classification point of view.

The formal definition of the core is given below:

Let $S = (U, A, V, f)$ be a decision table, $P \subseteq A$ and $a \in P$.

We say that an attribute $a \in P$ is *superfluous* in P if $IND(P - \{a\}) = IND(P)$; otherwise the attribute a is *indispensible* in P.

Obviously an attribute $a \in P$ is superfluous in P if and only if

$$P - \{a\} \rightarrow a.$$

The set of all indispensable attributes of P will be called the *core* of P, i.e. the core of P is the set

$$CORE(P) = \{a \in P: IND(P - \{a\}) \neq IND(P)\}.$$

Example 16. The set of attributes $B = \{a, b, c\}$ in the Table 10 has the empty core. In the Table 12

Table 12

U	a	b	c	d
1	1	0	2	1
2	0	1	1	2
3	2	0	0	0
4	1	1	0	1
5	1	0	2	0
6	2	0	0	2
7	2	1	1	1
8	0	1	1	1

the core of the set of attributes $\{a, b, c\}$ is the attribute a. ■

The following property explains the relationship between the core and reducts of attributes.

Proposition 14.

$$CORE(P) = \bigcap_{Q \in RED(P)} Q$$

where $RED(P)$ is the family of all reducts of P.

Proof. If Q is a reduct of P and $p \in P-Q$, then $IND(P) = IND(Q)$, $Q \subseteq P-\{a\} \subseteq P$. Hence by Lemma 1 we get $IND(P) = IND(P-\{p\})$. Thus p is superfluous, i.e. $p \notin CORE(P)$, and $CORE(P) \subseteq \bigcap \{Q: Q \in RED(P)\}$.

Suppose $p \notin CORE(P)$, i.e. p is superfluous in P. That means $IND(P) = IND(P-\{p\})$, which implies that there exists independent subset $R \subset P-\{p\}$, such that $IND(R) = IND(P)$. Obviously R is a reduct of P and $p \notin R$. This shows $CORE(P) \supseteq \bigcap \{Q: Q \in RED(P)\}$. ■

Example 17. The set $B = \{a, b, c\}$ of attributes in the decision Table 8 has two reducts $\{a, b\}$ and $\{a, c\}$ and consequently the core $\{a, b\} \cap \{a, c\} = \{a\}$. ■

The use of the concept of core is twofold. First, it can be used as a basis for computation of all reducts, for the core is included in every reduct, and its computation is straightforward. Secondly, the core can be interpreted as the set of most important attributes in the case when there are many reducts.

4. Relative core of attributes

The concept of core can be generalized in a way similar as that of a reduct.

The generalized core, called relative core, as the relative reduct, is based on the concept of partial dependency between two groups of condition and decision attributes. In this case the core of attributes consists of most important or significant attributes with respect to determining values of decision attributes by values of condition ones. In what follows we present more systematic and precise definition to the relative case.

Let $S = (U, A, V, f)$ be an information system and $P, Q \subseteq A$.

The attribute $a \in P$ is *superfluous* in P *with respect* to Q (*Q-superfluous*), if

$$POS_P(Q^*) = POS_{P-\{a\}}(Q^*);$$

otherwise a is *indispensable in* P *with respect to* Q (Q-indispensable).

The relative core of P with respect to Q (Q-core of P), denoted $CORE_Q(P)$, is the set of all indispensable attributes of P with respect to Q, i.e.

$$CORE_Q(P) = \{a \in P: POS_P(Q^*) \neq POS_{P-\{a\}}(Q^*)\}.$$

For relative core we have the following property.

Proposition 15.

$$CORE_Q(P) = \bigcap_{R \in RED_Q(P)} R.$$ ■

The proof of this theorem is similar to the previous one.

It is easy to see that if $P = Q$, then the relative core coincide with the core, which is defined in the previous section.

IV. Decision tables

In this section we will consider special, important class of information systems, called *decision tables*. Basics of the decision tables can be found in Hurley 1983 [6].

Decision table is a finite set of decision rules, which specify what decisions (actions) should be undertaken when some conditions are satisfied.

It turns out that the information system provides a very good framework as a basis of decision tables theory.

1. Formal definition and some properties

Decision tables can be defined in terms of information system as follows.

Let $S = (U, A, V, f)$ be an information system and $C, D \subset A$ two subsets of attributes such that $C \cap D = \emptyset$ and $C \cup D = A$, called *condition* and *decision attributes* respectively. Information system S with distinguished condition and decision attributes will be called a *decision table*, and will be denoted $S = (U, C, D, V, f)$.

Equivalence classes of the relations $IND(C)$ and $IND(D)$ will be called *condition* and *decision classes*, respectively.

The function $f_x: A \to V$, such that $f_x(a) = f(x, a)$, for every $a \in A$, $x \in U$ will be called a decision rule (in S).

If g is a decision rule, then the restriction of g to C, denoted $g|C$, and the restriction of g to D, denoted $g|D$ will be called *conditions* and *decisions* (*actions*) of g respectively.

The decision rule is *deterministic* (in S) if for every $y \neq x$, $f_x|C = f_y|C$ implies $f_x|D = f_y|D$; otherwise f_x is *nondeterministic*.

A decision table is *deterministic* (*consistent*) if all its decision rules are deterministic; otherwise a decision table is *nondeterministic* (*inconsistent*).

The following is the important property that establishes relationship between determinism (consistency) and dependency of attributes in a decision table.

Proposition 16. A decision table $S = (U, C, D, V, f)$ is deterministic (consistent) if and only if $C \to D$. ■

From the Proposition 16 it follows practical method of checking consistency of decision table by simply computing the degree of dependency between condition and decision attributes. If the degree of dependency equals to 1 then we conclude that the table is consistent; otherwise it is inconsistent.

The next property is also important form practical perspective.

Proposition 17. Each decision table $S = (U, C, D, V, f)$ can be uniquely decomposed into two decision tables $S_1 = (U_1, C, D, V_1, f_1)$ and $S_2 = (U_2, C, D, V_2, f_2)$ such that $C \xrightarrow{1} D$ in S_1 and $C \xrightarrow{0} D$ in S_2, where $U_1 = POS_C(D^*)$

f_1 is the restriction of f to U_1

$U_2 = \bigcup_{X \in D^*} BN_C(X)$

f_2 is the restriction of f to U_2
and V_1, V_2 are ranges of functions f_1 and f_2, respectively. ■

Similarly to Proposition 16, the Proposition 17 also leads to an interesting practical method of processing decision tables. Suppose that we used computation of dependency between condition and decision attributes according to Proposition 16 to verify the consistency of a decision table. If the table turned out to be inconsistent i.e. the dependency degree was less than 1, then based on Proposition 17, we could decompose the table into two sub tables: one totally inconsistent with dependency coefficient equal to zero and the second entirely consistent with the dependency equal to one. This decomposition however is possible only if the degree of dependency is greater than zero and different form 1.

Example 18. Let us consider the decision table given in Table 8. We will assume that a, b and c are condition attributes, and d and f are decision attributes. In this table for instance the decision rule 1 is nondeterministic, whereas the decision rule 3 is deterministic. By employing the Proposition 16 we can decompose the decision table 8 into the following two decision tables:

Table 13

U_1	a	b	c	d	e
3	2	0	0	1	1
4	1	1	0	2	2
6	2	2	0	1	1
7	2	1	1	1	2

Table 14

U_2	a	b	c	d	e
1	1	0	2	2	0
2	0	1	1	1	2
5	1	0	2	0	1
8	0	1	1	0	1

The decision Table 12 is deterministic whereas the decision Table 13 is totally nondeterministic, which means that all decision rules in Table 12 are deterministic, and in the decision Table 13 – all decision rules are nondeterministic. ■

2. Simplification of decision tables

Simplification of decision tables is of primary importance in many applications. Example of simplification is the reduction of condition attributes in a decision table. In the reduced decision table the same decisions can be based on smaller number of conditions. This kind of simplification eliminates the need for checking unnecessary conditions or, in some applications, performing expensive test to arrive at a con-

clusion which eventually could be achieved by simpler means. Simplification of decision tables has been investigated by many authors [6], and there is a variety of informal approaches, to this problem.

Let us also mention that the simplification of boolean functions in the context of digital circuits design (cf. Muroga 1973) may be also viewed as a simplification of decision tables.

The approach to table simplification presented here consists of three steps:

(1) Computation of reducts of condition attributes, which is equivalent to elimination of some column from the decision table.

(2) Elimination of duplicate rows.

(3) Elimination of superfluous values of attributes.

Remark

We should note that in contrast to the general notion of an information system rows do not represent here any real objects. Consequently duplicate rows can be eliminated as they correspond to the same decision.

Thus the proposed method consists in removing superfluous condition attributes (columns), duplicate rows and in addition to that irrelevant values of condition attributes.

In this way we obtain "incomplete" decision table, containing only those values of condition attributes which are necessary to make decisions.

From mathematical point of view, removing attributes and removing values of attributes are alike and will be explained in what follows.

For the sake of simplicity we assume that the set of condition attributes is already reduced, i.e. there are not superfluous condition attributes in the decision table.

As we have already mentioned with every subset of attributes P we can associate the partition P^*, and consequently set of condition and decision attributes define partitions of objects into condition and decision classes.

Because we want to discern every decision class using minimal number of conditions — our problem can be reduce now to searching for relative reducts of condition classes with respect to decision classes. To this end we can use similar methods to that of finding reducts of attributes.

Now we define all necessary notions needed in this section.

Let $F = \{X_1, \ldots, X_n\}$ be a family of sets and let $\bigcap F = \bigcap_{i=1}^{n} X_i$. We say that X_i is *superfluous* in F, if $\bigcap F = \bigcap (F - \{X_i\})$, otherwise set X_i is *indispensable* in F.

The family of all indispensable sets in F will be called a *core* of F.

A family of $G \subseteq F$ is *independent* if all of its components are indispensable. A subset $H \subseteq F$ will be called a *reduct* of F if $\bigcap H = \bigcap F$ and H is independent.

As we see the introduced notions are counterparts to that introduced to that introduced in Section 2.6 with the only difference, that instead of relations we deal now with sets.

By analogy to the previous section we can also introduce the notions of relative reduct and relative core.

Suppose we are given a family $F = \{X_1, \ldots, X_n\}$, $X_i \subseteq U$ and a subset $Y \subseteq U$, such that $\bigcap F \subseteq Y$.

We say that X_i is *Y-superfluous* in F, if $\bigcap(F - \{X_i\}) \subseteq Y$, otherwise the set X_i is *Y-indispensable* in F.

A family $H \subseteq F$ is a *Y-relative reduct* of F, if H is Y-indispensable and $\bigcap H \subseteq Y$.

As we can see the introduced definitions again differ from the one discussed previously only in this regard that instead of relations we deal now with sets.

Let us also notice that the counterparts of Propositions 2.13 and 2.14 is also valid in the present framework as shown below.

Proposition 18.

$$\text{(a)} \quad \bigcap_{H \in RED(F)} H = CORE(F)$$

$$\text{(b)} \quad \bigcap_{H \in RED_Y(F)} = CORE_Y(F)$$

where $RED(F)$ $(RED_Y(F))$ is the family of all reducts (relative *Y-reducts*) of F. ■

Again the proof is a modification of that given previously.

Now we are in a position to explain how to reduce superfluous values of condition attributes form a decision table.

From Proposition 3 it follows that with every subset of attributes $P \subseteq A$ and object x we may associate set $[x]_P$. Thus with each row labelled by object x in the decision table, and set of condition attributes C, we may associate set $[x]_C = \bigcap_{a \in C} [x]_a$. But each set $[x]_a$ is uniquely determined by attribute value $f(x, a)$, hence in order to remove superfluous values of condition attributes we have to eliminate all superfluous equivalence classes from the equivalence class $[x]_C$. Thus problems of elimination of superfluous values of attributes and elimination of corresponding equivalence classes are equivalent and consequently we may use the Proposition 18 to eliminate superfluous values of attributes in the same manner as Propositions 14 and 15 was used to eliminate superfluous attributes.

3. Example of application

We shall illustrate the above defined concepts by means of an example of a decision table simplification describing the control of a cement kiln.

The decision table, taken from Mrózek [19], is shown in Table 15, where a, b, c and d are condition attributes, whereas e and f are decision attributes. We shall not discuss the meaning of the table, which can be found in the paper of Mrózek.

It is easy to check that the set of condition attributes is independent in absolute sense, but it is dependent with respect to the set of decision attributes. The only D — reduct of condition attributes is the set $\{a, c, d\}$.

Table 15

U	a	b	c	d	e	f
1	1	0	0	1	1	2
2	1	0	0	0	1	2
3	1	1	0	1	1	2
4	0	0	0	0	0	2
5	0	0	0	1	0	2
6	1	0	0	2	1	1
7	1	1	0	2	1	1
8	2	1	0	2	1	1
9	2	1	1	2	1	0
10	2	2	1	2	1	0
11	2	2	1	1	1	0
12	2	1	1	1	1	0
13	2	0	1	1	1	0

Thus after removing duplicate decision rules the decision Table 15 can be simplified as shown below.

Table 16

U	a	c	d	e	f
1	1	0	1	1	2
2	1	0	0	1	2
4	0	0	0	0	2
5	0	0	1	0	2
7	1	0	2	1	1
8	2	0	2	1	1
9	2	1	2	1	0
11	2	1	1	1	0

Let us also remark that both decision Tables 15 and 16 are deterministic.

Let us note that in the decision table there are four kinds of possible decisions. The possible decision are specified by the following pairs of values of decision attributes e and f: (1, 2), (0, 2), (1, 1) and (1, 0), denoted in what follows by I, II, III and IV, respectively.

Thus we can represent decision Table 16 as shown in the Table 17.

Table 17

U	a	c	d	e	f
1	1	0	1	I	
2	1	0	0		
4	0	0	0	II	
5	0	0	1		
7	1	0	2	III	
8	2	0	2		
9	2	1	2	IV	
11	2	1	1		

It is easy to check that in the first row we have two indispensable attribute-values a_1 and d_1 (a_i denotes value i of attribute (a) and $\{a_1, d_1\}$ is the core and also the reduct of the set corresponding to this row.

Proceeding in the same way we obtain the following table of value cores:

Table 18

U	a	c	d	e	f	
1	1		1			I
2	1		0			
4	0					II
5	0					
7			2			III
8		0				
9		1				IV
11		1				

Removing again duplicate rows from Table 18 we obtain Table 19.

Table 19

U	a	c	d	e	f	
1	1		1			I
2	1		0			
4	0					II
7			2			III
8		0				
9		1				IV

It can be easily seen that in the decision classes I, II and IV core values in each row are also reducts of values. For the decision class III however values a_2, c_0 do not form value reducts of corresponding rows. All possible value reducts for this rows are listed in tables below:

Table 20

U	a	c	d	e	f	
1	1		1			I
2	1		0			
4	0					II
7	1		2			III
8	2	0				
9		1				IV

Table 21

U	a	c	d	e	f	
1	1		1			I
2	1		0			
4	0					II
7	1		2			III
8		0	2			
9		1				IV

Table 22

U	a	c	d	e	f	
1	1		1			I
2	1		0			
4	0					II
7		0	2			III
8	2	0				
9		1				IV

Table 23

U	a	c	d	e	f	
1	1		1			I
2	1		0			
4	0					II
7		0	2			III
8		0	2			
9		1				IV

The above results means that in order to distinguish decision classes III from the remaining decision classes I, II and IV we can use four alternative sets of values of condition attributes as shown in Tables 20-23, and consequently we can represent our original decision table as shown for example in Table 24.

Table 24

U	a	c	d	e	f	
1	1	x	1			I
2	1	x	0			
4	0	x	x			II
8	x	0	2			III
9	x	1	x			IV

The Table 24 is obtained from Table 23 by removing duplicate row 7 and crosses in the table denote "don't care" values of attributes.

In summary, to simplify a decision table we should first find reducts of conditions attributes, remove duplicate rows and then find value-reducts of condition attributes and again, if necessary, remove duplicate rows. This method leads to a simple algorithm for decision table simplification or generation of decision rules (algorithms) from examples, which, according to our experiments, out performs other methods, in terms of achievable degree in the number of conditions and what more, gives all possible solutions to the problem. Compare for example Quinlan [40].

We conclude this section with the following remark.

Because, in general, a subset of attributes may have more that one reduct (relative reduct) the simplification of decision tables does not yield unique results. The table possibly can be also optimized according to pre assumed criteria.

Bibliographical remarks

The idea of an information system considered in this book was introduced first in Pawlak [32], and extended in Marek and Pawlak [15]. Information systems in the

context of rough sets were considered in Novotny and Pawlak [21–24], Marek Pawlak [16], Orłowska and Pawlak [29], and Pagliani [31].

A more general concept of information system, so called nondetermi information systems was introduced by Jaegerman [7, 8] and investigate Lipski Jr. [11, 12], Orłowska and Pawlak [30], Słowiński and Stefanowski [47] others.

Investigation of independence of attributes in information systems was prop in Pawlak [34]. Some mathematical consideration concerning this problem on find in Łoś [13]. It is also worthwhile to mention that the notion of independen attributes in the information system is a special case of a general notio independence in mathematics introduced by Marczewski [14] and investigate many authors (cf. Głazek [4]).

Dependency of attributes in information system was introduced in Pawlak and rough dependency – in Pawlak [37] and Novotny and Pawlak [26]. relationship between the notion of dependency in information systems and in relational data bases was compared and investigated by many authors (cf. Buszk ski and Orłowska [2], Rauszer [42–44].

The applications of information systems to decision tables in the framew of rough sets was proposed in Pawlak [36, 39], and investigated by Wong Ziarko [51], Boryczka and Słowiński [1].

References

[1] M. Boryczka & Słowiński, *Properties of Optimal Decision Algorithms from Decision Tables U Rough Sets.* Bull. Polish Acad. Sci. Tech. (to appear), 1988.

[2] W. Buszkowski, & E. Orłowska, *On the Logic of Database Dependencies.* Bull. Polish Acad. Math. **34** (5–6), 345–354, 1986.

[3] E. F. Codd, *A Relational Model of Data for Large Shared Data Banks.* Comm. ACM. **13** (6), 377, 1970.

[4] K. Głazek, *Some Old and New Problems in the Independence Theory.* Colloquium Mathematic XLII, 127–189.

[5] E. D. Hunt, J. Martin & P. J. Stone, *Experiments in Induction.* Academic Press, New York 1

[6] R. B. Hurley, *Decision Tables in Software Engineering.* Van Nostrand Reinhold Company, York 1983.

[7] M. Jaegerman, *Information Storage and Retrieval Systems with Incomplete Information. Par* Fundamenta Informaticae **2**, 17–41, 1978.

[8] M. Jaegerman, *Information Storage and Retrieval Systems with Incomplete Information. Part* Fundamenta Informaticae **2**, 141–166, 1978.

[9] Chen Keh-Hsun, Z. Raś and A. Skowron, *Attributes and Rough Properties in Informat Systems.* Proc. of Conf. on Information Science and Systems. March 14–16, 1984, Prince University, Princeton, pp. 362–366, 1984.

[10] H. Krasowski, *Aircraft Pilot Performance Evaluation with Rough Sets.* Ph. D. Dissertation Polish), 1988.

[11] W. Lipski Jr., *On Semantic Issues Connected with Incomplete Information Databases.* AC Transactions on Database Systems **4** (5), 269–296, 1979.

[12] W. Lipski Jr., *On Databases with Incomplete Information.* Journal of the ACM **28** (1), 41–70, 19

[13] J. Łoś, *Characteristic Sets of a System of Equivalence Relations.* Colloquium Mathematicum XLII, 291–293, 1979.
[14] E. Marczewski, *A general Scheme of Independence in Mathematics*, BA PS **6**, 731–736, 1958.
[15] W. Marek & Z. Pawlak, *Information Storage and Retrieval Systems – Mathematical Foundations.* Theoretical Computer Science **1**, 331–354, 1976.
[16] W. Marek & Z. Pawlak, *Rough Sets and Information Systems.* Fundamenta Informaticae VII (1), 105–115, 1984.
[17] M. Montalbano, *Decision Tables.* Science Research Associates. Chicago 1974.
[18] G. W. Moore, U. N. Riede & G. Sandritter, *Application of Quine's Nullities to a Quantitive Organelle Pathology.* Journal of Theoretical Biology **65**, 633–657, 1977.
[19] A. Mrózek, *Rough Sets and Some Aspects of Expert Systems Realization.* Proc. 7-th Internal Workshop on Expert Systems and their Applications, Avignon, France, 597–611, 1987.
[20] S. Muroga, *Logic Design and Switching Theory.* Wiley, New York 1979.
[21] M. Novotny & Z. Pawlak, *On a Representation of Rough Sets by Means of Information Systems.* Fundamenta Informaticae **6** (3–4), 286–296, 1983.
[22] M. Novotny & Z. Pawlak, *Black Box Analysis and Rough Top Equality.* Bull. Polish Acad. Sci. Math. **33** (1–2), 105–113, 1985.
[23] M. Novotny & Z. Pawlak, *Characterization of Rough Top Equalities and Rough Bottom Equalities.* Bull. Polish. Acad. Sci. Math. **33**, 91–97, 1985.
[24] M. Novotny & Z. Pawlak, *On Rough Equalities.* Bull. Polish. Acad. Sci. Math. **33** (1–2), 99–104, 1985.
[25] M. Novotny & Z. Pawlak, *Concept Forming and Black Boxes.* Bull. Polish Acad. Sci. Math. **35**, (1–2), 133–141, 1987.
[26] M. Novotny & Z. Pawlak, *Partial Dependency of Attributes.* Bull. Polish Acad. Sci. Math. (to appear), 1988.
[27] M. Novotny & Z. Pawlak, *Independence of Attributes.* Bull. Polish Acad. Sci. Math. (to appear), 1988.
[28] E. Orłowska, *Dependencies of attributes in Pawlak's Information Systems.* Fundamenta Informaticae **6** (3–4), 247–256, 1983.
[29] E. Orłowska & Z. Pawlak, *Expressive Power of Knowledge Representation.* International Journal of Man-Machine Studies **20**, 485–500, 1984.
[30] E. Orłowska & Z. Pawlak *Representation of Nondeterministic Information.* Theoretical Computer Science **29**, 27–39, 1984.
[31] P. Pagliani, *Polystructured Model Spaces as Algorithm Oriented Models for Approximation Spaces and Pawlak's Information Systems.* Facolta di Science dell Informazione, Internal Raport PAG/3, 1987.
[32] Z. Pawlak, *Mathematical Foundations of Information Retrieval.* CC PAS Report **101**, Warszawa 1973.
[33] Z. Pawlak, *Information Systems-Theoretical Foundations.* (The book in Polish) PWN, Warszawa 1981.
[34] Z. Pawlak, *Information Systems-Theoretical Foundations.* Information Systems **6**, 205–218, 1981.
[35] Z. Pawlak, *Rough Sets.* International Journal of Information and Computer Sciences **11**, 341–356, 1982.
[36] Z. Pawlak, *Rough Sets and Decision Tables.* Lecture Notes in Computer Science **208**. Springer Verlag 186–196, 1985.
[37] Z. Pawlak, *On Rough Dependency of Attributes in Information Systems.* Bull. Polish Acad. Sci. Tech. **33**, 551–559, 1985.
[38] Z. Pawlak, *On Decision Tables.* Bull. Polish Acad. Sci. Tech. **34** (9–10), 563–572, 1986.
[39] Z. Pawlak, *Decision Tables – A Rough Sets Approach.* Bull. of EATCS **33**, 85–96, 1987.
[40] J. R. Quinlan, *Discovering Rules from Large Collections of Examples: A Case Study.* In Michie D. Ed., Expert Systems in Micro-Electronic Age. Edinburgh: Edinburgh Press, 168–201, 1979.

[41] Z. Raś, *An Algebraic Approach to Information Retrieval Systems.* Int. Journal of Comp. and Inf. Sciences **17** (4), 275–293, 1982.

[42] C. M. Rauszer, *On Equivalence Between Indiscernibility Relations in Information Systems and a Fragment of Intuitionistic Logic.* Lecture Notes in Computer Science **208**, Springer Verlag, 298–318, 1985.

[43] C. M. Rauszer, *Algebraic and Logical Description of Functional and Multivalued Dependencies.* Proc. of the Sec. Int. Symp. on Methodologies for Intelligent Systems. October 17, 1987, Charlotte, North Holland, 145–155, 1987.

[44] C. M. Rauszer, *Algebraic Properties of Functional Dependencies.* Bull. Polish Acad. Sci. Math. (to appear), 1988.

[45] G. Salton, *Automatic Information Organization and Retrieval.* McGraw-Hill, New York, 1968.

[46] K. Słowiński, R. Słowiński and R. Stefanowski, *Rough Sets Approach to Analysis of Data from Peritoneal Lavage in Acute Pancreatisis.* Medical Informatics 13, 143–156, 1988.

[47] R. Słowiński & R. Stefanowski, *Rough Classification in Incomplete Information Systems.* Mathematical Modeling (to appear), 1988.

[48] R. Wille, *Restructing Lattice Theory: An Approach Based on Hierarchies of Concepts.* In: I. Rival (Ed.), Ordered Sets, Reidel, Dordrecht-Boston, 445–470, 1982.

[49] R. Wille, *Liniendiagramme Hierarhischer Begriffssysteme.* Technische Hochschule, Preprint No 812, Darmstad 1984.

[50] A. Wiweger, *Knowledge Representation Systems, Logical Kits and Contexts.* Bull. Polish Acad. Sci. Math. (to appear), 1988.

[51] S. K. M. Wong & W. Ziarko, *On Optimal Decision Rules in Decision Tables.* Bull. Polish Acad. Sci. Math. **33** (11–12) 693–696, 1986.

[52] W. Ziarko, *On Reduction of Knowledge Representation.* Proc. of the Second Int. Symposium on Methodologies for Intelligent Systems (Coll. Series), 99–111, 1984.