Some Issues on Rough Sets

Zdzisław Pawlak[1,2]

[1] Institute for Theoretical and Applied Informatics
Polish Academy of Sciences
ul. Bałtycka 5, 44-100 Gliwice, Poland
[2] Warsaw School of Information Technology[*]
ul. Newelska 6, 01-447 Warsaw, Poland
`zpw@ii.pw.edu.pl`

1 Introduction

The aim of this paper is to give rudiments of rough set theory and present some recent research directions proposed by the author.

Rough set theory is a new mathematical approach to imperfect knowledge.

The problem of imperfect knowledge has been tackled for a long time by philosophers, logicians and mathematicians. Recently it became also a crucial issue for computer scientists, particularly in the area of artificial intelligence. There are many approaches to the problem of how to understand and manipulate imperfect knowledge. The most successful one is, no doubt, the fuzzy set theory proposed by Lotfi Zadeh [1].

Rough set theory proposed by the author in [2] presents still another attempt to this problem. This theory has attracted attention of many researchers and practitioners all over the world, who have contributed essentially to its development and applications. Rough set theory overlaps with many other theories. However we will refrain to discuss these connections here. Despite this, rough set theory may be considered as an independent discipline in its own right.

Rough set theory has found many interesting applications. The rough set approach seems to be of fundamental importance to AI and cognitive sciences, especially in the areas of machine learning, knowledge acquisition, decision analysis, knowledge discovery from databases, expert systems, inductive reasoning and pattern recognition.

The main advantage of rough set theory in data analysis is that it does not need any preliminary or additional information about data – like probability in statistics, or basic probability assignment in Dempster-Shafer theory, grade of membership or the value of possibility in fuzzy set theory.

One can observe the following about the rough set approach:

- introduction of efficient algorithms for finding hidden patterns in data,
- determination of minimal sets of data (data reduction),
- evaluation of the significance of data,
- generation of sets of decision rules from data,

---

[*] Former University of Information Technology and Management.

- easy-to-understand formulation,
- straightforward interpretation of obtained results,
- suitability of many of its algorithms for parallel processing.

Rough set theory has been extended in many ways (see, e.g., [3–17]) but we will not discuss these issues in this paper.

Basic ideas of rough set theory and its extensions, as well as many interesting applications can be found in books (see, e.g., [18–27, 12, 28–30]), special issues of journals (see, e.g., [31–34, 34–38]), proceedings of international conferences (see, e.g., [39–49]), tutorials (e.g., [50–53]), and on the internet (see, e.g., www.roughsets.org, logic.mimuw.edu.pl,rsds.wsiz.rzeszow.pl).

The paper is organized as follows:

Section 2 (Basic Concepts) contains general formulation of basic ideas of rough set theory together with brief discussion of its place in classical set theory.

Section 3 (Rough Sets and Reasoning from Data) presents the application of rough set concept to reason from data (data mining).

Section 4 (Rough Sets and Bayes' Theorem) gives a new look on Bayes' theorem and shows that Bayes' rule can be used differently to that offered by classical Bayesian reasoning methodology.

Section 5 (Rough Sets and Conflict Analysis) discusses the application of rough set concept to study conflict.

In Section 6 (Data Analysis and Flow Graphs) we show that many problems in data analysis can be boiled down to flow analysis in a flow network.

This paper is a modified version of lectures delivered at the Taragona University seminar on Formal Languages and Rough Sets in August 2003.

2   Rough Sets – Basic Concepts

2.1   Introduction

In this section we give some general remarks on a concept of a set and the place of rough sets in set theory.

The concept of a set is fundamental for the whole mathematics. Modern set theory was formulated by George Cantor [54].

Bertrand Russell discovered that the intuitive notion of a set proposed by Cantor leads to antinomies [55]. Two kinds of remedy for this discontent have been proposed: axiomatization of Cantorian set theory and alternative set theories.

Another issue discussed in connection with the notion of a set or a concept is vagueness (see, e.g., [56–61]). Mathematics requires that all mathematical notions (including set) must be exact (Gottlob Frege [62]). However, philosophers and recently computer scientists have become interested in vague concepts.

In fuzzy set theory vagueness is defined by graduated membership.

Rough set theory expresses vagueness, not by means of membership, but employing a boundary region of a set. If the boundary region of a set is empty it means that the set is crisp, otherwise the set is rough (inexact). Nonempty boundary region of a set means that our knowledge about the set is not sufficient to define the set precisely.

The detailed analysis of sorites paradoxes for vague concepts using rough sets and fuzzy sets is presented in [63].

In this section the relationship between sets, fuzzy sets and rough sets will be outlined and briefly discussed.

2.2   Sets

The notion of a set is not only basic for mathematics but it also plays an important role in natural language. We often speak about sets (collections) of various objects of interest, e.g., collection of books, paintings, people etc. Intuitive meaning of a set according to some dictionaries is the following:

"A number of things of the same kind that belong or are used together."

Webster's Dictionary

"Number of things of the same kind, that belong together because they are similar or complementary to each other."

The Oxford English Dictionary

Thus a set is a collection of things which are somehow related to each other but the nature of this relationship is not specified in these definitions.

In fact these definitions are due to Cantor [54], which reads as follows:

"Unter einer Mannigfaltigkeit oder Menge verstehe ich nämlich allgenein jedes Viele, welches sich als Eines denken lässt, d.h. jeden Inbegriff bestimmter Elemente, welcher durch ein Gesetz zu einem Ganzen verbunden werden kann."

Thus according to Cantor a set is a collection of any objects, which according to some law can be considered as a whole.

All mathematical objects, e.g., relations, functions, numbers, etc., are some kind of sets. In fact set theory is needed in mathematics to provide rigor.

Russell discovered that the Cantorian notion of a set leads to antinomies (contradictions). One of the best known antinomies called the powerset antinomy goes as follows: consider (infinite) set X of all sets. Thus X is the greatest set. Let Y denote the set of all subsets of X. Obviously Y is greater then X, because the number of subsets of a set is always greater the number of its elements. Hence X is not the greatest set as assumed and we arrived at contradiction.

Thus the basic concept of mathematics, the concept of a set, is contradictory. This means that a set cannot be a collection of arbitrary elements as was stipulated by Cantor.

As a remedy for this defect several improvements of set theory have been proposed. For example,

- Axiomatic set theory (Zermello and Fraenkel, 1904).
- Theory of types (Whitehead and Russell, 1910).
- Theory of classes (v. Neumann, 1920).

All these improvements consist in restrictions, put on objects which can form a set. The restrictions are expressed by properly chosen axioms, which say how

the set can be build. They are called, in contrast to Cantors' intuitive set theory, axiomatic set theories.

Instead of improvements of Cantors' set theory by its axiomatization, some mathematicians proposed escape from classical set theory by creating completely new idea of a set, which would free the theory from antinomies. Some of them are listed below.

– Mereology (Leśniewski, 1915).
– Alternative set theory (Vopenka, 1970).
– "Penumbral" set theory (Apostoli and Kanada, 1999).

No doubt the most interesting proposal was given by Stanisaw Leśniewski [64], who proposed instead of membership relation between elements and sets, employed in classical set theory, the relation of "being a part". In his set theory, called *mereology*, this relation is a fundamental one.

None of the three mentioned above "new" set theories were accepted by mathematicians, however Leśniewski's mereology attracted some attention of philosophers and recently also computer scientists, (e.g., Lech Polkowski and Andrzej Skowron [6]).

In classical set theory a set is uniquely determined by its elements. In other words, this means that every element must be uniquely classified as belonging to the set or not. In contrast, the notion of a beautiful painting is vague, because we are unable to classify uniquely all paintings into two classes: beautiful and not beautiful. Thus *beauty* is not a precise but a vague concept. That is to say the notion of a set is a *crisp* (precise) one. For example, the set of odd numbers is crisp because every number is either odd or even. In mathematics we have to use crisp notions, otherwise precise reasoning would be impossible. However philosophers for many years were interested also in *vague* (imprecise) notions.

Almost all concepts we are using in natural language are vague. Therefore common sense reasoning based on natural language must be based on vague concepts and not on classical logic. This is why vagueness is important for philosophers and recently also for computer scientists.

Vagueness is usually associated with the boundary region approach (i.e., existence of objects which cannot be uniquely classified to the set or its complement) which was first formulated in 1893 by the father of modern logic Gottlob Frege [62], who wrote:

"Der Begriff muss scharf begrenzt sein. Einem unscharf begrenzten Begriffe würde ein Bezirk entsprechen, der nicht überall eine scharfe Grenzlinie hätte, sondern stellenweise ganz verschwimmend in die Umgebung überginge. Das wäre eigentlich gar kein Bezirk; und so wird ein unscharf definirter Begriff mit Unrecht Begriff genannt. Solche begriffsartige Bildungen kann die Logik nicht als Begriffe anerkennen; es ist unmöglich, von ihnen genaue Gesetze aufzustellen. Das Gesetz des ausgeschlossenen Dritten ist ja eigentlich nur in anderer Form die Forderung, dass der Begriff scharf begrenzt sei. Ein beliebiger Gegenstand x fällt entweder unter der Begriff y, oder er fällt nicht unter ihn: *tertium non datur*."

Thus according to Frege

"The concept must have a sharp boundary. To the concept without a sharp boundary there would correspond an area that had not a sharp boundary-line all around."

That is, mathematics must use crisp, not vague concepts, otherwise it would be impossible to reason precisely.

Summing up, vagueness is

– Not allowed in mathematics.
– Interesting for philosophy.
– Necessary for computer science.

2.3   Fuzzy Sets

Zadeh proposed completely new, elegant approach to vagueness called *fuzzy set theory* [1]. In his approach an element can belong to a set to a degree $k(0 \leq k \leq 1)$, in contrast to classical set theory where an element must definitely belong or not to a set. For example, in classical set theory language we can state that one is definitely ill or healthy, whereas in fuzzy set theory we can say that someone is ill (or healthy) in 60 percent (i.e., in the degree 0.6). Of course, at once the question arises where we get the value of degree from. This issue raised a lot of discussion, but we will refrain from considering this problem here.

Thus fuzzy membership function can be presented as

$$\mu_X(x) \in < 0, 1 >,$$

where, X is a set and x is an element.

Let us observe that the definition of fuzzy set involves more advanced mathematical concepts, real numbers and functions, whereas in classical set theory the notion of a set is used as a fundamental notion of whole mathematics and is used to derive any other mathematical concepts, e.g., numbers and functions. Consequently fuzzy set theory cannot replace classical set theory, because, in fact, the theory is needed to define fuzzy sets.

Fuzzy membership function has the following properties:

$$\mu_{U-X}(x) = 1 - \mu_X(x) \text{ for any } x \in U, \tag{1}$$
$$\mu_{X \cup Y}(x) = max(\mu_X(x), \mu_Y(x)) \text{ for any } x \in U,$$
$$\mu_{X \cap Y}(x) = min(\mu_X(x), \mu_Y(x)) \text{ for any } x \in U.$$

This means that the membership of an element to the union and intersection of sets is uniquely determined by its membership to constituent sets. This is a very nice property and allows very simple operations on fuzzy sets, which is a very important feature both theoretically and practically.

Fuzzy set theory and its applications developed very extensively over recent years and attracted attention of practitioners, logicians and philosophers worldwide.

2.4   Rough Sets

Rough set theory [2, 18] is still another approach to vagueness. Similarly to fuzzy set theory it is not an alternative to classical set theory but it is embedded in it. Rough set theory can be viewed as a specific implementation of Frege's idea of vagueness, i.e., imprecision in this approach is expressed by a boundary region of a set, and not by a partial membership, like in fuzzy set theory.

Rough set concept can be defined quite generally by means of topological operations, *interior* and *closure*, called *approximations*.

Let us describe this problem more precisely. Suppose we are given a set of objects U called the *universe* and an indiscernibility relation $R \subseteq U \times U$, representing our lack of knowledge about elements of U. For the sake of simplicity we assume that R is an equivalence relation. Let X be a subset of U. We want to characterize the set X with respect to R. To this end we will need the basic concepts of rough set theory given below.

- The *lower approximation* of a set X (with respect to R) is the set of all objects, which can be for *certain* classified as X with respect to R (are *certainly X* with respect to R).
- The *upper approximation* of a set X (with respect to R) is the set of all objects which can be *possibly* classified as X with respect to R (are *possibly X* with respect to R).
- The *boundary region* of a set X (with respect to R) is the set of all objects, which can be classified neither as X nor as not-X with respect to R.

Now we are ready to give the definition of rough sets.

- Set X is *crisp* (exact with respect to R), if the boundary region of X is empty.
- Set X is *rough* (inexact with respect to R), if the boundary region of X is nonempty.

Thus a set is *rough* (imprecise) if it has nonempty boundary region; otherwise the set is *crisp* (precise). This is exactly the idea of vagueness proposed by Frege.

The approximations and the boundary region can be defined more precisely. To this end we need some additional notation.

The equivalence class of R determined by element x will be denoted by $R(x)$. The indiscernibility relation in certain sense describes our lack of knowledge about the universe. Equivalence classes of the indiscernibility relation, called *granules* generated by R, represent elementary portion of knowledge we are able to perceive due to R. Thus in view of the indiscernibility relation, in general, we are unable to observe individual objects but we are forced to reason only about the accessible granules of knowledge.

Formal definitions of approximations and the boundary region are as follows:

R-lower approximation of X

$$R_*(X) = \bigcup_{x \in U} \{R(x) : R(x) \subseteq X\}, \tag{2}$$

*R-upper approximation* of X

$$R^*(X) = \bigcup_{x \in U} \{R(x) : R(x) \cap X \neq \emptyset\}, \tag{3}$$

R-boundary region of X

$$BN_R(X) = R^*(X) - R_*(X). \tag{4}$$

As we can see from the definition approximations are expressed in terms of granules of knowledge. The lower approximation of a set is union of all granules which are entirely included in the set; the upper approximation – is union of all granules which have non-empty intersection with the set; the boundary region of set is the difference between the upper and the lower approximation.

In other words, due to the granularity of knowledge, rough sets cannot be characterized by using available knowledge. Therefore with every rough set we associate two *crisp* sets, called its *lower* and *upper approximation*. Intuitively, the lower approximation of a set consists of all elements that *surely* belong to the set, whereas the upper approximation of the set constitutes of all elements that *possibly* belong to the set, and the *boundary region* of the set consists of all elements that cannot be classified uniquely to the set or its complement, by employing available knowledge. Thus any rough set, in contrast to a crisp set, has a non-empty boundary region. The approximation definition is clearly depicted in Figure 1.
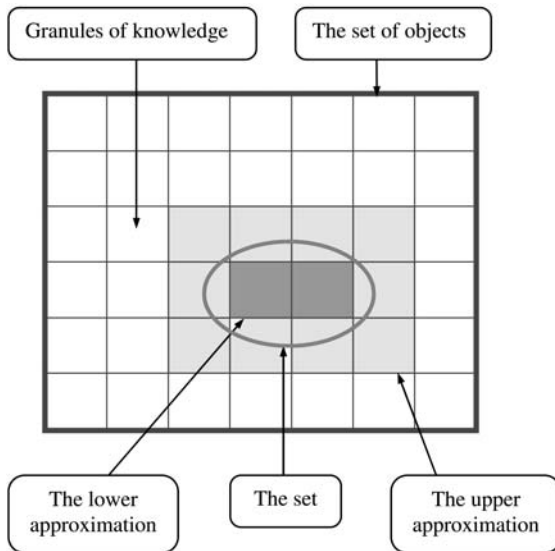
Fig. 1. A rough set

Approximations have the following properties:

$$R_*(X) \subseteq X \subseteq R^*(X), \tag{5}$$
$$R_*(\emptyset) = R^*(\emptyset) = \emptyset; R_*(U) = R^*(U) = U,$$
$$R^*(X \cup Y) = R^*(X) \cup R^*(Y),$$
$$R_*(X \cap Y) = R_*(X) \cap R_*(Y),$$
$$R_*(X \cup Y) \supseteq R_*(X) \cup R_*(Y),$$
$$R^*(X \cap Y) \subseteq R^*(X) \cap R^*(Y),$$
$$X \subseteq Y \to R_*(X) \subseteq R_*(Y) \& R^*(X) \subseteq R^*(Y),$$
$$R_*(-X) = -R^*(X),$$
$$R^*(-X) = -R_*(X),$$
$$R_* R_*(X) = R^* R_*(X) = R_*(X),$$
$$R^* R^*(X) = R_* R^*(X) = R^*(X).$$

It is easily seen that approximations are in fact interior and closure operations in a topology generated by the indiscernibility relation. Thus fuzzy set theory and rough set theory require completely different mathematical setting.

Rough sets can be also defined employing, instead of approximation, rough membership function [65]

$$\mu_X^R : U \to <0, 1>, \tag{6}$$

where

$$\mu_X^R(x) = \frac{card(X \cap R(x))}{card(R(x))}, \tag{7}$$

and $card(X)$ denotes the cardinality of X.

The rough membership function expresses conditional probability that x belongs to X given R and can be interpreted as a degree that x belongs to X in view of information about x expressed by R.

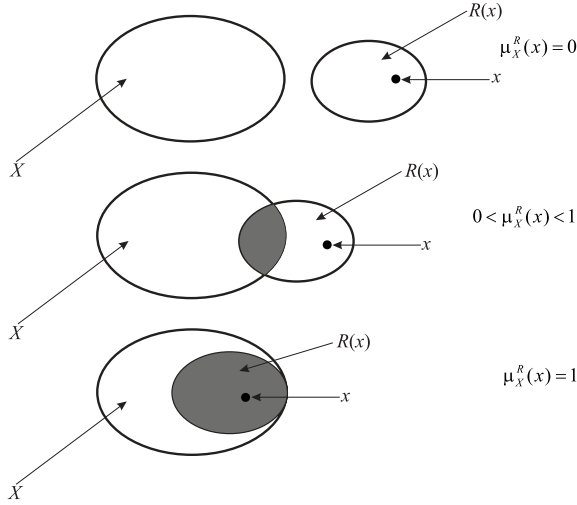The meaning of rough membership function can be depicted as shown in Figure 2.

The rough membership function can be used to define approximations and the boundary region of a set, as shown below:

$$R_*(X) = \{x \in U : \mu_X^R(x) = 1\}, \tag{8}$$
$$R^*(X) = \{x \in U : \mu_X^R(x) > 0\},$$
$$BN_R(X) = \{x \in U : 0 < \mu_X^R(x) < 1\}.$$

It can be shown that the membership function has the following properties [65]:

$$\mu_X^R(x) = 1 \text{ iff } x \in R_*(X), \tag{9}$$
$$\mu_X^R(x) = 0 \text{ iff } x \in U - R^*(X),$$
$$0 < \mu_X^R(x) < 1 \text{ iff } x \in BN_R(X),$$

Fig. 2. Rough membership function

$$\mu_{U-X}^{R}(x) = 1 - \mu_{X}^{R}(x) \text{ for any } x \in U,$$
$$\mu_{X \cup Y}^{R}(x) \geq max(\mu_{X}^{R}(x), \mu_{Y}^{R}(x)) \text{ for any } x \in U,$$
$$\mu_{X \cap Y}^{R}(x) \leq min(\mu_{X}^{R}(x), \mu_{Y}^{R}(x)) \text{ for any } x \in U.$$

From the properties it follows that the rough membership differs essentially from the fuzzy membership, because the membership for union and intersection of sets, in general, cannot be computed as in the case of fuzzy sets from their constituents membership. Thus formally the rough membership is a generalization of fuzzy membership. Besides, the rough membership function, in contrast to fuzzy membership function, has a probabilistic flavour.

Now we can give two definitions of rough sets.

Set X is *rough* with respect to R if $R_*(X) \neq R^*(X)$.

Set X *rough* with respect to R if for some x, $0 < \mu_X^R(x) < 1$.

It is interesting to observe that the above definitions are not equivalent [65], but we will not discuss this issue here.

One can define the following four basic classes of rough sets, i.e., four categories of vagueness:

$$R_*(X) \neq \emptyset \text{ and } R^*(X) \neq U, \text{ iff } X \text{ is } roughly \ R\text{-}definable, \quad (10)$$
$$R_*(X) = \emptyset \text{ and } R^*(X) \neq U, \text{ iff } X \text{ is } internally \ R\text{-}indefinable,$$
$$R_*(X) \neq \emptyset \text{ and } R^*(X) = U, \text{ iff } X \text{ is } externally \ R\text{-}definable,$$
$$R_*(X) = \emptyset \text{ and } R^*(X) = U, \text{ iff } X \text{ is } totally \ R\text{-}indefinable.$$

The intuitive meaning of this classification is the following.

If X is roughly R-*definable*, this means that we are able to decide for some elements of U whether they belong to X or $-X$, using R.

If X is internally R-indefinable, this means that we are able to decide whether some elements of U belong to $-X$, but we are unable to decide for any element of U, whether it belongs to X or not, using R.

If X is externally R-indefinable, this means that we are able to decide for some elements of U whether they belong to X, but we are unable to decide, for any element of U whether it belongs to $-X$ or not, using R.

If X is totally R-indefinable, we are unable to decide for any element of U whether it belongs to X or $-X$, using R.

A rough set can also be characterized numerically by the following coefficient

$$\alpha_R(X) = \frac{card(R_*(X))}{card(R^*(X))}, \tag{11}$$

called *accuracy of approximation*.

Obviously, $0 \leq \alpha_R(X) \leq 1$. If $\alpha_R(X) = 1$, X is *crisp* with respect to R (X is *precise* with respect to R), and otherwise, if $\alpha_R(X) < 1$, X is *rough* with respect to R (X is *vague* with respect to R).

It is interesting to compare definitions of classical sets, fuzzy sets and rough sets. Classical set is a primitive notion and is defined intuitively or axiomatically. Fuzzy sets are defined by employing the fuzzy membership function, which involves advanced mathematical structures, numbers and functions. Rough sets are defined by approximations. Thus this definition also requires advanced mathematical concepts.

Let us also mention that rough set theory clearly distinguishes two very important concepts, vagueness and uncertainty, very often confused in the AI literature. Vagueness is the property of sets and can be described by approximations, whereas uncertainty is the property of elements of a set and can expressed by the rough membership function.

3   Rough Sets and Reasoning from Data

3.1   Introduction

In this section we define basic concepts of rough set theory in terms of data, in contrast to general formulation presented in Section 2. This is necessary if we want to apply rough sets to reason from data.

In what follows we assume that, in contrast to classical set theory, we have some additional data (information, knowledge) about elements of a *universe of discourse*. Elements that exhibit the same features are indiscernible (similar) and form blocks that can be understood as *elementary granules* (*concepts*) of knowledge about the universe. For example, patients suffering from a certain disease, displaying the same symptoms are indiscernible and may be thought of as representing a granule (disease unit) of medical knowledge. These granules can be considered as elementary building blocks of knowledge. Elementary concepts can be combined into compound concepts, i.e., concepts that are uniquely determined in terms of elementary concepts. Any union of elementary sets is called a *crisp* set, and any other sets are referred to as *rough* (*vague, imprecise*).

3.2   An Example

Before we will formulate the above ideas more precisely let us consider a simple tutorial example.

Data are often presented as a table, columns of which are labeled by *attributes*, rows by *objects* of interest and entries of the table are *attribute values*. For example, in a table containing information about patients suffering from a certain disease objects are *patients* (strictly speaking their ID's), attributes can be, for example, *blood pressure, body temperature* etc., whereas the entry corresponding to object *Smith* and the attribute *blood preasure* can be *normal*. Such tables are known as *information tables, attribute-value* tables or *information system*. We will use here the term *information system*.

Below an example of information system is presented.

Suppose we are given data about 6 patients, as shown in Table 1.

Table 1. Exemplary information system

Patient	Headache	Muscle-pain	Temperature	Flu
p1	no	yes	high	yes
p2	yes	no	high	yes
p3	yes	yes	very high	yes
p4	no	yes	normal	no
p5	yes	no	high	no
p6	no	yes	very high	yes

Columns of the table are labeled by attributes (symptoms) and rows – by objects (patients), whereas entries of the table are attribute values. Thus each row of the table can be seen as information about specific patient.

For example, patient $p2$ is characterized in the table by the following attribute-value set

(*Headache, yes*), (*Muscle-pain, no*), (*Temperature, high*), (*Flu, yes*),

which form the information about the patient.

In the table patients $p2$, $p3$ and $p5$ are indiscernible with respect to the attribute *Headache*, patients $p3$ and $p6$ are indiscernible with respect to attributes *Muscle-pain* and *Flu*, and patients $p2$ and $p5$ are indiscernible with respect to attributes *Headache, Muscle-pain* and *Temperature*. Hence, for example, the attribute *Headache* generates two elementary sets $\{p2, p3, p5\}$ and $\{p1, p4, p6\}$, whereas the attributes *Headache* and *Muscle-pain* form the following elementary sets: $\{p1, p4, p6\}$, $\{p2, p5\}$ and $\{p3\}$. Similarly one can define elementary sets generated by any subset of attributes.

Patient $p2$ has flu, whereas patient $p5$ does not, and they are indiscernible with respect to the attributes *Headache, Muscle-pain* and *Temperature*, hence flu cannot be characterized in terms of attributes *Headache, Muscle-pain* and

Temperature. Hence $p2$ and $p5$ are the boundary-line cases, which cannot be properly classified in view of the available knowledge. The remaining patients $p1$, $p3$ and $p6$ display symptoms which enable us to classify them with certainty as having flu, patients $p2$ and $p5$ cannot be excluded as having flu and patient $p4$ for sure does not have flu, in view of the displayed symptoms. Thus the lower approximation of the set of patients having flu is the set $\{p1, p3, p6\}$ and the upper approximation of this set is the set $\{p1, p2, p3, p5, p6\}$, whereas the boundary-line cases are patients $p2$ and $p5$. Similarly $p4$ does not have flu and $p2$, $p5$ cannot be excluded as having flu, thus the lower approximation of this concept is the set $\{p4\}$ whereas – the upper approximation – is the set $\{p2, p4, p5\}$ and the boundary region of the concept "not flu" is the set $\{p2, p5\}$, the same as in the previous case.

3.3   Information Systems

Now, we are ready to formulate basic concepts of rough set theory using data.

Suppose we are given two finite, non-empty sets U and A, where U is the *universe*, and A – a set of *attributes*. The pair $S = (U, A)$ will be called an *information system*. With every attribute $a \in A$ we associate a set V_a, of its *values*, called the *domain* of a. Any subset B of A determines a binary relation $I(B)$ on U, which will be called an *indiscernibility relation*, and is defined as follows:

$$xI(B)y \text{ if and only if } a(x) = a(y) \text{ for every } a \in A, \tag{12}$$

where $a(x)$ denotes the value of attribute a for element x.

Obviously $I(B)$ is an equivalence relation. The family of all equivalence classes of $I(B)$, i.e., partition determined by B, will be denoted by $U/I(B)$, or simple U/B; an equivalence class of $I(B)$, i.e., block of the partition U/B, containing x will be denoted by $B(x)$.

If (x, y) belongs to $I(B)$ we will say that x and y are *B-indiscernible*. Equivalence classes of the relation $I(B)$ (or blocks of the partition U/B) are referred to as *B-elementary sets*. In the rough set approach the elementary sets are the basic building blocks (concepts) of our knowledge about reality.

Now approximations can be defined as follows:

$$B_*(X) = \{x \in U : B(x) \subseteq X\}, \tag{13}$$

$$B^*(X) = \{x \in U : B(x) \cap X \neq \emptyset\}, \tag{14}$$

called the *B-lower* and the *B-upper approximation* of X, respectively. The set

$$BN_B(X) = B^*(X) - B_*(X), \tag{15}$$

will be referred to as the *B-boundary region* of X.

If the boundary region of X is the empty set, i.e., $BN_B(X) = \emptyset$, then the set X is *crisp* (*exact*) with respect to B; in the opposite case, i.e., if $BN_B(X) \neq \emptyset$, the set X is referred to as *rough* (*inexact*) with respect to B.

The properties of approximations can be presented now as:

$$B_*(X) \subseteq X \subseteq B^*(X), \tag{16}$$
$$B_*(\emptyset) = B^*(\emptyset) = \emptyset, B_*(U) = B^*(U) = U,$$
$$B^*(X \cup Y) = B^*(X) \cup B^*(Y),$$
$$B_*(X \cap Y) = B_*(X) \cap B_*(Y),$$
$$X \subseteq Y \text{ implies } B_*(X) \subseteq B_*(Y) \text{ and } B^*(X) \subseteq B^*(Y),$$
$$B_*(X \cup Y) \supseteq B_*(X) \cup B_*(Y),$$
$$B^*(X \cap Y) \subseteq B^*(X) \cap B^*(Y),$$
$$B_*(-X) = -B^*(X),$$
$$B^*(-X) = -B_*(X),$$
$$B_*(B_*(X)) = B^*(B_*(X)) = B_*(X),$$
$$B^*(B^*(X)) = B_*(B^*(X)) = B^*(X).$$

3.4   Decision Tables

An information system in which we distinguish two classes of attributes, called *condition* and *decision* (*action*) attributes are called *decision tables*.

The condition and decision attributes define partitions of the decision table universe. We aim at approximation of the partition defined by the decision attributes by means of the partition defined by the condition attributes.

For example, in Table 1 attributes *Headache, Muscle-pain* and *Temperature* can be considered as condition attributes, whereas the attribute *Flu* – as a decision attribute. A decision table with condition attributes C and decision attributes D will be denoted by $S = (U, C, D)$.

Each row of a decision table determines a *decision rule*, which specifies decisions (*actions*) that should be taken when conditions pointed out by *condition* attributes are satisfied. For example, in Table 1 the condition (*Headache, no*), (*Muscle-pain, yes*), (*Temperature, high*) determines uniquely the decision (*Flu, yes*). Objects in a decision table are used as labels of decision rules.

Decision rules 2) and 5) in Table 1 have the same conditions but different decisions. Such rules are called *inconsistent* (*nondeterministic, conflicting*); otherwise the rules are referred to as *consistent* (*certain, deterministic, non-conflicting*). Sometimes consistent decision rules are called *sure* rules, and inconsistent rules are called *possible* rules. Decision tables containing inconsistent decision rules are called *inconsistent* (*nondeterministic, conflicting*); otherwise the table is *consistent* (*deterministic, non-conflicting*).

The number of consistent rules to all rules in a decision table can be used as *consistency factor* of the decision table, and will be denoted by $\gamma(C, D)$, where C and D are condition and decision attributes respectively. Thus if $\gamma(C, D) = 1$ the decision table is consistent and if $\gamma(C, D) \neq 1$ the decision table is inconsistent. For example, for Table 1, we have $\gamma(C, D) = 4/6$.

Decision rules are often presented in a form called *if... then...* rules. For example, rule 1) in Table 1 can be presented as follows

if (Headache,no) and (Muscle-pain,yes) and (Temperature,high) then (Flu,yes).

A set of decision rules is called a *decision algorithm.* Thus with each decision table we can associate a decision algorithm consisting of all decision rules occurring in the decision table.

We must however, make distinction between decision tables and decision algorithms. A decision table is a collection of data, whereas a decision algorithm is a collection of rules, e.g., logical expressions. To deal with data we use various mathematical methods, e.g., statistics but to analyze rules we must employ logical tools. Thus these two approaches are not equivalent, however for simplicity we will often present here decision rules in form of implications, without referring deeper to their logical nature, as it is often practiced in AI.

3.5   Dependency of Attributes

Another important issue in data analysis is discovering *dependencies* between attributes. Intuitively, a set of attributes D *depends totally* on a set of attributes C, denoted $C \Rightarrow D$, if all values of attributes from D are uniquely determined by values of attributes from C. In other words, D depends totally on C, if there exists a functional dependency between values of D and C. For example, in Table 1 there are no total dependencies whatsoever. If in Table 1, the value of the attribute *Temperature* for patient $p5$ were "*no*" instead of "*high*", there would be a total dependency $\{Temperature\} \Rightarrow \{Flu\}$, because to each value of the attribute *Temperature* there would correspond unique value of the attribute *Flu.*

We would need also a more general concept of dependency of attributes, called a *partial dependency* of attributes.

Let us depict the idea by example, referring to Table 1. In this table, for example, the attribute *Temperature* determines uniquely only some values of the attribute *Flu.* That is, (*Temperature, very high*) implies (*Flu, yes*), similarly (*Temperature, normal*) implies (*Flu, no*), but (*Temperature, high*) does not imply always (*Flu, yes*). Thus the partial dependency means that only some values of D are determined by values of C.

Formally dependency can be defined in the following way. Let D and C be subsets of A.

We will say that D *depends on* C in a *degree* k $(0 \leq k \leq 1)$, denoted $C \Rightarrow_k D$, if $k = \gamma(C, D)$.

If $k = 1$ we say that D *depends totally* on C, and if $k < 1$, we say that D *depends partially* (in a *degree* k) on C.

The coefficient k expresses the ratio of all elements of the universe, which can be properly classified to blocks of the partition U/D, employing attributes C.

Thus the concept of dependency of attributes is strictly connected with that of consistency of the decision table.

For example, for dependency {*Headache, Muscle-pain, Temperature*} ⇒ {Flu} we get $k = 4/6 = 2/3$, because four out of six patients can be uniquely classified as having flu or not, employing attributes *Headache, Muscle-pain and Temperature*.

If we were interested in how exactly patients can be diagnosed using only the attribute *Temperature*, that is – in the degree of the dependence {*Temperature*} ⇒ {*Flu*}, we would get $k = 3/6 = 1/2$, since in this case only three patients $p3, p4$ and $p6$ out of six can be uniquely classified as having flu. In contrast to the previous case patient $p4$ cannot be classified now as having flu or not. Hence the single attribute *Temperature* offers worse classification than the whole set of attributes *Headache, Muscle-pain* and *Temperature*. It is interesting to observe that neither *Headache* nor *Muscle-pain* can be used to recognize flu, because for both dependencies {*Headache*} ⇒ {*Flu*} and {*Muscle-pain*} ⇒ {*Flu*} we have $k = 0$.

It can be easily seen that if D depends totally on C then $I(C) \subseteq I(D)$. That means that the partition generated by C is finer than the partition generated by D. Observe, that the concept of dependency discussed above corresponds to that considered in relational databases.

If D *depends in degree* $k, 0 \leq k \leq 1$, on C, then

$$\gamma(C, D) = \frac{card(POS_C(D))}{card(U)}, \tag{17}$$

where

$$POS_C(D) = \bigcup_{X \in U/I(D)} C_*(X). \tag{18}$$

The expression $POS_C(D)$, called a *positive region* of the partition U/D with respect to C, is the set of all elements of U that can be uniquely classified to blocks of the partition U/D, by means of C.

Summing up: D is *totally* (*partially*) dependent on C, if *all* (*some*) elements of the universe U can be uniquely classified to blocks of the partition U/D, employing C.

3.6   Reduction of Attributes

We often face a question whether we can remove some data from a data table preserving its basic properties, that is – whether a table contains some superfluous data.

For example, it is easily seen that if we drop in Table 1 either the attribute *Headache* or *Muscle-pain* we get the data set which is equivalent to the original one, in regard to approximations and dependencies. That is we get in this case the same accuracy of approximation and degree of dependencies as in the original table, however using smaller set of attributes.

In order to express the above idea more precisely we need some auxiliary notions. Let B be a subset of A and let a belong to B.

- We say that a is *dispensable* in B if $I(B) = I(B - \{a\})$; otherwise a is *indispensable* in B.
- Set B is *independent* if all its attributes are indispensable.
- Subset B' of B is a *reduct* of B if B' is independent and $I(B') = I(B)$.

Thus a reduct is a set of attributes that preserves partition. This means that a reduct is the minimal subset of attributes that enables the same classification of elements of the universe as the whole set of attributes. In other words, attributes that do not belong to a reduct are superfluous with regard to classification of elements of the universe.

Reducts have several important properties. In what follows we will present two of them.

First, we define a notion of a *core of attributes*.

Let B be a subset of A. The *core* of B is the set off all indispensable attributes of B.

The following is an important property, connecting the notion of the core and reducts

$$Core(B) = \bigcap Red(B), \tag{19}$$

where $Red(B)$ is the set off all reducts of B.

Because the core is the intersection of all reducts, it is included in every reduct, i.e., each element of the core belongs to some reduct. Thus, in a sense, the core is the most important subset of attributes, for none of its elements can be removed without affecting the classification power of attributes.

To further simplification of an information table we can eliminate some values of attribute from the table in such a way that we are still able to discern objects in the table as the original one. To this end we can apply similar procedure as to eliminate superfluous attributes, which is defined next.

- We will say that the value of attribute $a \in B$, is *dispensable* for x, if $B(x) = B^a(x)$, where $B^a = B - \{a\}$; otherwise the value of attribute a is *indispensable* for x.
- If for every attribute $a \in B$ the value of a is indispensable for x, then B will be called *orthogonal* for x.
- Subset $B' \subseteq B$ is a *value reduct* of B for x, iff B' is orthogonal for x and $B(x) = B'(x)$.

The set of all indispensable values of attributes in B for x will be called the *value core* of B for x, and will be denoted $CORE^x(B)$.

Also in this case we have

$$CORE^x(B) = \bigcap Red^x(B), \tag{20}$$

where $Red^x(B)$ is the family of all reducts of B for x.

Suppose we are given a dependency $C \Rightarrow D$. It may happen that the set D depends not on the whole set C but on its subset C' and therefore we might be interested to find this subset. In order to solve this problem we need the notion of a *relative reduct*, which will be defined and discussed next.

Let $C, D \subseteq A$. Obviously if $C' \subseteq C$ is a D-reduct of C, then C' is a minimal subset of C such that

$$\gamma(C, D) = \gamma(C', D). \tag{21}$$

– We will say that attribute $a \in C$ is D-*dispensable* in C, if $POS_C(D) = POS_{(C-\{a\})}(D)$; otherwise the attribute a is D-*indispensable* in C.
– If all attributes $a \in C$ are C-indispensable in C, then C will be called D-*independent*.
– Subset $C' \subseteq C$ is a D-*reduct* of C, iff C' is D-independent and $POS_C(D) = POS_{C'}(D)$.

The set of all D-indispensable attributes in C will be called $D - core$ of C, and will be denoted by $CORE_D(C)$. In this case we have also the property

$$CORE_D(C) = \bigcap Red_D(C), \tag{22}$$

where $Red_D(C)$ is the family of all D-reducts of C.

If $D = C$ we will get the previous definitions.

For example, in Table 1 there are two relative reducts with respect to *Flu*, {*Headache, Temperature*} and {*Muscle-pain, Temperature*} of the set of condition attributes *Headache, Muscle-pain, Temperature*. That means that either the attribute *Headache* or *Muscle-pain* can be eliminated from the table and consequently instead of Table 1 we can use either Table 2 or Table 3. For Table 1 the relative core of with respect to the set {*Headache, Muscle-pain, Temperature*} is the *Temperature*. This confirms our previous considerations showing that Temperature is the only symptom that enables, at least, partial diagnosis of patients.

Table 2. Data table obtained from Table 1 by drooping the attribute Muscle-pain

Patient	Headache	Temperature	Flu
p1	no	high	yes
p2	yes	high	yes
p3	yes	very high	yes
p4	no	normal	no
p5	yes	high	no
p6	no	very high	yes

Table 3. Data table obtained from Table 1 by drooping the attribute Headache

Patient	Muscle-pain	Temperature	Flu
p1	yes	high	yes
p2	no	high	yes
p3	yes	very high	yes
p4	yes	normal	no
p5	no	high	no
p6	yes	very high	yes

We will need also a concept of a *value reduct* and *value core*. Suppose we are given a dependency $C \Rightarrow D$ where C is relative D-reduct of C. To further investigation of the dependency we might be interested to know exactly how values of attributes from D depend on values of attributes from C. To this end we need a procedure eliminating values of attributes form C which does not influence on values of attributes from D.

– We say that value of attribute $a \in C$, is *D-dispensable* for $x \in U$, if

$$C(x) \subseteq D(x) \text{ implies } C^a(x) \subseteq D(x),$$

  otherwise the value of attribute a is *D-indispensable* for x.
– If for every attribute $a \in C$ value of a is D-indispensable for x, then C will be called *D-independent (orthogonal)* for x.
– Subset $C' \subseteq C$ is a *D-reduct* of C for x (a value reduct), iff C' is D-independent for x and

$$C(x) \subseteq D(x) \text{ implies } C'(x) \subseteq D(x).$$

The set of all D-indispensable for x values of attributes in C will be called the $D - core$ of C for x (the value core), and will be denoted $CORE_D^x(C)$.

We have also the following property

$$CORE_D^x(C) = \bigcap Red_D^x(C), \tag{23}$$

where $Red_D^x(C)$ is the family of all D-reducts of C for x.

Using the concept of a value reduct, Table 2 and Table 3 can be simplified and we obtain Table 4 and Table 5, respectively.

For Table 4 we get its representation by means of rules

if (*Headache, no*) *and* (*Temperature, high*) *then* (*Flu, yes*),
if (*Headache, yes*) *and* (*Temperature, high*) *then* (*Flu, yes*),
if (*Temperature, very high*) *then* (*Flu, yes*),
if (*Temperature, normal*) *then* (*Flu, no*),
if (*Headache, yes*) *and* (*Temperature, high*) *then* (*Flu, no*),
if (*Temperature, very high*) *then* (*Flu, yes*).

For Table 5 we have

if (*Muscle-pain, yes*) *and* (*Temperature, high*) *then* (*Flu, yes*),
if (*Muscle-pain, no*) *and* (*Temperature, high*) *then* (*Flu, yes*),
if (*Temperature, very high*) *then* (*Flu, yes*),
if (*Temperature, normal*) *then* (*Flu, no*),
if (*Muscle-pain, no*) *and* (*Temperature, high*) *then* (*Flu, no*),
if (*Temperature, very high*) *then* (*Flu, yes*).

Table 4. Simplified Table 2

Patient	Headache	Temperature	Flu
p1	no	high	yes
p2	yes	high	yes
p3	–	very high	yes
p4	–	normal	no
p5	yes	high	no
p6	–	very high	yes

Table 5. Simplified Table 3

Patient	Muscle-pain	Temperature	Flu
p1	yes	high	yes
p2	no	high	yes
p3	–	very high	yes
p4	–	normal	no
p5	no	high	no
p6	–	very high	yes

The following important property

a) $B' \Rightarrow B - B'$, where B' is a reduct of B,

connects reducts and dependency.

Besides, we have:

b) If $B \Rightarrow C$, then $B \Rightarrow C'$, for every $C' \subseteq C$,

in particular

c) If $B \Rightarrow C$, then $B \Rightarrow \{a\}$, for every $a \in C$.

Moreover, we have:

d) If B' is a reduct of B, then neither $\{a\} \Rightarrow \{b\}$ nor $\{b\} \Rightarrow \{a\}$ holds, for every $a, b \in B'$, i.e., all attributes in a reduct are pairwise independent.

3.7   Indiscernibility Matrices and Functions

To compute easily reducts and the core we will use discernibility matrix [66], which is defined next.

By an discernibility matrix of $B \subseteq A$ denoted $M(B)$ we will mean $n \times n$ matrix with entries defined by:

$$c_{ij} = \{a \in B : a(x_i) \neq a(x_j)\} \text{ for } i, j = 1, 2, \ldots, n. \tag{24}$$

Thus entry c_{ij} is the set of all attributes which discern objects x_i and x_j.

The discernibility matrix $M(B)$ assigns to each pair of objects x and y a subset of attributes $\delta(x,y) \subseteq B$, with the following properties:

$$\delta(x,x) = \emptyset, \tag{25}$$
$$\delta(x,y) = \delta(y,x),$$
$$\delta(x,z) \subseteq \delta(x,y) \cup \delta(y,z).$$

These properties resemble properties of semi-distance, and therefore the function δ may be regarded as *qualitative semi-matrix* and $\delta(x,y)$ – *qualitative semi-distance*. Thus the discernibility matrix can be seen as a *semi-distance* (*qualitative*) matrix.

Let us also note that for every $x, y, z \in U$ we have

$$card(\delta(x,x)) = 0, \tag{26}$$
$$card(\delta(x,y)) = card(\delta(y,x)),$$
$$card(\delta(x,z)) \leq card(\delta(x,y)) + card(\delta(y,z)).$$

It is easily seen that the core is the set of all single element entries of the discernibility matrix $M(B)$, i.e.,

$$CORE(B) = \{a \in B : c_{ij} = \{a\}, \text{ for some } i,j\}. \tag{27}$$

Obviously $B' \subseteq B$ is a reduct of B, if B' is the minimal (with respect to inclusion) subset of B such that

$$B' \cap c \neq \emptyset \text{ for any nonempty entry } c \ (c \neq \emptyset) \text{ in } M(B). \tag{28}$$

In other words reduct is the minimal subset of attributes that discerns all objects discernible by the whole set of attributes.

Every discernibility matrix $M(B)$ defines uniquely a *discernibility* (*boolean*) function $f(B)$ defined as follows.

Let us assign to each attribute $a \in B$ a binary boolean variable $\overline{a}$, and let $\Sigma\delta(x,y)$ denote Boolean sum of all Boolean variables assigned to the set of attributes $\delta(x,y)$. Then the discernibility function can be defined by the formula

$$f(B) = \prod_{(x,y) \in U^2} \{\Sigma\delta(x,y) : (x,y) \in U^2 \text{ and } \delta(x,y) \neq \emptyset\}. \tag{29}$$

The following property establishes the relationship between disjunctive normal form of the function $f(B)$ and the set of all reducts of B.

All constituents in the minimal disjunctive normal form of the function $f(B)$ are all reducts of B.

In order to compute the value core and value reducts for x we can also use the discernibility matrix as defined before and the discernibility function, which must be slightly modified:

$$f^x(B) = \prod_{y \in U} \{\Sigma\delta(x,y) : y \in U \text{ and } \delta(x,y) \neq \emptyset\}. \tag{30}$$

Relative reducts and core can be computed also using discernibility matrix, which needs slight modification

$$c_{ij} = \{a \in C : a(x_i) \neq a(x_j) \text{ and } w(x_i, x_j)\}, \tag{31}$$

where $w(x_i, x_j) \equiv x_i \in POS_C(D)$ and $x_j \notin POS_C(D)$ or
$\qquad\qquad x_i \notin POS_C(D)$ and $x_j \in POS_C(D)$ or
$\qquad\qquad x_i, x_j \in POS_C(D)$ and $(x_j, x_j) \notin I(D)$,

for $i, j = 1, 2, \ldots, n$.

If the partition defined by D is definable by C then the condition $w(x_i, x_j)$ in the above definition can be reduced to $(x_i, x_j) \notin I(D)$.

Thus entry c_{ij} is the set of all attributes which discern objects x_i and x_j that do not belong to the same equivalence class of the relation $I(D)$.

The remaining definitions need little changes.

The *D-core* is the set of all single element entries of the discernibility matrix $M_D(C)$, i.e.,

$$CORE_D(C) = \{a \in C : c_{ij} = (a), \text{ for some } i, j\}. \tag{32}$$

Set $C' \subseteq C$ is the *D-reduct* of C, if C' is the minimal (with respect to inclusion) subset of C such that

$$C' \cap c \neq \emptyset \text{ for any nonempty entry } c, (c \neq \emptyset) \text{ in } M_D(C). \tag{33}$$

Thus D-reduct is the minimal subset of attributes that discerns all equivalence classes of the relation $I(D)$.

Every discernibility matrix $M_D(C)$ defines uniquely a *discernibility (Boolean) function* $f_D(C)$ which is defined as before. We have also the following property:

All constituents in the disjunctive normal form of the function $f_D(C)$ are all D-reducts of C.

For computing value reducts and the value core for relative reducts we use as a starting point the discernibility matrix $M_D(C)$ and discernibility function will have the form:

$$f_D^x(C) = \prod_{y \in U} \{\Sigma \delta(x, y) : y \in U \text{ and } \delta(x, y) \neq \emptyset\}. \tag{34}$$

Let us illustrate the above considerations by computing relative reducts for the set of attributes {*Headache, Muscle-pain, Temperature*} with respect to *Flu*.

The corresponding discernibility matrix is shown in Table 6.

In Table 6 H, M, T denote *Headache, Muscle-pain* and *Temperature*, respectively.

The discernibility function for this table is

$$T(H + M)(H + M + T)(M + T),$$

Table 6. Discernibility matrix

	1	2	3	4	5	6
1						
2						
3						
4	T	H, M, T				
5	H, M		M, T			
6				T	H, M, T	

where $+$ denotes the boolean sum and the boolean multiplication is omitted in the formula.

After simplication the discernibility function using laws of Boolean algebra we obtain the following expression

$$TH + TH,$$

which says that there are two reducts TH and TM in the data table and T is the core.

3.8   Significance of Attributes and Approximate Reducts

As it follows from considerations concerning reduction of attributes, they cannot be equally important, and some of them can be eliminated from an information table without losing information contained in the table. The idea of attribute reduction can be generalized by introducing a concept of *significance of attributes*, which enables us evaluation of attributes not only by two-valued scale, *dispensable – indispensable*, but by assigning to an attribute a real number from the closed interval [0,1], expressing how important is an attribute in an information table.

Significance of an attribute can be evaluated by measuring effect of removing the attribute from an information table on classification defined by the table. Let us first start our consideration with decision tables.

Let C and D be sets of condition and decision attributes respectively and let a be a condition attribute, i.e., $a \in A$. As shown previously the number $\gamma(C, D)$ expresses a degree of consistency of the decision table, or the degree of dependency between attributes C and D, or accuracy of approximation of U/D by C. We can ask how the coefficient $\gamma(C, D)$ changes when removing the attribute a, i.e., what is the difference between $\gamma(C, D)$ and $\gamma(C - \{a\}, D)$. We can normalize the difference and define the significance of the attribute a as

$$\sigma_{(C,D)}(a) = \frac{(\gamma(C, D) - \gamma(C - \{a\}, D))}{\gamma(C, D)} = 1 - \frac{\gamma(C - \{a\}, D)}{\gamma(C, D)}, \qquad (35)$$

and denoted simple by $\sigma(a)$, when C and D are understood.

Obviously $0 \leq \sigma(a) \leq 1$. The more important is the attribute a the greater is the number $\sigma(a)$. For example for condition attributes in Table 1 we have the following results:

$$\sigma(\textit{Headache}) = 0,$$
$$\sigma(\textit{Muscle-pain}) = 0,$$
$$\sigma(\textit{Temperature}) = 0.75.$$

Because the significance of the attribute *Temperature* or *Muscle-pain* is zero, removing either of the attributes from condition attributes does not effect the set of consistent decision rules, whatsoever. Hence the attribute *Temperature* is the most significant one in the table. That means that by removing the attribute *Temperature*, 75% (three out of four) of consistent decision rules will disappear from the table, thus lack of the attribute essentially effects the "decisive power" of the decision table.

For a reduct of condition attributes, e.g., {*Headache, Temperature*}, we get

$$\sigma(\textit{Headache}) = 0.25,$$
$$\sigma(\textit{Temperature}) = 1.00.$$

In this case, removing the attribute *Headache* from the reduct, i.e., using only the attribute *Temperature*, 25% (one out of four) of consistent decision rules will be lost, and dropping the attribute *Temperature*, i.e., using only the attribute *Headache* 100% (all) consistent decision rules will be lost. That means that in this case making decisions is impossible at all, whereas by employing only the attribute *Temperature* some decision can be made.

Thus the coefficient $\sigma(a)$ can be understood as an error which occurs when attribute a is dropped. The significance coefficient can be extended to set of attributes as follows:

$$\varepsilon_{(C,D)}(B) = \frac{(\gamma(C,D) - \gamma(C-B,D))}{\gamma(C,D)} = 1 - \frac{\gamma(C-B,D)}{\gamma(C,D)}, \qquad (36)$$

denoted by $\varepsilon(B)$, if C and D are understood, where B is a subset of C.

If B is a reduct of C, then $\varepsilon(B) = 1$, i.e., removing any reduct from a set of decision rules unables to make sure decisions, whatsoever.

Any subset B of C will be called an *approximate reduct* of C, and the number

$$\varepsilon_{(C,D)}(B) = \frac{(\gamma(C,D) - \gamma(B,D))}{\gamma(C,D)} = 1 - \frac{\gamma(B,D)}{\gamma(C,D)}, \qquad (37)$$

denoted simple as $\varepsilon(B)$, will be called an *error of reduct approximation*. It expresses how exactly the set of attributes B approximates the set of condition attributes C. Obviously $\varepsilon(B) = 1 - \sigma(B)$ and $\varepsilon(B) = 1 - \varepsilon(C-B)$. For any subset B of C we have $\varepsilon(B) \leq \varepsilon(C)$. If B is a reduct of C, then $\varepsilon(B) = 0$.

For example, either of attributes *Headache* and *Temperature* can be considered as approximate reducts of {*Headache, Temperature*}, and

$$\varepsilon(\textit{Headache}) = 1,$$
$$\varepsilon(\textit{Temperature}) = 0.25.$$

But for the whole set of condition attributes {*Headache, Muscle-pain, Temperature*} we have also the following approximate reduct

$$\varepsilon(Headache,\ Muscle\text{-}pain) = 0.75.$$

The concept of an approximate reduct is a generalization of the concept of a reduct considered previously. The minimal subset B of condition attributes C, such that $\gamma(C, D) = \gamma(B, D)$, or $\varepsilon_{(C,D)}(B) = 0$ is a reduct in the previous sense. The idea of an approximate reduct can be useful in cases when a smaller number of condition attributes is preferred over accuracy of classification.

4   Rough Sets and Bayes' Theorem

4.1   Introduction

Bayes' theorem is the essence of statistical inference.

"The result of the Bayesian data analysis process is the posterior distribution that represents a revision of the prior distribution on the light of the evidence provided by the data" [67].

"Opinion as to the values of Bayes' theorem as a basic for statistical inference has swung between acceptance and rejection since its publication on 1763" [68].

Rough set theory offers new insight into Bayes' theorem [69–71]. The look on Bayes' theorem presented here is completely different to that studied so far using the rough set approach (see, e.g., [72–85]) and in the Bayesian data analysis philosophy (see, e.g., [67, 86, 68, 87]). It does not refer either to prior or posterior probabilities, inherently associated with Bayesian reasoning, but it reveals some probabilistic structure of the data being analyzed. It states that any data set (decision table) satisfies total probability theorem and Bayes' theorem. This property can be used directly to draw conclusions from data without referring to prior knowledge and its revision if new evidence is available. Thus in the presented approach the only source of knowledge is the data and there is no need to assume that there is any prior knowledge besides the data. We simple look what the data are telling us. Consequently we do not refer to any prior knowledge which is updated after receiving some data.

Moreover, the presented approach to Bayes' theorem shows close relationship between logic of implications and probability, which was first studied by Jan Łukasiewicz [88] (see also [89]). Bayes' theorem in this context can be used to "invert" implications, i.e., to give reasons for decisions. This is a very important feature of utmost importance to data mining and decision analysis, for it extends the class of problem which can be considered in this domains.

Besides, we propose a new form of Bayes' theorem where basic role plays strength of decision rules (implications) derived from the data. The strength of decision rules is computed from the data or it can be also a subjective assessment. This formulation gives new look on Bayesian method of inference and also simplifies essentially computations.

4.2   Bayes' Theorem

"In its simplest form, if H denotes an hypothesis and D denotes data, the theorem says that

$$P(H \mid D) = P(D \mid H) \times P(H)/P(D). \tag{38}$$

With $P(H)$ regarded as a probabilistic statement of belief about H before obtaining data D, the left-hand side $P(H \mid D)$ becomes an probabilistic statement of belief about H after obtaining D. Having specified $P(D \mid H)$ and $P(D)$, the mechanism of the theorem provides a solution to the problem of how to learn from data.

In this expression, $P(H)$, which tells us what is known about H without knowing of the data, is called the *prior* distribution of H, or the distribution of H *priori*. Correspondingly, $P(H \mid D)$, which tells us what is known about H given knowledge of the data, is called the *posterior* distribution of H given D, or the distribution of H a *posteriori* [87].

"A prior distribution, which is supposed to represent what is known about unknown parameters before the data is available, plays an important role in Bayesian analysis. Such a distribution can be used to represent prior knowledge or relative ignorance" [68].

4.3   Decision Tables and Bayes' Theorem

In this section we will show that decision tables satisfy Bayes' theorem but the meaning of this theorem differs essentially from the classical Bayesian methodology.

Every decision table describes decisions (actions, results etc.) determined, when some conditions are satisfied. In other words each row of the decision table specifies a decision rule which determines decisions in terms of conditions.

In what follows we will describe decision rules more exactly.

Let $S = (U, C, D)$ be a decision table. Every $x \in U$ determines a sequence $c_1(x), \ldots, c_n(x), d_1(x), \ldots, d_m(x)$ where $\{c_1, \ldots, c_n\} = C$ and $\{d_1, \ldots, d_m\} = D$

The sequence will be called a *decision rule induced by x* (in S) and denoted by $c_1(x), \ldots, c_n(x) \to d_1(x), \ldots, d_m(x)$ or in short $C \to_x D$.

The number $supp_x(C, D) = card(C(x) \cap D(x))$ will be called a *support* of the decision rule $C \to_x D$ and the number

$$\sigma_x(C, D) = \frac{supp_x(C, D)}{card(U)}, \tag{39}$$

will be referred to as the *strength* of the decision rule $C \to_x D$. With every decision rule $C \to_x D$ we associate a *certainty factor* of the decision rule, denoted $cer_x(C, D)$ and defined as follows:

$$cer_x(C, D) = \frac{card(C(x) \cap D(x))}{card(C(x))} = \frac{supp_x(C, D)}{card(C(x))} = \frac{\sigma_x(C, D)}{\pi(C(x))}, \tag{40}$$

where $\pi(C(X)) = \frac{card(C(x))}{card(U)}$.

The certainty factor may be interpreted as a conditional probability that y belongs to $D(x)$ given y belongs to $C(x)$, symbolically $\pi_x(D \mid C)$.

If $cer_x(C, D) = 1$, then $C \rightarrow_x D$ will be called a *certain decision rule* in S; if $0 < cer_x(C, D) < 1$ the decision rule will be referred to as an *uncertain decision rule* in S.

Besides, we will also use a *coverage factor* of the decision rule, denoted $cov_x(C, D)$ defined as

$$cov_x(C, D) = \frac{card(C(x) \cap D(x))}{card(D(x))} = \frac{supp_x(C, D)}{card(D(x))} = \frac{\sigma_x(C, D)}{\pi(D(x))}, \qquad (41)$$

where $\pi(D(X)) = \frac{card(D(x))}{card(U)}$.

Similarly

$$cov_x(C, D) = \pi_x(C \mid D). \qquad (42)$$

The certainty and coverage coefficients have been widely used for years by data mining and rough set communities. However, Łukasiewicz [88] (see also [89]) was first who used this idea to estimate the probability of implications.

If $C \rightarrow_x D$ is a decision rule then $C \rightarrow_x D$ will be called an *inverse decision rule*. The inverse decision rules can be used to give explanations (*reasons*) for a decision.

Let us observe that

$$cer_x(C, D) = \pi^C_{D(x)}(x) \text{ and } cov_x(C, D). \qquad (43)$$

That means that the certainty factor expresses the degree of membership of x to the decision class $D(x)$, given C, whereas the coverage factor expresses the degree of membership of x to condition class $C(x)$, given D.

Decision tables have important probabilistic properties which are discussed next.

Let $C \rightarrow_x D$ be a decision rule in S and let $\Gamma = C(x)$ and $\Delta = D(x)$. Then the following properties are valid:

$$\sum_{y \in \Gamma} cer_y(C, D) = 1, \qquad (44)$$

$$\sum_{y \in \Gamma} cov_y(C, D) = 1, \qquad (45)$$

$$\pi(D(x)) = \sum_{y \in \Gamma} cer_y(C, D) \cdot \pi(C(y)) = \sum_{y \in \Gamma} \sigma_y(C, D), \qquad (46)$$

$$\pi(C(x)) = \sum_{y \in \Delta} cov_y(C, D) \cdot \pi(D(y)) = \sum_{y \in \Delta} \sigma_y(C, D), \qquad (47)$$

$$cer_x(C, D) = \frac{cov_x(C, D) \cdot \pi(D(x))}{\sum\limits_{y \in \Gamma} cov_y(C, D) \cdot \pi(D(y))} = \frac{\sigma_x(C, D)}{\sum\limits_{y \in \Delta} \sigma_y(C, D)} = \frac{\sigma_x(C, D)}{\pi(C(x))}, \qquad (48)$$

$$cov_x(C, D) = \frac{cer_x(C, D) \cdot \pi(C(x))}{\sum_{y \in \Gamma} cer_y(C, D) \cdot \pi(C(y))} = \frac{\sigma_x(C, D)}{\sum_{y \in \Gamma} \sigma_x(C, D)} = \frac{\sigma_x(C, D)}{\pi(D(x))}. \quad (49)$$

That is, any decision table, satisfies (44)-(49). Observe that (46) and (47) refer to the well known *total probability theorem*, whereas (48) and (49) refer to *Bayes' theorem*.

Thus in order to compute the certainty and coverage factors of decision rules according to formula (48) and (49) it is enough to know the strength (support) of all decision rules only. The strength of decision rules can be computed from data or can be a subjective assessment.

4.4 Decision Language and Decision Algorithms

It is often useful to describe decision tables in logical terms. To this end we define a formal language called a *decision language*.

Let $S = (U, A)$ be an information system. With every $B \subseteq A$ we associate a formal language, i.e., a set of formulas $For(B)$. Formulas of $For(B)$ are built up from attribute-value pairs (a, v) where $a \in B$ and $v \in V_a$ by means of logical connectives $\wedge(and)$, $\vee(or)$, $\sim (not)$ in the standard way.

For any $\Phi \in For(B)$ by $\| \Phi \|_S$ we denote the set of all objects $x \in U$ satisfying Φ in S and refer to as the *meaning* of Φ in S.

The meaning $\| \Phi \|_S$ of Φ in S is defined inductively as follows:
$\| (a, v) \|_S = \{x \in U : a(v) = x\}$ for all $a \in B$ and $v \in V_a$, $\| \Phi \wedge \Psi \|_S = \| \Phi \|_S \cup \| \Psi \|_S$, $\| \Phi \wedge \Psi \|_S = \| \Phi \|_S \cap \| \Psi \|_S$, $\| \sim \Phi \|_S = U - \| \Phi \|_S$.

If $S = (U, C, D)$ is a decision table then with every row of the decision table we associate a decision rule, which is defined next.

A *decision rule* in S is an expression $\Phi \rightarrow_S \Psi$ or simply $\Phi \rightarrow \Psi$ if S is understood, read *if Φ then Ψ*, where $\Phi \in For(C)$, $\Psi \in For(D)$ and C, D are condition and decision attributes, respectively; Φ and Ψ are referred to as *conditions* part and *decisions* part of the rule, respectively.

The number $supp_S(\Phi, \Psi) = card((\| \Phi \wedge \Psi \|_S))$ will be called the *support* of the rule $\Phi \rightarrow \Psi$ in S. We consider a probability distribution $p_U(x) = 1/card(U)$ for $x \in U$ where U is the (non-empty) universe of objects of S; we have $p_U(X) = card(X)/card(U)$ for $X \subseteq U$. For any formula Φ we associate its probability in S defined by

$$\pi_S(\Phi) = p_U(\| \Phi \|_S). \quad (50)$$

With every decision rule $\Phi \rightarrow \Psi$ we associate a conditional probability

$$\pi_S(\Psi \mid \Phi) = p_U(\| \Psi \|_S \mid \| \Phi \|_S) \quad (51)$$

called the *certainty factor* of the decision rule, denoted $cer_S(\Phi, \Psi)$. We have

$$cer_S(\Phi, \Psi) = \pi_S(\Psi \mid \Phi) = \frac{card(\| \Phi \wedge \Psi \|_S)}{card(\| \Phi \|_S)}, \quad (52)$$

where $\| \Phi \|_S \neq \emptyset$.

If $\pi_S(\Psi \mid \Phi) = 1$, then $\Phi \rightarrow \Psi$ will be called a *certain decision* rule; if $0 < \pi_S(\Psi \mid \Phi) < 1$ the decision rule will be referred to as a *uncertain decision* rule.

There is an interesting relationship between decision rules and their approximations: certain decision rules correspond to the lower approximation, whereas the uncertain decision rules correspond to the boundary region.

Besides, we will also use a *coverage factor* of the decision rule, denoted $cov_S(\Phi, \Psi)$ defined by

$$\pi_S(\Phi \mid \Psi) = p_U(\| \Phi \|_S \mid \| \Psi \|_S). \tag{53}$$

Obviously we have

$$cov_S(\Phi, \Psi) = \pi_S(\Phi \mid \Psi) = \frac{card(\| \Phi \wedge \Psi \|_S)}{card(\| \Psi \|_S)}. \tag{54}$$

There are three possibilities to interpret the certainty and the coverage factors: statistical (frequency), logical (degree of truth) and mereological (degree of inclusion).

We will use here mainly the statistical interpretation, i.e., the certainty factors will be interpreted as the frequency of objects having the property Ψ in the set of objects having the property Φ and the coverage factor – as the frequency of objects having the property Φ in the set of objects having the property Ψ.

Let us observe that the factors are not assumed arbitrarily but are computed from the data.

The number

$$\sigma_S(\Phi, \Psi) = \frac{supp_S(\Phi, \Psi)}{card(U)} = \pi_S(\Psi \mid \Phi) \cdot \pi_S(\Phi), \tag{55}$$

will be called the *strength* of the decision rule $\Phi \rightarrow \Psi$ in S.

We will need also the notion of an equivalence of formulas.

Let Φ, Ψ be formulas in $For(A)$ where A is the set of attributes in $S = (U, A)$.

We say that Φ and Ψ are equivalent in S, or simply, equivalent if S is understood, in symbols $\Phi \equiv \Psi$, if and only if $\Phi \rightarrow \Psi$ and $\Psi \rightarrow \Phi$. This means that $\Phi \equiv$ if and only if $\| \Phi \|_S = \| \Psi \|_S$.

We need also approximate equivalence of formulas which is defined as follows:

$$\Phi \equiv_S \Psi \text{ if and only if } cer(\Phi, \Psi) = cov(\Phi, \Psi) = k. \tag{56}$$

Besides, we define also approximate equivalence of formulas with the accuracy ε $(0 \leq \varepsilon \leq 1$, which is defined as follows:

$$\Phi \equiv_{k,\varepsilon} \Psi \text{ if and only if } \quad k = min\{(cer(\Phi, \Psi), cov(\Phi, \Psi)\} \tag{57}$$
$$\text{and } |cer(\Phi, \Psi) - cov(\Phi, \Psi)| \leq \varepsilon.$$

Now, we define the notion of a decision algorithm, which is a logical counterpart of a decision table.

Let $Dec(S) = \{\Phi_i \rightarrow \Psi\}_{i=1}^m$, $m \geq 2$, be a set of decision rules in a decision table $S = (U, C, D)$.

1) If for every $\Phi \to \Psi$, $\Phi' \to \Psi' \in Dec(S)$ we have $\Phi = \Phi'$ or $\| \Phi \wedge \Phi' \|_S = \emptyset$, and $\Psi = \Psi'$ or $\| \Psi \wedge \Psi' \|_S = \emptyset$, then we will say that $Dec(S)$ is the set of pairwise *mutually exclusive* (*independent*) decision rules in S.

2) If $\| \bigwedge\limits_{i=1}^{m} \Phi_i \|_S = U$ and $\| \bigwedge\limits_{i=1}^{m} \Psi_i \|_S = U$ we will say that the set of decision rules $Dec(S)$ *covers* U.

3) If $\Phi \to \Psi \in Dec(S)$ and $supp_S(\Phi, \Psi) \neq 0$ we will say that the decision rule $\Phi \to \Psi$ is *admissible* in S.

4) If $\bigcup\limits_{X \in U/D} C_*(X) = \bigwedge\limits_{\Phi \to \Psi \in Dec^+(S)} \| \Phi \|_S$, where $Dec^+(S)$ is the set of all certain decision rules from $Dec(S)$, we will say that the set of decision rules $Dec(S)$ preserves the *consistency* part of the decision table $S = (U, C, D)$.

The set of decision rules $Dec(S)$ that satisfies 1), 2) 3) and 4), i.e., is independent, covers U, preserves the consistency of S and all decision rules $\Phi \to \Psi \in Dec(S)$ are admissible in S – will be called a *decision algorithm* in S. Hence, if $Dec(S)$ is a decision algorithm in S then the conditions of rules from $Dec(S)$ define in S a partition of U. Moreover, the *positive region of D with respect to C*, i.e., the set

$$\bigcup_{X \in U/D} C_*(X), \tag{58}$$

is partitioned by the conditions of some of these rules, which are certain in S.

If $\Phi \to \Psi$ is a decision rule then the decision rule $\Psi \to \Psi$ will be called an *inverse* decision rule of $\Phi \to \Psi$.

Let $Dec^*(S)$ denote the set of all inverse decision rules of $Dec(S)$.

It can be shown that $Dec^*(S)$ satisfies 1), 2), 3) and 4), i.e., it is a decision algorithm in S.

If $Dec(S)$ is a decision algorithm then $Dec^*(S)$ will be called an *inverse* decision algorithm of $Dec(S)$.

The inverse decision algorithm gives *reasons* (explanations) for decisions pointed out by the decision algorithms.

A decision algorithm is a description of a decision table in the decision language.

Generation of decision algorithms from decision tables is a complex task and we will not discuss this issue here, for it does not lie in the scope of this paper. The interested reader is advised to consult the references (see, e.g., [18, 66, 90–97, 50, 98–104] and the bibliography in these articles).

4.5   An Example

Let us now consider an example of decision table, shown in Table 7.

Attributes *Disease, Age* and *Sex* are condition attributes, whereas *test* is the decision attribute.

We want to explain the test result in terms of patients state, i.e., to describe attribute *Test* in terms of attributes *Disease, Age* and *Sex*.

Table 7. Exemplary decision table

Fact	Disease	Age	Sex	Test	Support
1	yes	old	man	+	400
2	yes	middle	woman	+	80
3	no	old	man	−	100
4	yes	old	man	−	40
5	no	young	woman	−	220
6	yes	middle	woman	−	60

Table 8. Certainty and coverage factors for decision table shown in Table 7

Fact	Strength	Certaint	Coverage
1	0.44	0.92	0.83
2	0.09	0.56	0.17
3	0.11	1.00	0.24
4	0.04	0.08	0.10
5	0.24	1.00	0.52
6	0.07	0.44	0.14

The strength, certainty and coverage factors for decision table are shown in Table 8.

Below a decision algorithm associated with Table 7 is presented.

1) *if (Disease, yes) and (Age, old) then (Test, +);*
2) *if (Disease, yes) and (Age, middle) then (Test, +);*
3) *if (Disease, no) then (Test, −);*
4) *if (Disease, yes) and (Age, old) then (Test, −);*
5) *if (Disease, yes) and (Age, middle) then (Test, −).*

The certainty and coverage factors for the above algorithm are given in Table 9.

Table 9. Certainty and coverage factors for the decision algorithm

Rule	Strength	Certaint	Coverage
1	0.44	0.92	0.83
2	0.09	0.56	0.17
3	0.36	1.00	0.76
4	0.04	0.08	0.10
5	0.24	0.44	0.14

The certainty factors of the decision rules lead the following conclusions:

– 92% ill and old patients have positive test result,
– 56% ill and middle age patients more positive test result,
– all healthy patients have negative test result,
– 8% ill and old patients have negative test result,
– 44% ill and old patients have negative test result.

In other words:

- ill and old patients most probably have positive test result (probability = 0.92),
- ill and middle age patients most probably have positive test result (probability = 0.56),
- healthy patients have certainly negative test result (probability = 1.00).

Now let us examine the inverse decision algorithm, which is given below:

1') *if* (*Test*, +) *then* (*Disease, yes*) *and* (*Age, old*);
2') *if* (*Test*, +) *then* (*Disease, yes*) *and* (*Age, middle*);
3') *if* (*Test*, −) *then* (*Disease, no*);
4') *if* (*Test*, −) *then* (*Disease, yes*) *and* (*Age, old*);
5') *if* (*Test*, −) *then* (*Disease, yes*) *and* (*Age, middle*).

Employing the inverse decision algorithm and the coverage factor we get the following explanation of test results:

- reason for positive test results are most probably patients disease and old age (probability = 0.83),
- reason for negative test result is most probably lack of the disease (probability = 0.76).

It follows from Table 7 that there are two interesting approximate equivalences of test results and the disease.

According to rule 1) the disease and old age are approximately equivalent to positive test result ($k = 0.83$, $\varepsilon = 0.11$), and lack of the disease according to rule 3) is approximately equivalent to negative test result ($k = 0.76$, $\varepsilon = 0.24$).

5   Rough Sets and Conflict Analysis

5.1   Introduction

Knowledge discovery in databases considered in the previous sections boiled down to searching for functional dependencies in the data set.

In this section we will discuss another kind of relationship in the data – not dependencies, but conflicts.

Formally, the conflict relation can be seen as a negation (not necessarily, classical) of indiscernibility relation which was used as a basis of rough set theory. Thus dependencies and conflict are closely related from logical point of view.

It turns out that the conflict relation can be used to the conflict analysis study.

Conflict analysis and resolution play an important role in business, governmental, political and lawsuits disputes, labor-management negotiations, military operations and others. To this end many mathematical formal models of conflict situations have been proposed and studied, e.g., [105–110].

Various mathematical tools, e.g., graph theory, topology, differential equations and others, have been used to that purpose.

Needless to say that game theory can be also considered as a mathematical model of conflict situations.

In fact there is no, as yet, "universal" theory of conflicts and mathematical models of conflict situations are strongly domain dependent.

We are going to present in this paper still another approach to conflict analysis, based on some ideas of rough set theory – along the lines proposed in [110]. We will illustrate the proposed approach by means of a simple tutorial example of voting analysis in conflict situations.

The considered model is simple enough for easy computer implementation and seems adequate for many real life applications but to this end more research is needed.

5.2   Basic Concepts of Conflict Theory

In this section we give after [110] definitions of basic concepts of the proposed approach.

Let us assume that we are given a finite, non-empty set U called the *universe*. Elements of U will be referred to as *agents*. Let a function $v : U \to \{-1, 0, 1\}$, or in short $\{-, 0, +\}$, be given assigning to every agent the number $-1, 0$ or 1, representing his opinion, view, voting result, etc. about some discussed issue, and meaning *against, neutral* and *favorable*, respectively.

The pair $S = (U, v)$ will be called a *conflict* situation. In order to express relations between agents we define three basic binary relations on the universe: *conflict, neutrality* and *alliance*. To this end we first define the following auxiliary function:

$$\phi_v(x, y) = \begin{cases} 1, & \text{if } v(x)v(y) = 1 \text{ or } x = y \\ 0, & \text{if } v(x)v(y) = 0 \text{ and } x \neq y \\ -1, & \text{if } v(x)v(y) = -1. \end{cases} \tag{59}$$

This means that, if $\phi_v(x, y) = 1$, agents x and y have the same opinion about issue v (are *allied* on v); if $\phi_v(x, y) = 0$ means that at least one agent x or y has neutral approach to issue a (is *neutral* on a), and if $\phi_v(x, y) = -1$, means that both agents have different opinions about issue v (are in *conflict* on v).

In what follows we will define three basic relations R_v^+, R_v^0 and R_v^- on U^2 called *alliance, neutrality* and *conflict* relations respectively, and defined as follows:

$$R_v^+(x, y) \text{ iff } \phi_v(x, y) = 1, \tag{60}$$
$$R_v^0(x, y) \text{ iff } \phi_v(x, y) = 0,$$
$$R_v^-(x, y) \text{ iff } \phi_v(x, y) = -1.$$

It is easily seen that the alliance relation has the following properties:

$$R_v^+(x, x), \tag{61}$$
$$R_v^+(x, y) \text{ implies } R_v^+(y, x),$$
$$R_v^+(x, y) \text{ and } R_v^+(y, z) \text{ implies } R_v^+(x, z),$$

i.e., R_v^+ is an *equivalence* relation. Each equivalence class of alliance relation will be called *coalition* with respect to v. Let us note that the last condition in (61) can be expressed as "a friend of my friend is my friend".

For the conflict relation we have the following properties:

$$\text{not } R_v^-(x,x), \tag{62}$$
$$R_v^-(x,y) \text{ implies } R_v^-(y,x),$$
$$R_v^-(x,y) \text{ and } R_v^-(y,z) \text{ implies } R_v^+(x,z),$$
$$R_v^-(x,y) \text{ and } R_v^+(y,z) \text{ implies } R_v^-(x,z).$$

The last two conditions in (62) refer to well known sayings "an enemy of my enemy is my friend" and "a friend of my enemy is my enemy".

For the neutrality relation we have:

$$\text{not } R_v^0(x,x), \tag{63}$$
$$R_v^0(x,y) = R_v^0(y,x).$$

Let us observe that in the conflict and neutrality relations there are no coalitions.

The following property holds: $R_v^+ \cup R_v^0 \cup R_v^- = U^2$ because if $(x,y) \in U^2$ then $\Phi_v(x,y) = 1$ or $\Phi_v(x,y) = 0$ or $\Phi_v(x,y) = -1$ so $(x,y) \in R_v^+$ or $(x,y) \in R_v^-$ or $(x,y) \in R_v^-$. All the three relations R_v^+, R_v^0 , R_v^- are pairwise disjoint, i.e., every pair of objects (x,y) belongs to exactly one of the above defined relations (is in conflict, is allied or is neutral).

With every conflict situation we will associate a *conflict graph*

$$G_S = (R_v^+, R_v^0, R_v^-). \tag{64}$$

An example of a conflict graph is shown in Figure 3. Solid lines are denoting conflicts, doted line – alliance, and neutrality, for simplicity, is not shown explicitly in the graph. Of course, B, C, and D form a coalition.
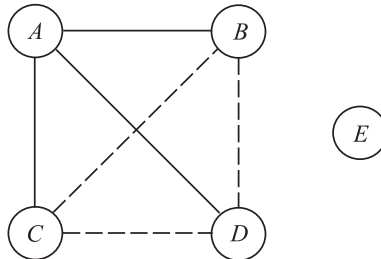
Fig. 3. Exemplary conflict graph

5.3   An Example

In this section we will illustrate the above presented ideas by means of a very simple tutorial example using concepts presented in the previous.

Table 10 presents a decision table in which the only condition attribute is *Party*, whereas the decision attribute is *Voting*. The table describes voting results in a parliament containing 500 members grouped in four political parties denoted A, B, C and D. Suppose the parliament discussed certain issue (e.g., membership of the country in European Union) and the voting result is presented in column *Voting*, where $+$, 0 and $-$ denoted *yes*, *abstention* and *no* respectively. The column *support* contains the number of voters for each option.

Table 10. Decision table with one condition attribute *Party* and the decision *Voting*

Fact	Party	Voting	Support
1	A	+	200
2	A	0	30
3	A	−	10
4	B	+	15
5	B	−	25
6	C	0	20
7	C	−	40
8	D	+	25
9	D	0	35
10	D	−	100

Table 11. Certainty and the coverage factors for Table 10

Fact	Strength	Certainty	Coverage
1	0.40	0.83	0.83
2	0.06	0.13	0.35
3	0.02	0.04	0.06
4	0.03	0.36	0.06
5	0.05	0.63	0.14
6	0.04	0.33	0.23
7	0.08	0.67	0.23
8	0.05	0.16	0.10
9	0.07	0.22	0.41
10	0.20	0.63	0.57

The strength, certainty and the coverage factors for Table 10 are given in Table 11.

From the certainty factors we can conclude, for example, that:

 − 83.3% of party A voted *yes*,
 − 12.5% of party A *abstained*,
 − 4.2% of party A voted *no*.

From the coverage factors we can get, for example, the following explanation of voting results:

- 83.3% *yes* votes came from party A,
- 6.3% *yes* votes came from party B,
- 10.4% *yes* votes came from party C.

6   Data Analysis and Flow Graphs

6.1   Introduction

Pursuit for data patterns considered so far referred to data tables. In this section we will consider data represented not in a form of data table but by means of graphs. We will show that this method od data representation leads to a new look on knowledge discovery, new efficient algorithms, and vide spectrum of novel applications.

The idea presented here are based on some concepts given by Łukasiewicz [88].

In [88] Łukasiewicz proposed to use logic as mathematical foundation of probability. He claims that probability is "purely logical concept" and that his approach frees probability from its obscure philosophical connotation. He recommends to replace the concept of *probability* by *truth values* of *indefinite propositions*, which are in fact propositional functions.

Let us explain this idea more closely. Let U be a non empty finite set, and let $\Phi(x)$ be a propositional function. The meaning of $\Phi(x)$ in U, denoted by $\|\Phi(x)\|$, is the set of all elements of U, that satisfies $\Phi(x)$ in U. The truth value of $\Phi(x)$ is defined by $card(\|\Phi(x)\|)/card(U)$. For example, if $U = \{1, 2, 3, 4, 5, 6\}$ and $\Phi(x)$ is the propositional function $x > 4$, then the truth value of $\Phi(x) = 2/6 = 1/3$. If the truth value of $\Phi(x)$ is 1, then the propositional function is *true*, and if it is 0, then the function is *false*. Thus the truth value of any propositional function is a number between 0 and 1. Further, it is shown that the truth values can be treated as probability and that all laws of probability can be obtained by means of logical calculus.

In this paper we show that the idea of Łukasiewicz can be also expressed differently. Instead of using truth values in place of probability, stipulated by Łukasiewicz, we propose, in this paper, using of deterministic flow analysis in flow networks (graphs). In the proposed setting, flow is governed by some probabilistic rules (e.g., Bayes' rule), or by the corresponding logical calculus proposed by Łukasiewicz, though, the formulas have entirely deterministic meaning, and need neither probabilistic nor logical interpretation. They simply describe flow distribution in flow graphs. However, flow graphs introduced here are different from those proposed by Ford and Fulkerson [111] for optimal flow analysis, because they model rather, e.g., flow distribution in a plumbing network, than the optimal flow.

The flow graphs considered in this paper are basically meant not to physical media (e.g., water) flow analysis, but to information flow examination in decision algorithms. To this end branches of a flow graph are interpreted as decision

rules. With every decision rule (i.e. branch) three coefficients are associated, the *strength, certainty* and *coverage factors*. In classical decision algorithms language they have probabilistic interpretation. Using Łukasiewicz's approach we can understand them as truth values. However, in the proposed setting they can be interpreted simply as flow distribution ratios between branches of the flow graph, without referring to their probabilistic or logical nature.

This interpretation, in particular, leads to a new look on Bayes' theorem, which in this setting, has entirely deterministic explanation (see also [86]).

The presented idea can be used, among others, as a new tool for data analysis, and knowledge representation.

We start our considerations giving fundamental definitions of a flow graph and related notions. Next, basic properties of flow graphs are defined and investigated. Further, the relationship between flow graphs and decision algorithms is discussed. Finally, a simple tutorial example is used to illustrate the consideration.

6.2   Flow Graphs

A flow graph is a *directed, acyclic*, finite graph $G = (N, \mathcal{B}, \phi)$, where N is a set of *nodes*, $\mathcal{B} \subseteq N \times N$ is a set of *directed branches*, $\phi : \mathcal{B} \to R^+$ is a *flow function* and R^+ is the set of non-negative reals.

If $(x, y) \in \mathcal{B}$ then x is an *input* of y and y is an *output* of x.

If $x \in N$ then $I(x)$ is the set of all inputs of x and $O(x)$ is the set of all outputs of x.

Input and *output* of a graph G are defined $I(G) = \{x \in N : I(x) = \emptyset\}$, $O(G) = \{x \in N : O(x) = \emptyset\}$.

Inputs and outputs of G are *external nodes* of G; other nodes are *internal nodes* of G.

If $(x, y) \in \mathcal{B}$ then $\phi(x, y)$ is a *troughflow* from x to y. We will assume in what follows that $\phi(x, y) \neq 0$ for every $(x, y) \in \mathcal{B}$.

With every node x of a flow graph G we associate its *inflow*

$$\phi_+(x) = \sum_{y \in I(x)} \phi(y, x), \tag{65}$$

and *outflow*

$$\phi_-(x) = \sum_{y \in O(x)} \phi(x, y). \tag{66}$$

Similarly, we define an inflow and an outflow for the whole flow graph G, which are defined as

$$\phi_+(G) = \sum_{x \in I(G)} \phi_-(x), \tag{67}$$

$$\phi_-(G) = \sum_{x \in O(G)} \phi_+(x). \tag{68}$$

We assume that for any internal node x, $\phi_+(x) = \phi_-(x) = \phi(x)$, where is a *troughflow* of node x.

Obviously,$\phi_+(G) = \phi_-(G) = \phi(G)$, where $\phi(G)$ is a *troughflow* of graph G. The above formulas can be considered as *flow conservation equations* [111]. We will define now a *normalized flow graph*.

A normalized flow graph is a *directed, acyclic, finite* graph $G = (N, \mathcal{B}, \sigma)$, where N is a set of *nodes*, $\mathcal{B} \subseteq N \times N$ is a set of *directed branches* and $\sigma : \mathcal{B} \rightarrow <0, 1>$ is a *normalized flow* of (x, y) and

$$\sigma(x, y) = \frac{\sigma(x, y)}{\sigma(G)}, \tag{69}$$

is *strength* of (x, y). Obviously, $0 \leq \sigma(x, y) \leq 1$. The strength of the branch expresses simply the percentage of a total flow through the branch.

In what follows we will use normalized flow graphs only, therefore by a flow graphs we will understand normalized flow graphs, unless stated otherwise.

With every node x of a flow graph G we associate its normalized *inflow* and *outflow* defined as

$$\sigma_+(x) = \frac{\phi_+(x)}{\phi(G)} = \sum_{y \in I(x)} \sigma(y, x), \tag{70}$$

$$\sigma_-(x) = \frac{\phi_-(x)}{\phi(G)} = \sum_{y \in O(x)} \sigma(y, x). \tag{71}$$

Obviously for any internal node x, we have $\sigma_+(X) = \sigma_- = \sigma(x)$, where $\sigma(x)$ is a *normalized troughflow* of x.

Moreover, let

$$\sigma_+(G) = \frac{\phi_+(G)}{\phi(G)} = \sum_{x \in I(G)} \sigma_-(x), \tag{72}$$

$$\sigma_-(G) = \frac{\phi_-(G)}{\phi(G)} = \sum_{x \in O(G)} \sigma_+(x). \tag{73}$$

Obviously, $\sigma_+(G) = \sigma_-(G) = \sigma(G) = 1$.

6.3   Certainty and Coverage Factors

With every branch (x, y) of a flow graph G we associate the *certainty* and the *coverage factors*.

The *certainty* and the *coverage* of are defined as

$$cer(x, y) = \frac{\sigma(x, y)}{\sigma(x)}, \tag{74}$$

and

$$cov(x, y) = \frac{\sigma(x, y)}{\sigma(y)}. \tag{75}$$

respectively, where $\sigma(x) \neq 0$ and $\sigma(y) \neq 0$. Below some properties, which are immediate consequences of definitions given above are presented:

$$\sum_{y \in O(x)} cer(x, y) = 1, \tag{76}$$

$$\sum_{y \in I(y)} cov(x, y) = 1, \tag{77}$$

$$\sigma(x) = \sum_{y \in O(x)} cer(x, y)\sigma(x) = \sum_{y \in O(x)} \sigma(x, y), \tag{78}$$

$$\sigma(y) = \sum_{x \in I(y)} cov(x, y)\sigma(y) = \sum_{x \in I(y)} \sigma(x, y), \tag{79}$$

$$cer(x, y) = \frac{cov(x, y)\sigma(y)}{\sigma(x)}, \tag{80}$$

$$cov(x, y) = \frac{cer(x, y)\sigma(x)}{\sigma(y)}. \tag{81}$$

Obviously the above properties have a probabilistic flavor, e.g., equations (78) and (79) have a form of total probability theorem, whereas formulas (80) and (81) are Bayes' rules. However, these properties in our approach are interpreted in a deterministic way and they describe flow distribution among branches in the network.

A (*directed*) *path* from x to y, $x \neq y$ in G is a sequence of nodes $x_1, \ldots, x_n$ such that $x_1 = x$, $x_n = y$ and $(x_i, x_{i+1}) \in B$ for every $i, 1 \leq i \leq n - 1$. A path from x to y is denoted by $[x \ldots y]$.

The *certainty*, the *coverage* and the *strength* of the path $[x_1 \ldots x_n]$ are defined as

$$cer[x_1 \ldots x_n] = \prod_{i=1}^{n-1} cer(x_i, x_{i+1}), \tag{82}$$

$$cov[x_1 \ldots x_n] = \prod_{i=1}^{n-1} cov(x_i, x_{i+1}), \tag{83}$$

$$\sigma[x \ldots y] = \sigma(x)cer[x \ldots y] = \sigma(y)cov[x \ldots y], \tag{84}$$

respectively. The set of all paths from x to $y(x \neq y)$ in G denoted $< x, y >$, will be called a *connection* from x to y in G. In other words, connection $< x, y >$ is a sub-graph of G determined by nodes x and y.

For every connection $< x, y >$ we define its certainty, coverage and strength as shown below:

$$cer < x, y >= \sum_{[x...y] \in < x,y >} cer[x...y], \qquad (85)$$

the *coverage* of the connection $< x, y >$ is

$$cov < x, y >= \sum_{[x...y] \in < x,y >} cov[x...y], \qquad (86)$$

and the *strength* of the connection $< x, y >$ is

$$\sigma < x, y >= \sum_{[x...y] \in < x,y >} \sigma[x...y] = \sigma(x)cer < x, y >= \sigma(y)cov < x, y > . \qquad (87)$$

Let $[x...y]$ be a path such that x and y are input and output of the graph G, respectively. Such a *path* will be referred to as *complete*.

The set of all complete paths from x to y will be called a *complete connection* from x to y in G. In what follows we will consider complete paths and connections only, unless stated otherwise.

Let x and y be an input and output of a graph G respectively. If we substitute for every complete connection $< x, y >$ in G a single branch (x, y) such $\sigma(x, y) = \sigma < x, y >$, $cer(x, y) = cer < x, y >$, $cov(x, y) = cov < x, y >$ then we obtain a new flow graph G' such that $\sigma(G) = \sigma(G')$. The new flow graph will be called a *combined* flow graph. The combined flow graph for a given flow graph represents a relationship between its inputs and outputs.

6.4   Dependencies in Flow Graphs

Let $(x, y) \in \mathcal{B}$. Nodes x and y are independent on each other if

$$\sigma(x, y) = \sigma(x)\sigma(y). \qquad (88)$$

Consequently

$$\frac{\sigma(x, y)}{\sigma(x)} = cer(x, y) = \sigma(y), \qquad (89)$$

and

$$\frac{\sigma(x, y)}{\sigma(y)} = cov(x, y) = \sigma(x). \qquad (90)$$

This idea refers to some concepts proposed by Łukasiewicz [88] in connection with statistical independence of logical formulas.

If

$$cer(x, y) > \sigma(y), \tag{91}$$

or

$$cov(x, y) > \sigma(x), \tag{92}$$

then x and y *depend positively* on each other. Similarly, if

$$cer(x, y) < \sigma(y), \tag{93}$$

or

$$cov(x, y) < \sigma(x), \tag{94}$$

then x and y *depend negatively* on each other.

Let us observe that relations of independency and dependencies are symmetric ones, and are analogous to that used in statistics.

For every $(x, y) \in \mathcal{B}$ we define a *dependency factor* $\eta(x, y)$ defined as

$$\eta(x, y) = \frac{cer(x, y) - \sigma(y)}{cer(x, y) + \sigma(y)} = \frac{cov(x, y) - \sigma(x)}{cov(x, y) + \sigma(x)}. \tag{95}$$

It is easy to check that if $\eta(x, y) = 0$, then x and y are independent on each other, if $-1 < \eta(x, y) < 0$, then x and y are negatively dependent and if $0 < \eta(x, y) < 1$ then x and y are positively dependent on each other.

Thus the dependency factor expresses a degree of dependency, and can be seen as a counterpart of correlation coefficient used in statistics (see also [112]).

6.5   An Example

Now we will illustrate ideas introduced in the previous sections by means of a simple example concerning votes distribution of various age groups and social classes of voters between political parties.

Consider three disjoint age groups of voters y_1 (*old*), y_2 (*middle aged*) and y_3 (*young*) – belonging to three social classes x_1 (*high*), x_2 (*middle*) and x_3 (*low*). The voters voted for four political parties z_1 (*Conservatives*), z_2 (*Labor*), z_3 (*Liberal Democrats*) and z_4 (*others*).

Social class and age group votes distribution is shown in Figure 4.

First we want to find votes distribution with respect to age group. The result is shown in Figure 5. From the flow graph presented in Figure 5 we can see that, e.g., party z_1 obtained 19% of total votes, all of them from age group y_1; party z_2 – 44% votes, which 82% are from age group y_2 and 18% – from age group y_3, etc.

If we want to know how votes are distributed between parties with respects to social classes we have to eliminate age groups from the flow graph. Employing the algorithm presented in Section 6.3 we get results shown in Figure 6.
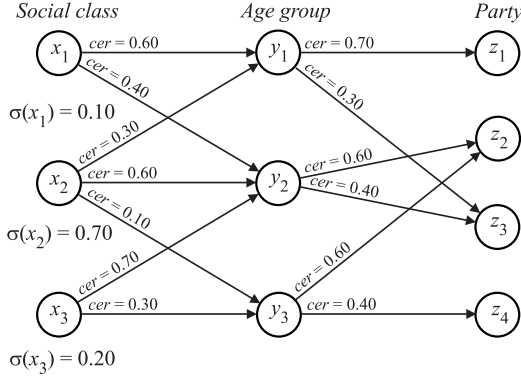
Fig. 4. Social class and age group votes distribution

From the flow graph presented in Figure 6 we can see that party z_1 obtained 22% votes from social class x_1 and 78% – from social class x_2, etc.

We can also present the obtained results employing decision rules. For simplicity we present only some decision rules of the decision algorithm. For example, from Figure 5 we obtain decision rules:

If Party (z_1) then Age group $(y_1)(0.19)$;
If Party (z_2) then Age group $(y_2(0.36)$;
If Party (z_2) then Age group $(y_3)(0.08)$, etc.

The number at the end of each decision rule denotes strength of the rule. Similarly, from Figure 6 we get:

If Party (z_1) then Soc. class $(x_1)(0.04)$;
If Party (z_1) then Soc. class $(x_2)(0.14)$, etc.
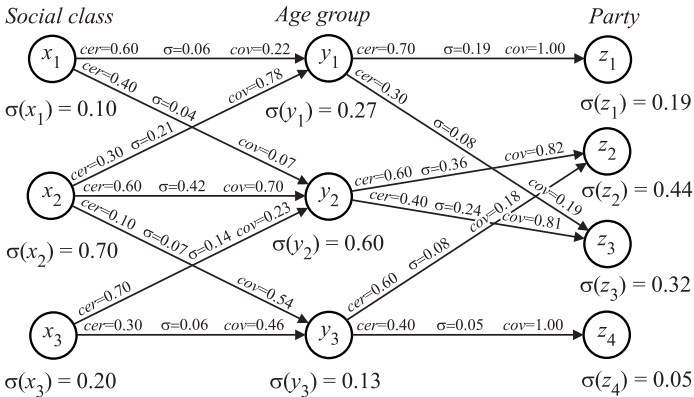
Fig. 5. Votes distribution with respect to the age group
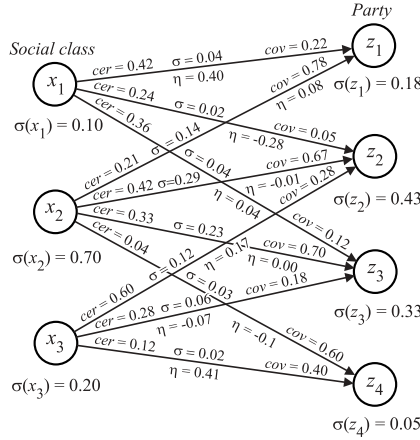
Fig. 6. Votes distribution between parties with respects to the social classes

From Figure 6 we have:

If Soc. class (x_1) then Party $(z_1)(0.04)$;
If Soc. class (x_1) then Party $(z_2)(0.02)$;
If Soc. class (x_1) then Party $(z_3)(0.04)$, etc.

Dependencies between Social class and Parties are shown in Figure 6.

6.6   An Example

In this section we continue the example from Section 5.3. The flow graph associated with Table 11 is shown in Figure 7.

Branches of the flow graph represent decision rules together with their certainty and coverage factors. For example, the decision rule $A \rightarrow 0$ has the certainty and coverage factors 0.13 and 0.35, respectively. The flow graph gives a clear insight into the voting structure of all parties. For many applications exact values of certainty of coverage factors of decision rules are not necessary. To this end we introduce "approximate" decision rules, denoted $C \rightsquigarrow D$ and read C *mostly implies* D. $C \rightsquigarrow D$ if and only if $cer(C, D) > 0.5$. Thus, we can replace flow graph shown in Figure 7 by *approximate* flow graph presented in Figure 8. From this graph we can see that parties B, C and D form a coalition, which is in conflict with party A, i.e., every member of the coalition is in conflict with party A. The corresponding conflict graph is shown in Figure 9.

Moreover, from the flow graph shown in Figure 7 we can obtain an "inverse" approximate flow graph which is shown in Figure 10. This flow graph contains all inverse decision rules with certainty factor greater than 0.5. From this graph we can see that yes votes were obtained *mostly* from party A and *no* votes – *mostly* from party D.
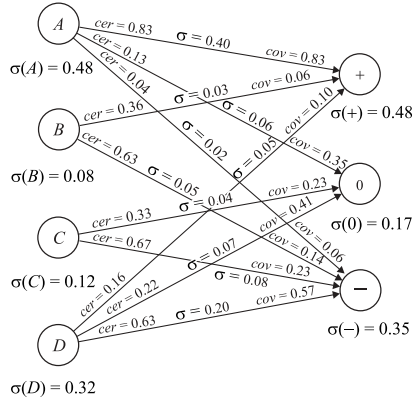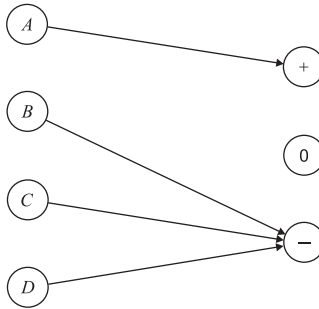
Fig. 7. Flow graph for Table 11
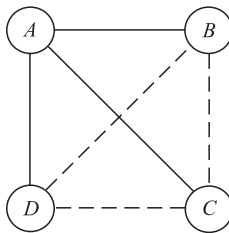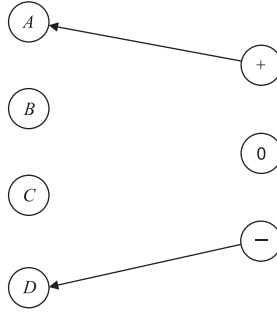

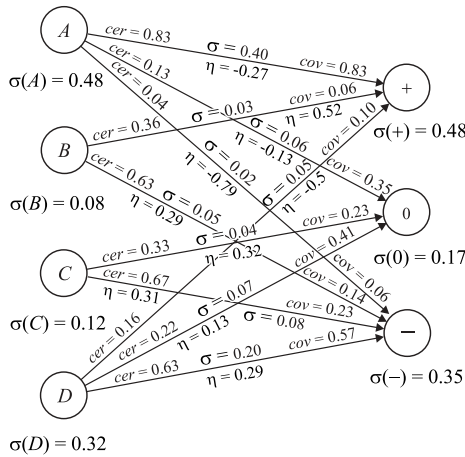**Fig. 8.** "Approximate" flow graph

Fig. 9. Conflict graph

We can also compute dependencies between parties and voting results the results are shown in Figure 11.

6.7   Decision Networks

Ideas given in the previous sections can be also presented in logical terms, as shown in what follows.

Fig. 10. An "inverse" approximate flow graph

Fig. 11. Dependencies between parties and voting results

The main problem in data mining consists in discovering patterns in data. The patterns are usually expressed in form of decision rules, which are logical expressions in the form *if Φ then Ψ*, where *Φ* and *Ψ* are logical formulas (propositional functions) used to express properties of objects of interest. Any set of decision rules is called a *decision algorithm*. Thus knowledge discovery from data consists in representing hidden relationships between data in a form of decision algorithms. However, for some applications, it is not enough to give only set of decision rules describing relationships in the database. Sometimes also knowledge of relationship between decision rules is necessary in order to understand better data structures. To this end we propose to employ a decision algorithm in which also relationship between decision rules is pointed out, called a *decision network*.

The decision network is a finite, directed acyclic graph, nodes of which represent logical formulas, whereas branches – are interpreted as decision rules. Thus

every path in the graph represents a chain of decisions rules, which will be used to describe compound decisions.

Some properties of decision networks will be given and a simple example will be used to illustrate the presented ideas and show possible applications.

Let U be a non empty finite set, called the *universe* and let Φ, Ψ be logical formulas. The meaning of Φ in U, denoted by $\|\Phi\|$, is the set of all elements of U, that satisfies Φ in U. The truth value of Φ denoted $val(\Phi)$ is defined as $card(\|\Phi\|)/card(U)$, where $card(X)$ denotes cardinality of X and $\mathcal{F}$ is a set of formulas.

By *decision network* over $S = (U, \mathcal{F})$ we mean a pair $N = (\mathcal{F}, \mathcal{R})$, where $\mathcal{R} \subseteq \mathcal{F} \times \mathcal{F}$ is a binary relation, called a *consequence relation* and $\mathcal{F}$ is a set of logical formulas.

Any pair $(\Phi, \Psi) \in \mathcal{R}, \Phi \neq \Psi$ is referred to as a decision rule (in N).

We assume that S is known and we will not refer to it in what follows.

A *decision rule* (Φ, Ψ) will be also presented as an expression $\Phi \rightarrow \Psi$, read *if Φ then Ψ*, where Φ and Ψ are referred to as *predecessor* (*conditions*) and *successor* (*decisions*) of the rule, respectively.

The number $supp(\Phi, \Psi) = card(\|\Phi \wedge \Psi\|)$ will be called a *support* of the rule $\Phi \rightarrow \Psi$. We will consider nonvoid decision rules only, i.e., rules such that $supp(\Phi, \Psi) \neq 0$.

With every decision rule $\Phi \rightarrow \Psi$ we associate its *strength* defined as

$$str(\Phi, \Psi) = \frac{supp(\Phi, \Psi)}{card(U)}. \tag{96}$$

Moreover, with every decision rule $\Phi \rightarrow \Psi$ we associate the *certainty factor* defined as

$$cer(\Phi, \Psi) = \frac{str(\Phi, \Psi)}{val(\Phi)}, \tag{97}$$

and the *coverage factor* of $\Phi \rightarrow \Psi$

$$cov(\Phi, \Psi) = \frac{str(\Phi, \Psi)}{val(\Psi)}, \tag{98}$$

where $val(\Phi) \neq 0$ and $val(\Psi) \neq 0$.

The coefficients can be computed from data or can be a subjective assessment.

We assume that

$$val(\Phi) = \sum_{\Psi \in Suc(\Phi)} str(\Phi, \Psi) \tag{99}$$

and

$$val(\Psi) = \sum_{\Phi \in Pre(\Psi)} str(\Phi, \Psi), \tag{100}$$

where $Suc(\Phi)$ and $Pre(\Psi)$ are sets of all successors and predecessors of the corresponding formulas, respectively.

Consequently we have

$$\sum_{Suc(\Phi)} cer(\phi, \Psi) = \sum_{Pre(\Psi)} cov(\Phi, \Psi) = 1. \tag{101}$$

If a decision rule $\Phi \rightarrow \Psi$ uniquely determines decisions in terms of conditions, i.e., if $cer(\Phi, \Psi) = 1$, then the rule is *certain*, otherwise the rule is *uncertain*.

If a decision rule $\Phi \rightarrow \Psi$ covers all decisions, i.e., if $cov(\Phi, \Psi) = 1$ then the decision rule is *total*, otherwise the decision rule is *partial*.

Immediate consequences of (97) and (98) are:

$$cer(\Phi, \Psi) = \frac{cov(\Phi, \Psi)val(\Psi)}{val(\Phi)}, \tag{102}$$

$$cov(\Phi, \Psi) = \frac{cer(\Phi, \Psi)val(\Phi)}{val(\Psi)}. \tag{103}$$

Note, that (102) and (103) are Bayes' formulas. This relationship, as mentioned previously, first was observed by Lukasiewicz [88].

Any sequence of formulas $\Phi_1, \ldots, \Phi_n$, $\Phi_i \in \mathcal{F}$ and for every i, $1 \leq i \leq n-1$, $(\Phi_i, \Phi_{i+1}) \in \mathcal{R}$ will be called a *path* from Φ_1 to Φ_n and will be denoted by $[\Phi_1 \ldots \Phi_n]$.

We define

$$cer[\Phi_1 \ldots \Phi_n] = \prod_{i=1}^{n-1} cer[\Phi_i, \Phi_{i+1}], \tag{104}$$

$$cov[\Phi_1 \ldots \Phi_n] = \prod_{i=1}^{n-1} cov[\Phi_i, \Phi_{i+1}], \tag{105}$$

$$str[\Phi_1 \ldots \Phi_n] = val(\Phi_1)cer[\Phi_1 \ldots \Phi_n] = val(\Phi_n)cov[\Phi_1 \ldots \Phi_n]. \tag{106}$$

The set of all paths form Φ to Ψ, denoted $< \Phi, \Psi >$, will be called a *connection* from Φ to Ψ.

For connection we have

$$cer < \Phi, \Psi >= \sum_{[\Phi \ldots \Psi] \in < \Phi, \Psi >} cer[\Phi \ldots \Psi], \tag{107}$$

$$cov < \Phi, \Psi >= \sum_{[\Phi \ldots \Psi] \in < \Phi, \Psi >} cov[\Phi \ldots \Psi], \tag{108}$$

$$str < \Phi, \Psi > = \sum_{[\Phi \ldots \Psi] \in < \Phi, \Psi >} str[\Phi \ldots \Psi] =$$
$$= val(\Phi)cer < \Phi, \Psi >= val(\Psi)cov < \Phi, \Psi > . \tag{109}$$

With every decision network we can associate a flow graph [70, 71]. Formulas of the network are interpreted as nodes of the graph, and decision rules – as directed branches of the flow graph, whereas strength of a decision rule is interpreted as flow of the corresponding branch.

Let $\Phi \rightarrow \Psi$ be a decision rule. Formulas Φ and Ψ are *independent* on each other if

$$str(\Phi, \Psi) = val(\Phi)val(\Psi). \tag{110}$$

Consequently

$$\frac{str(\Phi, \Psi)}{val(\Phi)} = cer(\Phi, \Psi) = val(\Psi), \tag{111}$$

and

$$\frac{str(\Phi, \Psi)}{val(\Psi)} = cov(\Phi, \Psi) = val(\Phi). \tag{112}$$

If

$$cer(\Phi, \Psi) > val(\Psi), \tag{113}$$

or

$$cov(\Phi, \Psi) > val(\Phi), \tag{114}$$

then Φ and Ψ *depend positively* on each other. Similarly, if

$$cer(\Phi, \Psi) < val(\Psi), \tag{115}$$

or

$$cov(\Phi, \Psi) < val(\Phi), \tag{116}$$

then Φ and Ψ *depend negatively* on each other.

For every decision rule $\Phi \rightarrow \Psi$ we define a *dependency factor* $\eta(\Phi, \Psi)$ defined as

$$\eta(\Phi, \Psi) = \frac{cer(\Phi, \Psi) - val(\Psi)}{cer(\Phi, \Psi) + val(\Psi)} = \frac{cov(\Phi, \Psi) - val(\Phi)}{cov(\Phi, \Psi) + val(\Phi)}. \tag{117}$$

It is easy to check that if $\eta(\Phi, \Psi) = 0$, then Φ and Ψ are independent on each other, if $-1 < \eta(\Phi, \Psi) < 0$, then Φ and Ψ are negatively dependent and if $0 < \eta(\Phi, \Psi) < 1$ then Φ and Ψ are positively dependent on each other.

6.8   An Example

Flow graphs given in Figures 4–6 can be now presented as shown in Figures 12–14, respectively. These flow graphs show clearly the relational structure between formulas involved in the voting process.
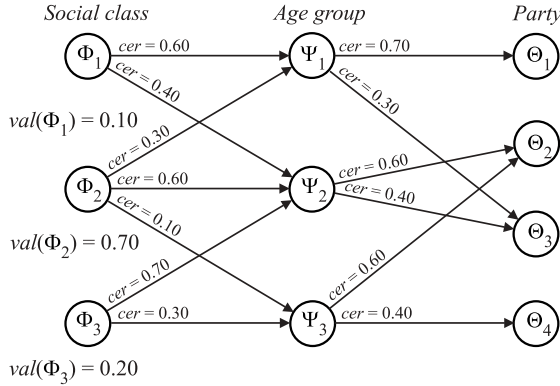
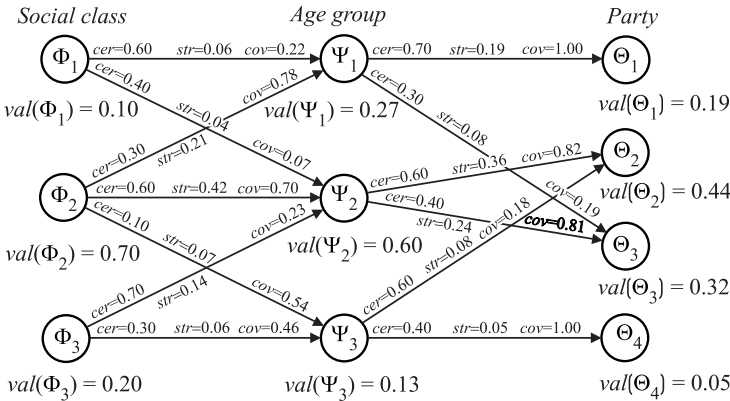**Fig. 12.** Decision network for flow graph from Figure 4

Fig. 13. Decision network for flow graph from Figure 5

6.9   Inference Rules and Decision Rules

In this section we are going to show relationship between previously discussed concepts and reasoning schemes used in logical inference.

Basic rules of inference used in classical logic are *Modus Ponens* (MP) and *Modus Tollens* (MT). These two reasoning patterns start from some general knowledge about reality, expressed by true implication, "*if Φ then Ψ*". Then basing on true *premise Φ* we arrive at true *conclusion Ψ* (MP), or if negation of conclusion Ψ is true we infer that negation of premise Φ is true (MT).

In reasoning from data (data mining) we also use rules *if Φ then Ψ*, called *decision rules*, to express our knowledge about reality, but the meaning of decision rules is different. It does not express general knowledge but refers to partial facts. Therefore decision rules are not true or false but probable (possible) only.
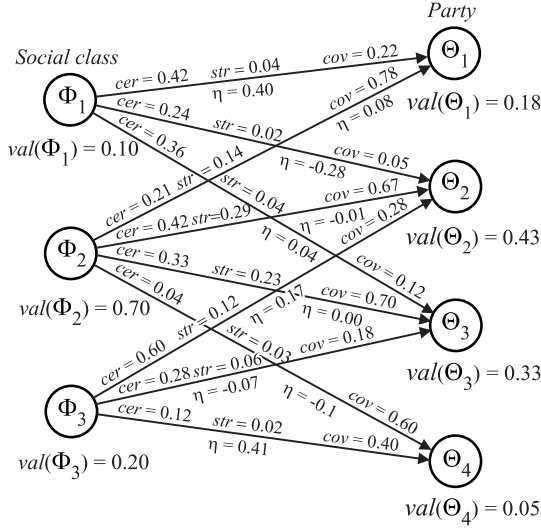
Fig. 14. Decision network for flow graph from Figure 6

In this paper we compare inference rules and decision rules in the context of decision networks, proposed by the author as a new approach to analyze reasoning patterns in data.

Decision network is a set of logical formulas $\mathcal{F}$ together with a binary relation over the set $\mathcal{R} \subseteq \mathcal{F} \times \mathcal{F}$ of formulas, called a *consequence relation*. Elements of the relation are called *decision rules*. The decision network can be perceived as a directed graph, nodes of which are formulas and branches – are decision rules. Thus the decision network can be seen as a knowledge representation system, revealing data structure of a data base.

Discovering patterns in the database represented by a decision network boils down to discovering some patterns in the network. Analogy to the *modus ponens* and *modus tollens* inference rules will be shown and discussed.

Classical rules of inference used in logic are *Modus Ponens* and *Modus Tollens*, which have the form

if	$\Phi \to \Psi$	is true
and	Φ	is true
then	Ψ	is true

and

if	$\Phi \to \Psi$	is true
and	$\sim \Psi$	is true
then $\sim \Phi$		is true

respectively.

Modus Ponens allows us to obtain true consequences from true premises, whereas *Modus Tollens* yields true negation of premise from true negation of conclusion.

In reasoning about data (data analysis) the situation is different. Instead of true propositions we consider propositional functions, which are true to a "degree", i.e., they assume truth values which lie between 0 and 1, in other words, they are probable, not true.

Besides, instead of true inference rules we have now decision rules, which are neither true nor false. They are characterized by three coefficients, *strength, certainty* and *coverage factors*. Strength of a decision rule can be understood as a counterpart of truth value of the inference rule, and it represents frequency of the decision rule in a database.

Thus employing decision rules to discovering patterns in data boils down to computation probability of conclusion in terms of probability of the premise and strength of the decision rule, or – the probability of the premise from the probability of the conclusion and strength of the decision rule.

Hence, the role of decision rules in data analysis is somehow similar to classical inference patterns, as shown by the schemes below.

Two basic rules of inference for data analysis are as follows:

if	$\Phi \rightarrow \Psi$	*has* $cer(\Phi, \Psi)$ *and* $cov(\Phi, \Psi)$
and	Φ	*is true with the probability* $val(\Phi)$
then	Ψ	*is true with the probability* $val(\Psi) = \alpha val(\Phi)$.

Similarly

if	$\Phi \rightarrow \Psi$	*has* $cer(\Phi, \Psi)$ *and* $cov(\Phi, \Psi)$
and	Ψ	*is true with the probability* $val(\Psi)$
then	Φ	*is true with the probability* $val(\Phi) = \alpha^{-1} val(\Phi)$.

The above inference rules can be considered as counterparts of *Modus Ponens* and *Modus Tollens* for data analysis and will be called Rough *Modus Ponens* (RMP) and Rough *Modus Tollens* (RMT), respectively.

There are however essential differences between MP (MT) and RMP (RMT).

First, instead of truth values associated with inference rules we consider certainly and coverage factors (conditional probabilities) assigned to decision rules.

Second, in the case of decision rules, in contrast to inference rules, truth value of a conclusion (RMP) depends not only on a single premise but in fact depends on truth values of premises of all decision rules having the same conclusions. Similarly, for RMT.

Let us also notice that inference rules are transitive, i.e., *if* $\Phi \rightarrow \Psi$ *and* $\Psi \rightarrow \Theta$ *then* $\Phi \rightarrow \Theta$ and decision rules are not. *If* $\Phi \rightarrow \Psi$ *and* $\Psi \rightarrow \Theta$, then we have to compute the certainty, coverage and strength of the rule $\Phi \rightarrow \Theta$, employing formulas (104),(105),(107),(108).

This shows clearly the difference between reasoning patterns using classical inference rules in logical reasoning and using decision rules in reasoning about data.

6.10   An Example

Suppose that three models of cars Φ_1, Φ_2 and Φ_3 are sold to three disjoint groups of customers Θ_1, Θ_2 and Θ_3 through four dealers Ψ_1, Ψ_2, Ψ_3 and Ψ_4.

Moreover, let us assume that car models and dealers are distributed as shown in Figure 15. Applying RMP to data shown in Figure 15 we get results shown in Figure 16. In order to find how car models are distributed among customer
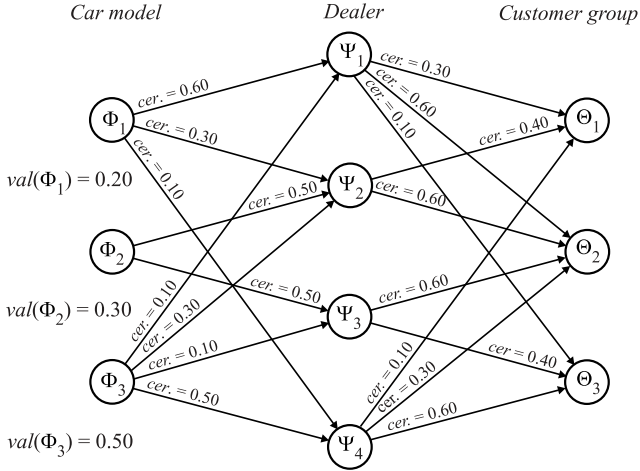
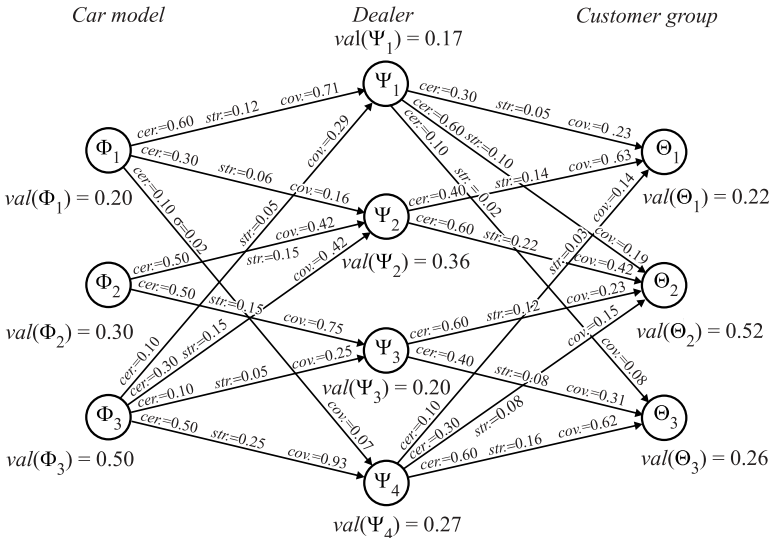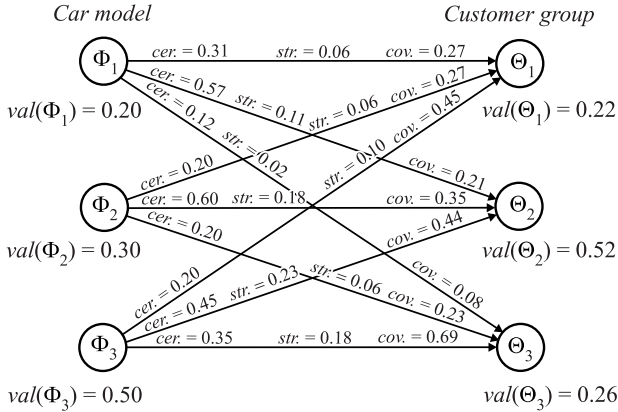**Fig. 15.** Distributions of car models and dealers

Fig. 16. The result of application of RMP to data from Figure 15

Fig. 17. Distribution of car models among customer groups

groups we have to compute all connections among cars models and consumers groups, i.e., to apply RMP to data given in Figure 16. The results are shown in Figure 17.

For example, we can see from the decision network that consumer group Θ_2 bought 21% of car model Φ_1, 35% of car model Φ_2 and 44% of car model Φ_3. Conversely, for example, car model Φ_1 is distributed among customer groups as follows: 31% cars bought group Θ_1, 57% group Θ_2 and 12% group Θ_3.

7   Summary

Basic concept of mathematics, the set, leads to antinomies, i.e., it is contradictory. This deficiency of sets, has rather philosophical than practical meaning, for sets used in mathematics are free from the above discussed faults. Antinomies are associated with very "artificial" sets constructed in logic but not found in sets used in mathematics. That is why we can use mathematics safely.

Philosophically, fuzzy set theory and rough set theory are two different approaches to vagueness and are not remedy for classical set theory difficulties. Both theories represent two different approaches to vagueness. Fuzzy set theory addresses gradualness of knowledge, expressed by the fuzzy membership whereas rough set theory addresses granularity of knowledge, expressed by the indiscernibility relation.

Practically, rough set theory can be viewed as a new method of intelligent data analysis. Rough set theory has found many applications in medical data analysis, finance, voice recognition, image processing, and others. However the approach presented in this paper is too simple to many real-life applications and was extended in many ways by various authors. The detailed discussion of the above issues can be found in be found in books (see, e.g., [18–27, 12, 28–30]), special issues of journals (see, e.g., [31–34, 34–38]), proceedings of international conferences (see, e.g., [39–49]), tutorials (e.g., [50–53]), and on the internet (see, e.g., www.roughsets.org, logic.mimuw.edu.pl,rsds.wsiz.rzeszow.pl).

Besides, rough set theory inspired new look on Bayes' theorem. Bayesian inference consists in update prior probabilities by means of data to posterior probabilities. In the rough set approach Bayes' theorem reveals data patterns, which are used next to draw conclusions from data, in form of decision rules.

Moreover, we have shown a new mathematical model of flow networks, which can be used to decision algorithm analysis. In particular it has been revealed that the flow in the flow network is governed by Bayes' rule, which has entirely deterministic meaning, and can be used to decision algorithm study.

Also, a new look of dependencies in databases, based on Łukasiewiczs ideas of independencies of logical formulas, is presented.

Acknowledment

References

1. Zadeh, L.A.: Fuzzy sets. Information and Control **8** (1965) 338–353
2. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences **11** (1982) 341–356
3. Ziarko, W.: Variable precision rough set model. Journal of Computer and System Sciences **46** (1993) 39–59
4. Polkowski, L., Skowron, A., Żytkow, J.: Rough foundations for rough sets. In [40] 55–58
5. Skowron, A., Stepaniuk, J.: Tolerance approximation spaces. Fundamenta Informaticae **27** (1996) 245–253
6. Polkowski, L., Skowron, A.: Rough mereology: A new paradigm for approximate reasoning. International Journal of Approximate Reasoning **15** (1996) 333–365
7. Słowiński, R., Vanderpooten, D.: Similarity relation as a basis for rough approximations. In Wang, P.P., ed.: Machine Intelligence & Soft-Computing, Vol. IV. Bookwrights, Raleigh, NC (1997) 17–33
8. Słowiński, R., Vanderpooten, D.: A generalized definition of rough approximations based on similarity. IEEE Transactions on Data and Knowledge Engineering **12(2)** (2000) 331–336
9. Stepaniuk, J.: Knowledge discovery by application of rough set models. In [26] 137–233
10. Skowron, A.: Toward intelligent systems: Calculi of information granules. Bulletin of the International Rough Set Society **5** (2001) 9–30
11. Greco, A., Matarazzo, B., Słowiński, R.: Rough approximation by dominance relations. International Journal of Intelligent Systems **17** (2002) 153–171
12. Polkowski, L., ed.: Rough Sets: Mathematical Foundations. Advances in Soft Computing. Physica-Verlag, Heidelberg (2002)
13. Skowron, A., Stepaniuk, J.: Information granules and rough-neural computing. In [30] 43–84
14. Skowron, A.: Approximation spaces in rough neurocomputing. In [29] 13–22
15. Wróblewski, J.: Adaptive aspects of combining approximation spaces. In [30] 139–156

16. Yao, Y.Y.: Informaton granulation and approximation in a decision-theoretical model of rough sets. In [30] 491–520

17. Skowron, A., Swiniarski, R., Synak, P.: Approximation spaces and information granulation (submitted). In: Fourth International Conference on Rough Sets and Current Trends in Computing (RSCTC'04), Uppsala, Sweden, June 1-5, 2004. Lecture Notes in Computer Science. Springer-Verlag, Heidelberg, Germany (2004)

18. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Volume 9 of System Theory, Knowledge Engineering and Problem Solving. Kluwer Academic Publishers, Dordrecht, The Netherlands (1991)

19. Słowiński, R., ed.: Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory. Volume 11 of System Theory, Knowledge Engineering and Problem Solving. Kluwer Academic Publishers, Dordrecht, The Netherlands (1992)

20. Lin, T.Y., Cercone, N., eds.: Rough Sets and Data Mining - Analysis of Imperfect Data. Kluwer Academic Publishers, Boston, USA (1997)

21. Orłowska, E., ed.: Incomplete Information: Rough Set Analysis. Volume 13 of Studies in Fuzziness and Soft Computing. Springer-Verlag/Physica-Verlag, Heidelberg, Germany (1997)

22. Polkowski, L., Skowron, A., eds.: Rough Sets in Knowledge Discovery 1: Methodology and Applications. Volume 18 of Studies in Fuzziness and Soft Computing. Physica-Verlag, Heidelberg, Germany (1998)

23. Polkowski, L., Skowron, A., eds.: Rough Sets in Knowledge Discovery 2: Applications, Case Studies and Software Systems. Volume 19 of Studies in Fuzziness and Soft Computing. Physica-Verlag, Heidelberg, Germany (1998)

24. Pal, S.K., Skowron, A., eds.: Rough Fuzzy Hybridization: A New Trend in Decision-Making. Springer-Verlag, Singapore (1999)

25. Duentsch, I., Gediga, G.: Rough set data analysis: A road to non-invasive knowledge discovery. Methodos Publishers, Bangor, UK (2000)

26. Polkowski, L., Lin, T.Y., Tsumoto, S., eds.: Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems. Volume 56 of Studies in Fuzziness and Soft Computing. Springer-Verlag/Physica-Verlag, Heidelberg, Germany (2000)

27. Lin, T.Y., Yao, Y.Y., Zadeh, L.A., eds.: Rough Sets, Granular Computing and Data Mining. Studies in Fuzziness and Soft Computing. Physica-Verlag, Heidelberg (2001)

28. Demri, S., Orłowska, E., eds.: Incomplete Information: Structure, Inference, Complexity. Monographs in Theoretical Cpmputer Sience. Springer-Verlag, Heidelberg, Germany (2002)

29. Inuiguchi, M., Hirano, S., Tsumoto, S., eds.: Rough Set Theory and Granular Computing. Volume 125 of Studies in Fuzziness and Soft Computing. Springer-Verlag, Heidelberg (2003)

30. Pal, S.K., Polkowski, L., Skowron, A., eds.: Rough-Neural Computing: Techniques for Computing with Words. Cognitive Technologies. Springer-Verlag, Heidelberg, Germany (2003)

31. Słowiński, R., Stefanowski, J., eds.: Special issue: Proceedings of the First International Workshop on Rough Sets: State of the Art and Perspectives, Kiekrz, Poznań, Poland, September 2–4 (1992). Volume 18(3-4) of Foundations of Computing and Decision Sciences. (1993)

32. Ziarko, W., ed.: Special issue. Volume 11(2) of Computational Intelligence: An International Journal. (1995)

33. Ziarko, W., ed.: Special issue. Volume 27(2-3) of Fundamenta Informaticae. (1996)

34. Lin, T.Y., ed.: Special issue. Volume 2(2) of Journal of the Intelligent Automation and Soft Computing. (1996)

35. Peters, J., Skowron, A., eds.: Special issue on a rough set approach to reasoning about data. Volume 16(1) of International Journal of Intelligent Systems. (2001)

36. Cercone, N., Skowron, A., Zhong, N., eds.: (Special issue). Volume 17(3) of Computational Intelligence. (2001)

37. Pal, S.K., Pedrycz, W., Skowron, A., Swiniarski, R., eds.: Special volume: Rough-neuro computing. Volume 36 of Neurocomputing. (2001)

38. Skowron, A., Pal, S.K., eds.: Special volume: Rough sets, pattern recognition and data mining. Volume 24(6) of Pattern Recognition Letters. (2003)

39. Ziarko, W., ed.: Rough Sets, Fuzzy Sets and Knowledge Discovery: Proceedings of the Second International Workshop on Rough Sets and Knowledge Discovery (RSKD'93), Banff, Alberta, Canada, October 12–15 (1993). Workshops in Computing. Springer–Verlag & British Computer Society, London, Berlin (1994)

40. Lin, T.Y., Wildberger, A.M., eds.: Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Knowledge Discovery. Simulation Councils, Inc., San Diego, CA, USA (1995)

41. Tsumoto, S., Kobayashi, S., Yokomori, T., Tanaka, H., Nakamura, A., eds.: Proceedings of the The Fourth Internal Workshop on Rough Sets, Fuzzy Sets and Machine Discovery, November 6-8, University of Tokyo , Japan. The University of Tokyo, Tokyo (1996)

42. Polkowski, L., Skowron, A., eds.: First International Conference on Rough Sets and Soft Computing (RSCTC'98), Warsaw, Poland, June 22-26, 1998. Volume 1424 of Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg (1998)

43. Zhong, N., Skowron, A., Ohsuga, S., eds.: Proceedings of the 7-th International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing (RSFDGrC'99), Yamaguchi, November 9-11, 1999. Volume 1711 of Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg (1999)

44. Ziarko, W., Yao, Y., eds.: Proceedings of the 2-nd International Conference on Rough Sets and Current Trends in Computing (RSCTC'2000), Banff, Canada, October 16-19, 2000. Volume 2005 of Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg (2001)

45. Hirano, S., Inuiguchi, M., Tsumoto, S., eds.: Proceedings of International Workshop on Rough Set Theory and Granular Computing (RSTGC-2001), Matsue, Shimane, Japan, May 20-22, 2001. Volume 5(1-2) of Bulletin of the International Rough Set Society. International Rough Set Society, Matsue, Shimane (2001)

46. Terano, T., Nishida, T., Namatame, A., Tsumoto, S., Ohsawa, Y., Washio, T., eds.: New Frontiers in Artificial Intelligence, Joint JSAI'01 Workshop Post-Proceedings. Volume 2253 of Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg (2001)

47. Alpigini, J.J., Peters, J.F., Skowron, A., Zhong, N., eds.: Third International Conference on Rough Sets and Current Trends in Computing (RSCTC'02), Malvern, PA, October 14-16, 2002. Volume 2475 of Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg (2002)

48. Skowron, A., Szczuka, M., eds.: Proceedings of the Workshop on Rough Sets in Knowledge Discovery and Soft Computing at ETAPS 2003 (RSKD'03), April 12-13, 2003. Volume 82(4) of Electronic Notes in Computer Science. Elsevier, Amsterdam, Netherlands (2003)

49. Wang, G., Liu, Q., Yao, Y., Skowron, A., eds.: Proceedings of the 9-th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'03), Chongqing, China, May 26-29, 2003. Volume 2639 of Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg (2003)

50. Komorowski, J., , Pawlak, Z., Polkowski, L., Skowron, A.: Rough sets: a tutorial. In [24] 3–98

51. Pawlak, Z., Polkowski, L., Skowron, A.: Rough sets and rough logic: A KDD perspective. In [26] 583–646

52. Skowron, A., Pawlak, Z., Komorowski, J., Polkowski, L.: A rough set perspective on data and knowledge. In Kloesgen, W., Żytkow, J., eds.: Handbook of KDD. Oxford University Press, Oxford (2002) 134–149

53. Pawlak, Z., Polkowski, L., Skowron, A.: Rough set theory. In Wah, B., ed.: Encyclopedia Of Computer Science and Engineering. Wiley, New York, USA (2004)

54. Cantor, G.: Grundlagen einer allgemeinen Mannigfaltigkeitslehre, Leipzig, Germany (1883)

55. Russell, B.: The Principles of Mathematics. George Allen & Unwin Ltd., London, Great Britain (1903)

56. Russell, B.: Vagueness. The Australasian Journal of Psychology and Philosophy **1** (1923) 84–92

57. Black, M.: Vagueness: An exercise in logical analysis. Philosophy of Science **4(4)** (1937) 427–455

58. Hempel, C.G.: Vagueness and logic. Philosophy of Science **6** (1939) 163–180

59. Fine, K.: Vagueness, truth and logic. Synthese **30** (1975) 265–300

60. Keefe, R., Smith, P.: Vagueness: A Reader. MIT Press, Cambridge, MA (1999)

61. Keefe, R.: Theories of Vagueness. Cambridge University Press, Cambridge, U.K. (2000)

62. Frege, G.: Grundgesetzen der Arithmetik, 2. Verlag von Herman Pohle, Jena, Germany (1903)

63. Read, S.: Thinking about Logic - An Introduction to Philosophy of Logic. Oxford University Press, Oxford (1995)

64. Leśniewski, S.: Grungzüge eines neuen systems der grundlagen der mathematik. Fundamenta Matematicae **14** (1929) 1–81

65. Pawlak, Z., Skowron, A.: Rough membership functions. In Yager, R., Fedrizzi, M., Kacprzyk, J., eds.: Advances in the Dempster-Shafer Theory of Evidence, New York, NY, John Wiley & Sons (1994) 251–271

66. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In [19] 331–362

67. Berthold, M., Hand, D.J.: Intelligent Data Analysis. An Introduction. Springer-Verlag, Berlin, Heidelberg, New York (1999)

68. Box, G.E.P., Tiao, G.C.: Bayesian Inference in Statistical Analysis. John Wiley and Sons, Inc., New York, Chichester, Brisbane, Toronto, Singapore (1992)

69. Pawlak, Z.: Rough sets and decision algorithms. In [44] 30–45

70. Pawlak, Z.: In pursuit of patterns in data reasoning from data – the rough set way. In [47] 1–9

71. Pawlak, Z.: Probability, truth and flow graphs. In [48] 1–9

72. Wong, S., Ziarko, W.: Algebraic versus probabilistic independence in decision theory. In Ras, Z.W., Zemankova, M., eds.: Proceedings of the ACM SIGART First International Symposium on Methodologies for Intelligent Systems Knoxville (ISMIS'86), Tennessee, USA, October 22-24, 1986. ACM SIGART, USA (1986) 207–212

73. Wong, S., Ziarko, W.: On learning and evaluation of decision rules in the context of rough sets. In Ras, Z.W., Zemankova, M., eds.: Proceedings of the ACM SIGART First International Symposium on Methodologies for Intelligent Systems Knoxville (ISMIS'86), Tennessee, USA, October 22-24, 1986. ACM SIGART, USA (1986) 308–324

74. Pawlak, Z., Wong, S.K.M., Ziarko, W.: Rough sets: Probabilistic versus deterministic approach. International Journal of Man-Machine Studies **29(1)** (1988) 81–95

75. Yamauchi, Y., Mukaidono, M.: Probabilistic inference and bayeasian theorem based on logical implication. In [43] 334–342

76. Intan, R., an Y. Y. Yao, M.M.: Generalization of rough sets with alpha-coverings of the universe induced by conditional probability relations. In [46] 311–315

77. Ślęzak, D.: Approximate decision reducts (in Polish). PhD thesis, Warsaw University, Warsaw, Poland (2002)

78. Ślęzak, D.: Approximate bayesian networks. In Bouchon-Meunier, B., Gutierrez-Rios, J., Magdalena, L., Yager, R., eds.: Technologies for Constructing Intelligent Systems 2: Tools. Volume 90 of Studies in Fuzziness and Soft Computing. Springer-Verlag, Heidelberg, Germany (2002) 313–326

79. Ślęzak, D., Wróblewski, J.: Approximate bayesian network classifiers. In [47] 365–372

80. Yao, Y.Y.: Information granulation and approximation. In [30] 491–516

81. Ślęzak, D.: Approximate markov boundaries and bayesian networks: Rough set approach. In [29] 109–121

82. Ślęzak, D., Ziarko, W.: Attribute reduction in the bayesian version of variable precision rough set model. In [48]

83. Ślęzak, D., Ziarko, W.: Variable precision bayesian rough set model. In [49] 312–315

84. Wong, S.K.M., Wu, D.: A common framework for rough sets, databases, and bayesian networks. In [49] 99–103

85. Ślęzak, D.: The rough bayesian model for distributed decision systems (submitted). In: Fourth International Conference on Rough Sets and Current Trends in Computing (RSCTC'04), Uppsala, Sweden, June 1-5, 2004. Lecture Notes in Computer Science. Springer-Verlag, Heidelberg, Germany (2004)

86. Swinburne, R.: Bayes Theorem. Volume 113 of Proceedings of the British Academy. Oxford University Press, Oxford, UK (2003)

87. Bernardo, J.M., Smith, A.F.M.: Bayesian Theory. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Chichester, New York, Brisbane, Toronto, Singapore (1994)

88. Łukasiewicz, J.: Die logischen grundlagen der wahrscheinilchkeitsrechnung, Kraków 1913. In Borkowski, L., ed.: Jan Łukasiewicz - Selected Works. North Holland Publishing Company, Amsterdam, London, Polish Scientific Publishers, Warsaw (1970)

89. Adams, E.W.: The Logic of Conditionals. An Application of Probability to Deductive Logic. D. Reidel Publishing Company, Dordrecht, Boston (1975)

90. Grzymała-Busse, J.W.: LERS - a system for learning from examples based on rough sets. In [19] 3–18

91. Skowron, A.: Boolean reasoning for decision rules generation. In Komorowski, J., Raś, Z.W., eds.: Seventh International Symposium for Methodologies for Intelligent Systems (ISMIS'93), Trondheim, Norway, June 15-18. Volume 689 of Lecture Notes in Artificial Intelligence., Heidelberg, Springer-Verlag (1993) 295–305

92. Pawlak, Z., Skowron, A.: A rough set approach for decision rules generation. In: Thirteenth International Joint Conference on Artificial Intelligence (IJCAI'93), Chambéry, France, Morgan Kaufmann (1993) 114–119

93. Shan, N., Ziarko, W.: An incremental learning algorithm for constructing decision rules. In Ziarko, W., ed.: Rough Sets, Fuzzy Sets and Knowledge Discovery, Berlin, Germany, Springer Verlag (1994) 326–334

94. Nguyen, H.S.: Discretization of Real Value Attributes, Boolean Reasoning Approach. PhD thesis, Warsaw University, Warsaw, Poland (1997)

95. Słowiński, R., Stefanowski, J.: Rough family – software implementation of the rough set theory. In [23] 581–586

96. Nguyen, H.S., Nguyen, S.H.: Pattern extraction from data. Fundamenta Informaticae **34** (1998) 129–144

97. Nguyen, H.S., Nguyen, S.H.: Discretization methods for data mining. In [22] 451–482

98. Skowron, A.: Rough sets in KDD - plenary talk. In Shi, Z., Faltings, B., Musen, M., eds.: 16-th World Computer Congress (IFIP'00): Proceedings of Conference on Intelligent Information Processing (IIP'00). Publishing House of Electronic Industry, Beijing (2002) 1–14

99. Bazan, J., Nguyen, H.S., Nguyen, S.H., Synak, P., Wróblewski, J.: Rough set algorithms in classification problems. In [26] 49–88

100. Grzymala-Busse, J.W., Shah, P.: A comparison of rule matching methods used in aq15 and lers. In: Proceedings of the Twelfth International Symposium on Methodologies for Intelligent Systems (ISMIS'00), Charlotte, NC, October 11-14, 2000. Volume 1932 of Lecture Nites in Artificial Intelligence., Berlin, Germany, Springer-Verlag (2000) 148–156

101. Grzymała-Busse, J., Hu, M.: A comparison of several approaches to missing attribute values in data mining. In [44] 340 – 347

102. Greco, S., Matarazzo, B., Słowiński, R., Stefanowski, J.: An algorithm for induction of decision rules consistent with dominance principle. In [44] 304–313

103. Skowron, A.: Rough sets and boolean reasoning. In Pedrycz, W., ed.: Granular Computing: an Emerging Paradigm. Volume 70 of Studies in Fuzziness and Soft Computing. Springer-Verlag/Physica-Verlag, Heidelberg, Germany (2001) 95–124

104. Greco, S., Matarazzo, B., Słowiński, R.: Rough sets theory for multicriteria decision analysis. European J. of Operational Research **129(1)** (2001) 1–47

105. Casti, J.L.: Alternate Realities: Mathematical Models of Nature and Man. John Wiley and Sons, Inc., New York, Chichester, Brisbane, Toronto, Singapore (1989)

106. Coombs, C.H., Avruin, G.S.: The Structure of Conflicts. Lawrence Erlbaum, London (1988)

107. Deja, R.: Conflict analysis, rough set methods and applications. In [26] 491–520

108. Maeda, Y., Senoo, K., Tanaka, H.: Interval density function in conflict analysis. In [43] 382–389

109. Nakamura, A.: Conflict logic with degrees. In [24] 136–150

110. Pawlak, Z.: An inquiry into anatomy of conflicts. Journal of Information Sciences **109** (1998) 65–68

111. Ford, L.R., Fulkerson, D.R.: Flows in Networks. Princeton University Press, Princeton, New Jersey (1973)

112. Słowiński, R., Greco, S.: A note on dependency factor. (2004) (manuscript).