2

Rough Sets In Data Analysis: Foundations and Applications

Lech Polkowski[1,2] and Piotr Artiemjew[2]

[1] Polish-Japanese Institute of Information Technology, Koszykowa 86, 02008 Warszawa, Poland
polkow@pjwstk.edu.pl

[2] Department of Mathematics and Computer Science, University of Warmia and Mazury, Żołnierska 14, Olsztyn, Poland
artem@matman.uwm.edu.pl

Summary. Rough sets is a paradigm introduced in order to deal with uncertainty due to ambiguity of classification caused by incompleteness of knowledge. The idea proposed by Z. Pawlak in 1982 goes back to classical idea of representing uncertain and/or inexact notions due to the founder of modern logic, Gottlob Frege: uncertain notions should possess around them a region of uncertainty consisting of objects that can be qualified with certainty neither into the notion nor to its complement. The central tool in realizing this idea in rough sets is the relation of uncertainty based on the classical notion of indiscernibility due to Gottfried W. Leibniz: objects are indiscernible when no operator applied to each of them yields distinct values.

In applications, knowledge comes in the form of data; those data in rough sets are organized into an information system: a pair of the form (U, A) where U is a set of objects and A is a set of attributes, each of them a mapping $a : U \rightarrow V_a$, the value set of a.

Each attribute a does produce the *a-indiscernibility relation* $IND(a) = \{(u, v) : a(u) = a(v)\}$. Each set of attributes B does induce the *B-indiscernibility relation* $IND(B) = \bigcap IND(a) : a \in B$. Objects u, v that are in the relation $IND(B)$ are *B-indiscernible.* Classes $[u]_B$ of the relation $IND(B)$ form *B–elementary granules of knowledge.*

Rough sets allow for establishing dependencies among groups of attributes: a group B depends functionally on group C when $IND(C) \subseteq IND(B)$: in that case values of attributes in B are functions of values of attributes in C.

An important case is when data are organized into a decision system: a triple (U, A, d) where d is a new attribute called the *decision.* The decision gives a classification of object due to an expert, an external oracle; establishing dependencies between groups B of attributes in A and the decision is one of tasks of rough set theory.

The language for expressing dependencies is the descriptor logic. A *descriptor* is a formula $(a = v)$ where $v \in V_a$, interpreted in the set U as $[a = v] = \{u : a(u) = v\}$. Descriptor formulas are obtained from descriptors by means of connectives $\vee, \wedge, \neg, \Rightarrow$

L. Polkowski and P. Artiemjew: *Rough Sets In Data Analysis: Foundations and Applications*, Studies in Computational Intelligence (SCI) **122**, 33–54 (2008)
www.springerlink.com

of propositional calculus; their semantics is: $[\alpha \vee \beta] = [\alpha] \cup [\beta]$, $[\alpha \wedge \beta] = [\alpha] \cap [\beta]$, $[\neg\alpha] = U \setminus [\alpha]$, $[\alpha \Rightarrow \beta] = [\neg\alpha] \cup [\beta]$.

In the language of descriptors, dependency between a group B of attributes and the decision is expressed as a decision rule: $\bigwedge_{a \in B}(a = v_a) \Rightarrow (d = v)$; a set of decision rules is a decision algorithm. There exist a number of algorithms for inducing decision rules.

Indiscernibility relations proved to be too rigid for classification, and the search has been in rough sets for more flexible similarity relations. Among them one class that is formally rooted in logic is the class of rough inclusions. They allow for forming granules of knowledge more robust than traditional ones. Algorithms based on them allow for a substantial knowledge reduction yet with good classification quality.

The problem that is often met in real data is the problem of missing values. Algorithms based on granulation of knowledge allow for solving this problem with a good quality of classification.

In this Chapter, we discuss:

- Basics of rough sets;
- Language of descriptors and decision rules;
- Algorithms for rule induction;
- Examples of classification on real data;
- Granulation of knowledge;
- Algorithms for rule induction based on granulation of knowledge;
- Examples of classification of real data;
- The problem of missing values in data.

2.1 Basics of rough sets

Introduced by Pawlak in [20], rough set theory is based on ideas that – although independently fused into a theory of knowledge – borrow some thoughts from Gottlob Frege, Gottfried Wilhelm Leibniz, Jan Łukasiewicz, Stanislaw Leśniewski, to mention a few names of importance.

Rough set approach rests on the assumption that knowledge is classification of entities into concepts (notions). To perform the classification task, entities should be described in a formalized symbolic language.

In case of the rough set theory, this language is the language of attributes and values. The formal framework for allowing this description is an information system, see Pawlak [21].

2.1.1 Information systems: formal rendering of knowledge

An information system is a pair (U, A), in which U is a set of *objects* and A is a set of *attributes*. Each attribute $a \in A$ is a mapping $a : U \rightarrow V_a$ from the universe U into the *value set* V_a of a. A variant of this notion is a basic in data mining notion of a *decision system*: it is a pair $(U, A \cup \{d\})$, where $d \notin A$ is the *decision*. In applications, decision d is the attribute whose value is set by an expert whereas attributes in A, called in this case *conditional attributes*, are selected and valued by the system user. Description of entities is done in the attribute–value language.

2.1.2 Attribute–value language. Indiscernibility

Attribute–value language is built from elementary formulas called *descriptors*; a descriptor is a formula of the form $(a = v)$, where $v \in V_a$. From descriptors, complex formulas are formed by means of connectives $\vee, \wedge, \neg, \Rightarrow$ of propositional calculus: if α, β are formulas then $\alpha \vee \beta$, $\alpha \wedge \beta$, $\neg\alpha$, $\alpha \Rightarrow \beta$ are formulas. These formulas and no other constitute the syntax of the *descriptor logic*.

Semantics of descriptor logic formulas is defined recursively: for a descriptor $(a = v)$, its meaning $[a = v]$ is defined as the set $\{u \in U : a(u) = v\}$. For complex formulas, one adopts the recursive procedure, given by the following identities:

- $[\alpha \vee \beta] = [\alpha] \cup [\beta]$.
- $[\alpha \wedge \beta] = [\alpha] \cap [\beta]$.
- $[\neg\alpha] = U \setminus [\alpha]$.
- $[\alpha \Rightarrow \beta] = [\neg\alpha] \cup [\beta]$.

Descriptor logic allows for coding of objects in the set U as sets of descriptors: for an object $u \in U$, the *information set* $Inf_A(u)$ is defined as the set $\{(a = a(u)) : a \in A\}$. It may happen that two objects, u and v, have the same information set: $Inf_A(u) = Inf_A(v)$; in this case, one says that u and v are *A–indiscernible*. This notion maybe relativized to any set $B \subseteq A$ of attributes: the *B–indiscernibility* relation is defined as $IND(B) = \{(u, v) : Inf_B(u) = Inf_B(v)\}$, where $Inf_B(u) = \{(a = a(u)) : a \in B\}$ is the information set of u restricted to the set B of attributes.

A more general notion of a *template* was proposed and studied in [18]: a template is a formula of the form $(a \in W_a)$, where $W_a \subseteq V_a$ is a set of values of the attribute a; the meaning $[a \in W_a]$ of the template $(a \in W_a)$ is the set $\{u \in U : a(u) \in W_a\}$. Templates can also (like descriptors) be combined by means of propositional connectives with semantics defined as with descriptors.

The indiscernibility relations are very important in rough sets: one easily may observe that for $u \in U$, and the formula in descriptor logic: $\phi_u^B : \bigwedge_{a \in B}(a = a(u))$, the meaning $[\phi_u^B]$ is equal to the equivalence class $[u]_B = \{v \in U : (u, v) \in IND(B)$ of the equivalence relation $IND(B)$.

The moral is: classes $[u]_B$ are *definable*, i.e., they have descriptions in the descriptor logic; also unions of those classes are definable: for a union $X = \bigcup_{j \in J}[u_j]_{B_j}$ of such classes, the formula $\bigvee_{j \in J} \phi_{u_j}^{B_j}$ has the meaning equal to X.

Concepts $X \subseteq U$ that are definable are also called *exact*; other concepts are called *rough*. The fundamental difference between the two kinds of concepts is that only exact concepts are "seen" in data; rough concepts are "blurred" and they can be described by means of exact concepts only; to this aim, rough sets offer the notion of an approximation.

2.1.3 Approximations

Due to Fregean idea [6], an inexact concept should possess a boundary into which objects that can be classified with certainty neither to the concept nor to its complement fall. This boundary to a concept is constructed from indiscernibility relations induced by attributes (features) of objects.

To express the B–boundary of a concept X induced by the set B of attributes, approximations over B are introduced, i.e.,
$\underline{B}X = \bigcup\{[u]_B : [u]_B \subseteq X\}$ (the B–lower approximation)

$\overline{B}X = \bigcup\{[u]_B : [u]_B \cap X \neq \emptyset\}$ (the B–upper approximation).

The difference $Bd_B X = \overline{B}X \setminus \underline{B}X$ is the B–boundary of X; when non–empty it does witness that X is rough.
For a rough concept X, one has the double strict inclusion: $\underline{B}X \subset X \subset \overline{B}X$ as the description of X in terms of two nearest to it exact concepts.

2.1.4 Knowledge reduction. Reducts

Knowledge represented in an information system (U, A) can be reduced: a reduct B of the set A of attributes is a minimal subset of A with the property that $IND(B) = INDd(A)$. Thus, reducts are minimal with respect to inclusion sets of attributes which preserve classification, i.e., knowledge.

Finding all reducts is computationally hard: the problem of finding a minimal length reduct is NP–hard, see [35].

An algorithm for finding reducts based on Boolean Reasoning technique was proposed in [35]; the method of Boolean Reasoning consists in solving a problem by constructing a Boolean function whose prime implicants would give solutions to the problem [3].

The Skowron–Rauszer algorithm for reduct induction: a case of Boolean Reasoning

In the context of an information system (U, A), the method of Boolean Reasoning for reduct finding proposed by Skowron and Rauszer [35], given input (U, A) with $U = \{u_1, ..., u_n\}$, starts with the *discernibility matrix*,
$M_{U,A} = [c_{i,j} = \{a \in A : a(u_i) \neq a(u_j)\}]_{1 \geq i,j \leq n}$,
and builds the Boolean function in the CNF form,
$f_{U,A} = \bigwedge_{c_{i,j} \neq \emptyset, i<j} \bigvee_{a \in c_{i,j}} \overline{a}$, where $\overline{a}$ is the Boolean variable assigned to the attribute $a \in A$.

The function $f_{U,A}$ is converted to its DNF form: $f^*_{U,A} : \bigvee_{j \in J} \bigwedge_{k \in K_j} \overline{a_{j,k}}$.

Then: sets of the form $R_j = \{a_{j,k} : k \in K_j\}$ for $j \in J$, corresponding to prime implicants $\bigwedge_{k \in K_j} \overline{a_{j,k}}$ are all reducts of A.

On the soundness of the algorithm We give here a proof of the soundness of the algorithm in order to acquaint the reader with this method which is also exploited in a few variants described below; the reader will be able to supply own proofs in those cases on the lines shown here.

We consider a set B of attributes and the valuation val_B on the Boolean variable set $\{\overline{a} : a \in A\}$: $val_B(\overline{a}) = 1$ in case $a \in B$ and 0, otherwise.

Assume that the Boolean function $f_{U,A}$ is satisfied under this valuation:

$val_B(f_{U,A}) = 1$. This means that $val_B(\bigvee_{a \in c_{i,j}} \overline{a}) = 1$ for each $c_{i,j} \neq \emptyset$. An equivalent formula to this statement is: $\forall i, j.c_{i,j} \neq \emptyset \Rightarrow \exists a \in c_{i,j}.a \in B$. Applying tautology $p \Rightarrow q \Leftrightarrow \neg q \Rightarrow \neg p$ to the last implication, we obtain: $\forall a \in B.a \notin c_{i,j} \Rightarrow \forall a \in A.a \notin c_{i,j}$ for each pair i, j. By definition of the set $c_{i,j}$, the last implication reads: $IND(B) \subseteq IND(A)$. This means $IND(B) = IND(A)$ as $IND(A) \subseteq IND(B)$ always because $B \subseteq A$.

Now, we have $val_B(f^*_{U,A}) = 1$ as well; this means that $val_B(\bigwedge_{k \in K_j} \overline{a_{j,k}}) = 1$ for some $j_o \in J$. In turn, by definition of val_B, this implies that $B \subseteq \{a_{j_o,k} : k \in K_{j_o}\}$.

A conclusion from the comparison of values of val_B on $f_{U,A}$ and $f^*_{U,A}$ is that : $IND(B) = IND(A)$ if and only if $B \subseteq \{a_{j,k} : k \in K_j\}$ for the $j - th$ prime implicant of $f_{U,A}$. Thus, any minimal with respect to inclusion set B of attributes such that $IND(B) = IND(A)$ coincides with a set of attributes $\{a_{j,k} : k \in K_j\}$ corresponding to a prime implicant of the function $f_{U,A}$.

Choosing a reduct R, and forming the reduced information system (U, R) one is assured that no information encoded in (U, A) has been lost.

2.1.5 Decision systems. Decision rules: an introduction

A decision system $(U, A \cup \{d\})$ encodes information about the external classification d (by an oracle, expert etc.). Methods based on rough sets aim at finding a description of the concept d in terms of conditional attributes in A in the language of descriptors. This description is fundamental for expert systems, knowledge based systems and applications in Data Mining and Knowledge Discovery.

Formal expressions for relating knowledge in conditional part (U, A) to knowledge of an expert in (U, d) are *decision rules*; in descriptor logic they are of the form $\phi^B_U \Rightarrow (d = w)$, where $w \in V_d$, the value set of the decision.

Semantics of decision rules is given by general rules set in sect. 2.1.2: the rule $\phi^B_U \Rightarrow (d = w)$ is *certain* or *true* in case $[\phi^B_u] \subseteq [d = w]$, i.e., in case when each object v that satisfies ϕ^B_u, i.e., $(u, v) \in IND(B)$, satisfies also $d(v) = w$; otherwise the rule is said to be *partial.*

The simpler case is when the decision system is *deterministic*, i.e., $IND(A) \subseteq IND(d)$. In this case the relation between A and d is *functional*, given by the unique assignment $f_{A,d} : Inf_A(u) \rightarrow Inf_d(u)$, or, in the decision rule form as the set of rules: $\bigwedge_{a \in A}(a = a(u)) \Rightarrow (d = d(u))$. Each of these rules is clearly certain.

In place of A any reduct R of A can be substituted leading to shorter certain rules.

In the contrary case, some classes $[u]_A$ are split into more than one decision class $[v]_d$ leading to ambiguity in classification. In order to resolve the ambiguity, the notion of a δ–reduct was proposed in [35]; it is called a relative reduct in [2].

To define δ– reducts, first the generalized decision δ_B is defined for any $B \subseteq A$: for $u \in U$, $\delta_B(u) = \{v \in V_d : d(u') = v \wedge (u, u') \in IND(B)$ for some $u' \in U\}$. A subset B of A is a δ–reduct to d when it is a minimal subset od A with respect to the property that $\delta_B = \delta_A$.

δ–reducts can be obtained from the modified Skowron and Rauszer algorithm [35]: it suffices to modify the entries $c_{i,j}$ to the discernibility matrix, by letting $c^d_{i,j} = \{a \in A \cup \{d\} : a(u_i) \neq a(u_j)\}$ and then setting $c'_{i,j} = c^d_{i,j} \setminus \{d\}$ in case $d(u_i) \neq d(u_j)$ and $c'_{i,j} = \emptyset$ in case $d(u_i) = d(u_j)$. The algorithm described above input with entries $c'_{i,j}$ forming the matrix $M^{\delta}_{U,A}$ outputs all δ–reducts to d encoded as prime implicants of the associated Boolean function $f^{\delta}_{U,A}$.

For any δ–reduct R, rules of the form $\phi^R_u \Rightarrow \delta = \delta_R(u)$ are certain.

An example of reduct finding and decision rule induction

We conclude the first step into rough sets with a simple example of a decision system, its reducts and decision rules.

Table 2.1 shows a simple decision system.

Table 2.1. Decision system Simple

obj.	a_1	a_2	a_3	a_4	d
u_1	1	0	0	1	0
u_2	0	1	0	0	1
u_3	1	1	0	0	1
u_4	1	0	0	1	1
u_5	0	0	0	1	1
u_6	1	1	1	1	0

Reducts of the information system $(U, A = \{a_1, a_2, a_3, a_4\})$ can be found from the discernibility matrix $M_{U,A}$ in Table 2.2; by symmetry, cells $c_{i,j} = c_{j,i}$ with $i > j$ are not filled. Each attribute a_i is encoded by the Boolean variable i.

After reduction by means of absorption rules of sentential calculus: $(p \vee q) \wedge p \Leftrightarrow p$, $(p \wedge q) \vee p \Leftrightarrow p$, the DNF form $f^*_{U,A}$ is $1 \wedge 2 \wedge 3 \vee 1 \wedge 2 \wedge 4 \vee 1 \wedge 3 \wedge 4$. Reducts of A in the information system (U, A) are : $\{a_1, a_2, a_3\}$, $\{a_1, a_2, a_4\}$, $\{a_1, a_3, a_4\}$.

δ–reducts of the decision d in the decision system Simple, can be found from the modified discernibility matrix $M^{\delta}_{U,A}$ in Table 2.3.

Table 2.2. Discernibility matrix $M_{U,A}$ for reducts in (U, A)

obj.	u_1	u_2	u_3	u_4	u_5	u_6
u_1	$\emptyset$	$\{1,2,4\}$	$\{2,4\}$	$\emptyset$	$\{1\}$	$\{2,3\}$
u_2	–	$\emptyset$	$\{1\}$	$\{1,2,3\}$	$\{2,4\}$	$\{1,3,4\}$
u_3	–	–	$\emptyset$	$\{2,4\}$	$\{2,4\}$	$\{3,4\}$
u_4	–	–	–	$\emptyset$	$\{1\}$	$\{2,3\}$
u_5	–	–	–	–	$\emptyset$	$\{1,2,3\}$
u_6	–	–	–	–	–	$\emptyset$

Table 2.3. Discernibility matrix $M^{\delta}_{U,A}$ for δ–reducts in (U, A, d)

obj.	u_1	u_2	u_3	u_4	u_5	u_6
u_1	$\emptyset$	$\{1,2,4\}$	$\{2,4\}$	$\emptyset$	$\{1\}$	$\emptyset$
u_2	–	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\{1,3,4\}$
u_3	–	–	$\emptyset$	$\emptyset$	$\emptyset$	$\{3,4\}$
u_4	–	–	–	$\emptyset$	$\emptyset$	$\{2,3\}$
u_5	–	–	–	–	$\emptyset$	$\{1,2,3\}$
u_6	–	–	–	–	–	$\emptyset$

From the Boolean function $f^{\delta}_{U,A}$ we read off δ–reducts $R_1 = \{a_1, a_2, a_3\}$, $R_2 = \{a_1, a_2, a_4\}$, $R_3 = \{a_1, a_3, a_4\}$.

Taking R_1 as the reduct for inducing decision rules, we read the following certain rules:

$r_1 : (a_1 = 0) \wedge (a_2 = 1) \wedge (a_3 = 0) \Rightarrow (d = 1)$;
$r_2 : (a_1 = 1) \wedge (a_2 = 1) \wedge (a_3 = 0) \Rightarrow (d = 1)$;
$r_3 : (a_1 = 0) \wedge (a_2 = 0) \wedge (a_3 = 0) \Rightarrow (d = 1)$;
$r_4 : (a_1 = 1) \wedge (a_2 = 1) \wedge (a_3 = 1) \Rightarrow (d = 0)$;
and two possible rules
$r_5 : (a_1 = 1) \wedge (a_2 = 0) \wedge (a_3 = 0) \Rightarrow (d = 0)$;
$r_6 : (a_1 = 1) \wedge (a_2 = 0) \wedge (a_3 = 0) \Rightarrow (d = 1)$,
each with certainty factor =.5 as there are two objects with d=0.

2.1.6 Decision rules: advanced topics

In order to precisely discriminate between certain and possible rules, the notion of a *positive region* along with the notion of a *relative reduct* was proposed and studied in [35].

Positive region $pos_B(d)$ is the set $\{u \in U : [u]_B \subseteq [u]_d\} = \bigcup_{v \in V_d} \underline{B}[(d = v)]$; $pos_B(d)$ is the greatest subset X of U such that $(X, B \cup \{d\})$ is deterministic; it generates certain rules. Objects in $U \setminus pos_B(d)$ are subjected to ambiguity: given such u, and the collection $v_1, .., v_k$ of decision d values on the class $[u]_B$, the decision rule describing u can be formulated as, $\bigwedge_{a \in B}(a = a(u)) \Rightarrow \bigvee_{i=1,...,k}(d = v_i)$; each of the rules $\bigwedge_{a \in B}(a = a(u)) \Rightarrow (d = v_i)$ is possible but not certain as only for a fraction of objects in the class $[u]_B$ the decision takes the value v_i on.

Relative reducts are minimal sets B of attributes with the property that $pos_B(d) = pos_A(d)$; they can also be found by means of discernibility matrix $M^*_{U,A}$ [90]: $c^*_{i,j} = c^d_{i,j} \setminus \{d\}$ in case either $d(u_i) \neq d(u_j)$ and $u_i, u_j \in pos_A(d)$ or $pos(u_i) \neq pos(u_j)$ where pos is the characteristic function of $pos_A(d)$; otherwise, $c^*_{i,j} = \emptyset$.

For a relative reduct B, certain rules are induced from the deterministic system $(pos_B(d), A \cup \{d\})$, possible rules are induced from the non–deterministic system $(U \setminus pos_B(d), A \cup \{d\})$. In the last case, one can find δ–reducts to d in this system and turn the system into a deterministic one $(U \setminus pos_B(d), A, \delta)$ inducing certain rules of the form $\bigwedge_{a \in B}(a = a(u)) \Rightarrow \bigvee_{v \in \delta(u)}(d = v)$.

A method for obtaining decision rules with minimal number of descriptors [22], [34], consists in reducing a given rule $r : \phi/B, u \Rightarrow (d = v)$ by finding a set $R_r \subseteq B$ consisting of irreducible attributes in B only, in the sense that removing any $a \in R_r$ causes the inequality $[\phi/R_r, u \Rightarrow (d = v)] \neq [\phi/R_r \setminus \{a\}, u \Rightarrow (d = v)]$ to hold. In case $B = A$, reduced rules $\phi/R_r, u \Rightarrow (d = v)$ are called optimal basic rules (with minimal number of descriptors). The method for finding of all irreducible subsets of the set A [34], consists in considering another modification of discernibility matrix: for each object $u_k \in U$, the entry $c'_{i,j}$ into the matrix $M^\delta_{U,A}$ for δ–reducts is modified into $c^k_{i,j} = c'_{i,j}$ in case $d(u_i) \neq d(u_j)$ and $i = k \vee j = k$, otherwise $c^k_{i,j} = \emptyset$. Matrices $M^k_{U,A}$ and associated Boolean functions $f^k_{U,A}$ for all $u_k \in U$ allow for finding all irreducible subsets of the set A and in consequence all basic optimal rules (with minimal number of descriptors).

Decision rules are judged by their quality on the basis of the training set and by quality in classifying new unseen as yet objects, i.e., by their performance on the test set. Quality evaluation is done on the basis of some measures: for a rule $r : \phi \Rightarrow (d = v)$, and an object $u \in U$, one says that u matches r in case $u \in [\phi]$. $match(r)$ is the number of objects matching r. Support $supp(r)$ of r is the number of objects in $[\phi] \cap [(d = v)]$; the fraction $cons(r)=\frac{supp(r)}{match(r)}$ is the consistency degree of r: $cons(r) = 1$ means that the rule is certain.

Strength, $strength(r)$, of the rule r is defined, as the number of objects correctly classified by the rule in the training phase [15], [1], [8]; relative strength is defined as the fraction $rel - strength(r)= \frac{supp(r)}{|[(d=v)]|}$. Specificity of the rule r, $spec(r)$, is the number of descriptors in the premise ϕ of the rule r.

In the testing phase, rules vie among themselves for object classification when they point to distinct decision classes; in such case, negotiations among rules or their sets are necessary. In these negotiations, rules with better characteristics are privileged.

For a given decision class $c : d = v$, and an object u in the test set, the set $Rule(c, u)$ of all rules matched by u and pointing to the decision v, is characterized globally by $Support(Rule(c, u))=\sum_{r \in Rule(c,u)} strength(r) \cdot$

$spec(r)$. The class c for which $Support(Rule(c,u))$ is the largest wins the competition and the object u is classified into the class $c : d = v$.

It may happen that no rule in the available set of rules is matched by the test object u and partial matching is necessary, i.e., for a rule r, the matching factor $match-fact(r,u)$ is defined as the fraction of descriptors in the premise ϕ of r matched by u to the number $spec(r)$ of descriptors in ϕ. The rule for which the partial support $Part - Support(Rule(c,u))=\sum_{r\in Rule(c,u)} match - fact(r,u)\cdot strength(r)\cdot spec(r)$ is the largest wins the competition and it does assign the value of decision to u.

2.2 Discretization of continuous valued attributes

The important problem of treating continuous values of attributes has been resolved in rough sets with the help of discretization of attributes technique, common to many paradigms like decision trees, etc.; for a decision system (U,A,d), a cut is a pair (a,c), where $a \in A, c$ in reals. The cut (a,c) induces the binary attribute $b_{a,c}(u) = 1$ if $a(u) \geq c$ and it is 0, otherwise. Given a finite sequence $p_a{=}c_0^a < c_1^a < .... < c_m^a$ of reals, the set V_a of values of a is split into disjoint intervals: $(\leftarrow, c_0^a), [c_0^a, c_1^a), ..., [c_m^a, \rightarrow)$; the new attribute $D_a(u) = i$ when $b_{c_{i+1}^a} = 0, b_{c_i^a} = 1$, is a discrete counterpart to the continuous attribute a. Given a collection $P = \{p_a : a \in A\}$ (a cut system), the set $D = \{D_a : a \in A\}$ of attributes transforms the system (U,A,d) into the discrete system (U, D_P, d) called the P–segmentation of the original system. The set P is consistent in case generalized decision in both systems is identical, i.e., $\delta_A = \delta_{D_P}$; a consistent P is irreducible if P' is not consistent for any proper subset $P' \subset P$; P is optimal if its cardinality is minimal among all consistent cut systems, see [16], [17].

2.3 Classification

Classification methods can be divided according to the adopted methodology, into classifiers based on reducts and decision rules, classifiers based on templates and similarity, classifiers based on descriptor search, classifiers based on granular descriptors, hybrid classifiers.

For a decision system (U,A,d), classifiers are sets of decision rules. Induction of rules was a subject of research in rough set theory since its beginning. In most general terms, building a classifier consists in searching in the pool of descriptors for their conjuncts that describe decision classes sufficiently well. As distinguished in [37], there are three main kinds of classifiers searched for: *minimal*, i.e., consisting of the minimum possible number of rules describing decision classes in the universe, *exhaustive*, i.e., consisting of all possible rules, *satisfactory*, i.e., containing rules tailored to a specific use. Classifiers are evaluated globally with respect to their ability to properly classify objects, usually by *error* which is the ratio of the number of correctly classified objects to the number of test objects, *total accuracy* being the ratio of the number of correctly classified cases to the number of recognized cases, and *total coverage*, i.e, the ratio of the number of recognized test cases to the number of test cases.

Minimum size algorithms include LEM2 algorithm due to Grzymala–Busse [9] and covering algorithm in the RSES package [33]; exhaustive algorithms include, e.g., LERS system due to Grzymala–Busse [7], systems based on discernibility matrices and Boolean reasoning [34], see also [1], [2], implemented in the RSES package [33].

Minimal consistent sets of rules were introduced in Skowron and Rauszer [35]. Further developments include dynamic rules, approximate rules, and relevant rules as described in [1], [2], as well as local rules (op. cit.) effective in implementations of algorithms based on minimal consistent sets of rules. Rough set based classification algorithms, especially those implemented in the RSES system [33], were discussed extensively in [2].

In [1], a number of techniques were verified in experiments with real data, based on various strategies:

`discretization` of attributes (codes: N-no discretization, S-standard discretization, D-cut selection by dynamic reducts, G-cut selection by generalized dynamic reducts);

`dynamic selection of attributes` (codes: N-no selection, D-selection by dynamic reducts, G-selection based on generalized dynamic reducts);

`decision rule choice` (codes: A-optimal decision rules, G-decision rules on basis of approximate reducts computed by Johnson's algorithm, simulated annealing and Boltzmann machines etc., N-without computing of decision rules);

`approximation of decision rules` (codes: N-consistent decision rules, P-approximate rules obtained by descriptor dropping);

`negotiations among rules` (codes: S-based on strength, M-based on maximal strength, R-based on global strength, D-based on stability).

Any choice of a strategy in particular areas yields a compound strategy denoted with the alias being concatenation of symbols of strategies chosen in consecutive areas, e.g., NNAND etc.

We record here in Table 2.4 an excerpt from the comparison (Table 8, 9, 10 in [1]) of best of these strategies with results based on other paradigms in classification for two sets of data: Diabetes and Australian credit from UCI Repository [40].

An adaptive method of classifier construction was proposed in [43]; reducts are determined by means of a genetic algorithm, see [2], and in turn reducts induce subtables of data regarded as classifying agents; choice of optimal ensembles of agents is done by a genetic algorithm.

Table 2.4. A comparison of errors in classification by rough set and other paradigms

paradigm	*system/method*	*Diabetes*	*Austr.credit*
Stat.Methods	*Logdisc*	0.223	0.141
Stat.Methods	*SMART*	0.232	0.158
Neural Nets	*Backpropagation*2	0.248	0.154
Neural Networks	*RBF*	0.243	0.145
Decision Trees	*CART*	0.255	0.145
Decision Trees	*C*4.5	0.270	0.155
Decision Trees	*ITrule*	0.245	0.137
Decision Rules	*CN*2	0.289	0.204
Rough Sets	*NNANR*	0.335	0.140
Rough Sets	*DNANR*	0.280	0.165
Rough Sets	*best result*	0.255(*DNAPM*)	0.130(*SNAPM*)

2.4 Approaches to classification in data based on similarity

Algorithms mentioned in sect. 2.3 were based on indiscernibility relations which are equivalence relations. A softer approach is based on similarity relations, i.e., relations that are reflexive and possibly symmetric but need not be transitive. Classes of these relations provide coverings of the universe U instead of its partitions.

2.4.1 Template approach

Classifiers of this type were constructed by means of templates matching a given object or closest to it with respect to a certain distance function, or on coverings of the universe of objects by tolerance classes and assigning the decision value on basis of some of them [18]; we include in Table 2.5 excerpts from classification results in [18].

Table 2.5. Accuracy of classification by template and similarity methods

paradigm	*system/method*	*Diabetes*	*Austr.credit*
Rough Sets	*Simple.templ./Hamming*	0.6156	0.8217
Rough Sets	*Gen.templ./Hamming*	0.742	0.855
Rough Sets	*Simple.templ./Euclidean*	0.6312	0.8753
Rough Sets	*Gen.templ./Euclidean*	0.7006	0.8753
Rough Sets	*Match.tolerance*	0.757	0.8747
Rough Sets	*Clos.tolerance*	0.743	0.8246

A combination of rough set methods with the k–nearest neighbor idea is a further refinement of the classification based on similarity or analogy in [42]. In this approach, training set objects are endowed with a metric, and the test objects are classified by voting by k nearest training objects for some k that is subject to optimization.

2.4.2 Similarity measures based on rough inclusions

Rough inclusions offer a systematic way for introducing similarity into object sets. A rough inclusion $\mu(u, v, r)$ (read: u is a part of v to the degree of at least r) introduces a similarity that is not symmetric.

Rough inclusions in an information system (U, A) can be induced in some distinct ways as in [25], [27]. We describe here just one method based on using Archimedean t–norms, i.e., t–norms $t(x, y)$ that are continuous and have no idempotents, i.e., values x with $t(x, x) = x$ except $0, 1$ offer one way; it is well–known, see, e.g., [23], that up to isomorphism, there are two Archimedean t–norms: the Łukasiewicz t–norm $L(x, y) = max\{0, x + y - 1\}$ and the product (Menger) t–norm $P(x, y) = x \cdot y$. Archimedean t–norms admit a functional characterization, see, e.g, [23]: $t(x, y) = g(f(x) + f(y))$, where the function $f : [0, 1] \rightarrow R$ is continuous decreasing with $f(1) = 0$, and $g : R \rightarrow [0, 1]$ is the pseudo–inverse to f, i.e., $f \circ g = id$. The t–induced rough inclusion μ_t is defined [24] as $\mu_t(u, v, r) \Leftrightarrow g(\frac{|DIS(u,v)|}{|A|}) \geq r$ where $DIS(u, v) = \{a \in A : a(u) \neq a(v)\}$. With the Łukasiewicz t–norm, $f(x) = 1 - x = g(x)$ and $IND(u, v) = U \times U \setminus DIS(u, v)$, the formula becomes: $\mu_L(u, v, r) \Leftrightarrow \frac{|IND(u,v)|}{|A|} \geq r$; thus in case of Łukasiewicz logic, μ_L becomes the similarity measure based on the Hamming distance between information vectors of objects reduced modulo $|A|$; from probabilistic point of view, it is based on the relative frequency of descriptors in information sets of u, v. This formula permeates data mining algorithms and methods, see [10].

2.5 Granulation of knowledge

The issue of granulation of knowledge as a problem on its own, has been posed by L.A. Zadeh [44]. Granulation can be regarded as a form of clustering, i.e., grouping objects into aggregates characterized by closeness of certain parameter values among objects in the aggregate and greater differences in those values from aggregate to aggregate. The issue of granulation has been a subject of intensive studies within rough set community in, e.g., [14], [29], [31].

Rough set context offers a natural venue for granulation, and indiscernibility classes were recognized as *elementary granules* whereas their unions serve as *granules of knowledge.*

For an information system (U, A), and a rough inclusion μ on U, granulation with respect to similarity induced by μ is formally performed by exploiting the class operator Cls of mereology [13]. The class operator is applied to any non–vacuous property F of objects (i.e. a distributive entity) in the universe U and produces the object $ClsF$ (i.e., the collective entity) representing wholeness of F. The formal definition of Cls is: assuming a part relation in U and the associated ingredient relation ing, $ClsF$ does satisfy conditions,

1. if $u \in F$ then u is ingredient of $ClsF$.

2. if v is an ingredient of $ClsF$ then some ingredient w of v is an ingredient as well of a T that is in F;

in plain words, each ingredient of $ClsF$ has an ingredient in common with an object in F. An example of part relation is the proper subset $\subset$ relation on a family of sets; then the subset relation $\subseteq$ is the ingredient relation, and the class of a family F of sets is its union $\bigcup F$. The merit of class operator is in the fact that it always projects hierarchies onto the collective entity plane containing objects.

For an object u and a real number $r \in [0,1]$, we define the granule $g_\mu(u,r)$ about u of the radius r, relative to μ, as the class $ClsF(u,r)$, where the property $F(u,r)$ is satisfied with an object v if and only if $\mu(v,u,r)$ holds.

It was shown [24] that in case of a transitive μ, v is an ingredient of the granule $g_\mu(u,r)$ if and only if $\mu(v,u,r)$. This fact allows for writing down the granule $g_\mu(u,r)$ as a distributive entity (a set, a list) of objects v satisfying $\mu(v,u,r)$.

Granules of the form $g_\mu(u,r)$ have regular properties of a neighborhood system [25]. Granules generated from a rough inclusion μ can be used in defining a compressed form of the decision system: a granular decision system [25]; for a granulation radius r, and a rough inclusion μ, we form the collection $U^G_{r,\mu} = \{g_\mu(u,r)\}$. We apply a strategy $\mathcal{G}$ to choose a covering $Cov^G_{r,\mu}$ of the universe U by granules from $U^G_{r,\mu}$. We apply a strategy $\mathcal{S}$ in order to assign the value $a^*(g)$ of each attribute $a \in A$ to each granule $g \in Cov^G_{r,\mu}$: $a^*(g) = \mathcal{S}(\{a(u) : u \in g\})$. The granular counterpart to the decision system (U, A, d) is a tuple $(U^G_{r,\mu}, \mathcal{G}, \mathcal{S}, \{a* : a \in A\}, d^*)$. The heuristic principle that H: *objects, similar with respect to conditional attributes in the set A, should also reveal similar (i.e., close) decision values, and therefore, granular counterparts to decision systems should lead to classifiers satisfactorily close in quality to those induced from original decision systems* that is at the heart of all classification paradigms, can be also formulated in this context [25]. Experimental results bear out the hypothesis [28].

The granulated data set offers a compression of the size of the training set and a fortiori, a compression in size of the rule set. Table 2.6 shows this on the example of Pima Indians Diabetes data set [40]. Exhaustive algorithm of RSES [33] has been applied as the rule inducting algorithm. Granular covering has been chosen randomly, majority voting has been chosen as the strategy $\mathcal{S}$. Results have been validated by means of 10–fold cross validation, see, e.g., [5]. The radii of granulation have been determined by the chosen rough inclusion μ_L: according to its definition in sect.2.4.2, an object v is in the granule $g_r(u)$ in case at least r fraction of attributes agree on u and v; thus, values of r are multiplicities of the fraction $\frac{1}{|A|}$ less or equal to 1. The radius “nil” denotes the results of non–granulated data analysis.

Table 2.6. 10-fold CV; Pima; exhaustive algorithm. r=radius, macc=mean accuracy, mcov=mean coverage, mrules=mean rule number, mtrn=mean size of training set

r	*macc*	*mcov*	*mrules*	*mtrn*
nil	0.6864	0.9987	7629	692
0.125	0.0618	0.0895	5.9	22.5
0.250	0.6627	0.9948	450.1	120.6
0.375	0.6536	0.9987	3593.6	358.7
0.500	0.6645	1.0	6517.6	579.4
0.625	0.6877	0.9987	7583.6	683.1
0.750	0.6864	0.9987	7629.2	692
0.875	0.6864	0.9987	7629.2	692

For the exhaustive algorithm, the accuracy in granular case exceeds or equals that in non–granular case from the radius of .625 with slightly smaller sizes of training as well as rule sets and it reaches 95.2 percent of accuracy in non–granular case, from the radius of .25 with reductions in size of the training set of 82.6 percent and in the rule set size of 94 percent. The difference in coverage is less than .4 percent from $r = .25$ on, where reduction in training set size is 82.6 percent, and coverage in both cases is the same from the radius of .375 on with reductions in size of both training and rule set of 48, resp., 53 percent.

The fact of substantial reduction in size of the training set as well in size of the rule set coupled with the fact of a slight only decrease in classification accuracy testifies to validity of the idea of granulated data sets; this can be of importance in case of large biological or medical data sets which after granulation would become much smaller and easier to analyze.

2.5.1 Concept–dependent granulation

A variant of granulation idea is the concept-dependent granulation [28] in which granules are computed relative to decision classes, i.e., the restricted granule $g\mu^d(u,r)$ is equal to the intersection $g_\mu(u,r) \cap [d = d(u)]$ of the granule $g_\mu(u,r)$ with the decision class $[d = d(u)]$ of u. At the cost of an increased number of granules, the accuracy of classification is increased. In Table 2.7, we show the best results of classification obtained by means of various rough set methods on Australian credit data set [40]. The best result is obtained with concept–dependent granulation.

Table 2.7. Best results for Australian credit by some rough set based algorithms; in case $*$, reduction in object size is 49.9 percent, reduction in rule number is 54.6 percent; in case $**$, resp., 19.7, 18.2; in case $***$, resp., 3.6, 1.9

source	*method*	*accuracy*	*coverage*
[1]	$SNAPM(0.9)$	$error = 0.130$	–
[18]	$simple.templates$	0.929	0.623
[18]	$general.templates$	0.886	0.905
[18]	$closest.simple.templates$	0.821	1.0
[18]	$closest.gen.templates$	0.855	1.0
[18]	$tolerance.simple.templ.$	0.842	1.0
[18]	$tolerance.gen.templ.$	0.875	1.0
[43]	$adaptive.classifier$	0.863	–
[28]	$granular^{*}.r = 0.642$	0.8990	1.0
[28]	$granular^{**}.r = 0.714$	0.964	1.0
[28]	$granular^{***}.concept.r = 0.785$	0.9970	0.9995

2.6 Missing values

Incompleteness of data sets is an important problem in data especially biological and medical in which case often some attribute values have not been recorded due to difficulty or impossibility of obtaining them. An information/decision system is *incomplete* in case some values of conditional attributes from A are not known; some authors, e.g., Grzymala–Busse [8], [9], make distinction between values that are *lost* (denoted ?), i.e., they were not recorded or were destroyed in spite of their importance for classification, and values that are *missing* (denoted $*$) as those values that are not essential for classification. Here, we regard all lacking values as missing without making any distinction among them denoting all of them with $*$. Analysis of systems with missing values requires a decision on how to treat such values; Grzymala–Busse in his work [8], analyzes nine such methods known in the literature, among them, *1. most common attribute value*, *2. concept–restricted most common attribute value*, (...), *4. assigning all possible values to the missing location*, (...), *9. treating the unknown value as a new valid value*. Results of tests presented in [8] indicate that methods *4,9* perform very well among all nine methods. For this reason we adopt these methods in this work for the treatment of missing values and they are combined in our work with a modified method *1*: the missing value is defined as the most frequent value in the granule closest to the object with the missing value with respect to a chosen rough inclusion.

Analysis of decision systems with missing data in existing rough set literature relies on an appropriate treatment of indiscernibility: one has to reflect in this relation the fact that some values acquire a distinct character and must be treated separately; in case of missing or lost values, the relation of indiscernibility is usually replaced with a new relation called a *characteristic relation.* Examples of such characteristic functions are given in, e.g., Grzymala–Busse [9]: the function ρ is introduced, with $\rho(u, a) = v$ meaning that the attribute a takes on u the value v. Semantics of descriptors is changed, viz., the meaning $[(a = v)]$ has as elements all u such that $\rho(u, a) = v$, in case $\rho(u, a) =?$ the entity u is not included into $[(a = v)]$, and in case $\rho(u, a) = *$, the entity u is included into $[(a = v)]$ for all values $v \neq *, ?$. Then the characteristic relation is $R(B) = \{(u, v) : \forall.a \in B.\rho(u, a) =? \Rightarrow (\rho(u, a) = \rho(v, a) \vee \rho(u, a) = * \vee \rho(v, a) = *)\}$, where $B \subseteq A$. Classes of the relation $R(B)$ are then used in defining approximations to decision classes from which certain and possible rules are induced, see [9]. Specializations of the characteristic relation $R(B)$ were defined in [38] (in case of only lost values) and in [11] (in case of only "*don't care*" missing values). An analysis of the problem of missing values along with algorithms *IApriori Certain* and *IAprioriPossible* for certain and possible rule generation was given in [12].

We will use the symbol $*$ commonly used for denoting the missing value; we will use two methods *4, 9* for treating $*$, i.e, either $*$ is a "*don't care*" symbol meaning that any value of the respective attribute can be substituted for $*$,thus $* = v$ for each value v of the attribute, or $*$ is a new value on its own, i.e., if $* = v$ then v can be only $*$.

Our procedure for treating missing values is based on the granular structure $(U^G_{r,\mu}, \mathcal{G}, \mathcal{S}, \{a^* : a \in A\})$; the strategy $\mathcal{S}$ is the majority voting, i.e., for each attribute a, the value $a^*(g)$ is the most frequent of values in $\{a(u) : u \in g\}$, with ties broken randomly. The strategy $\mathcal{G}$ consists in random selection of granules for a covering.

For an object u with the value of $*$ at an attribute a, and a granule $g = g(v, r) \in U^G_{r,\mu}$, the question whether u is included in g is resolved according to the adopted strategy of treating $*$: in case $* =$ *don 't care*, the value of $*$ is regarded as identical with any value of a hence $|IND(u, v)|$ is automatically increased by 1, which increases the granule; in case $* = *$, the granule size is decreased. Assuming that $*$ is sparse in data, majority voting on g would produce values of a^* distinct from $*$ in most cases; nevertheless the value of $*$ may appear in new objects g^*, and then in the process of classification, such value is repaired by means of the granule closest to g^* with respect to the rough inclusion μ_L, in accordance with the chosen method for treating $*$.

In plain words, objects with missing values are in a sense absorbed by close to them granules and missing values are replaced with most frequent values in objects collected in the granule; in this way the method *4* or *9* in [8] is combined with the idea of the most frequent value *1*, in a novel way.

We have thus four possible strategies:

- Strategy A: in building granules $*=$*don't care*, in repairing values of $*$, $*=$*don't care*;
- Strategy B: in building granules $*=$*don't care*, in repairing values of $*$, $* = *$;
- Strategy C: in building granules $* = *$, in repairing values of $*$, $*=$*don't care*;
- Strategy D: in building granules $* = *$, in repairing values of $*$, $* = *$.

2.7 Case of real data with missing values

We include results of tests with Breast cancer data set [40] that contains missing values. We show in Tables 2.8, 2.9, 2.10, 2.11, results for intermediate values of radii of granulation for strategies A,B,C,D and exhaustive algorithm of RSES [33]. For comparison, results on error in classification by the endowed system LERS from [8] for approaches similar to our strategies A and D (methods 4 and 9, resp., in Tables 2 and 3 in [8]) in which ∗ is either always ∗ (method 9) or ∗ is always *don't care* (method 4) are recalled in Tables 2.8 and 2.11. We have applied here the 1-train–and–9 test, i.e., the data set is split randomly into 10 equal parts and training set is one part whereas the rules are tested on each of remaining 9 parts separately and results are averaged.

Table 2.8. Breast cancer data set with missing values. Strategy A: r=granule radius, mtrn=mean granular training sample size, macc=mean accuracy, mcov=mean covering, gb=LERS method 4, [8]

r	*mtrn*	*macc*	*mcov*	*gb*
0.555556	9	0.7640	1.0	0.7148
0.666667	14	0.7637	1.0	
0.777778	17	0.7129	1.0	
0.888889	25	0.7484	1.0	

Table 2.9. Breast cancer data set with missing values. Strategy B: r=granule radius, mtrn=mean granular training sample size, macc=mean accuracy, mcov=mean covering

r	*mtrn*	*macc*	*mcov*
0.555556	7	0.0	0.0
0.666667	13	0.7290	1.0
0.777778	16	0.7366	1.0
0.888889	25	0.7520	1.0

Table 2.10. Breast cancer data set with missing values. Strategy C: r=granule radius, mtrn=mean granular training sample size, macc=mean accuracy, mcov=mean covering

r	*mtrn*	*macc*	*mcov*
0.555556	8	0.7132	1.0
0.666667	14	0.6247	1.0
0.777778	17	0.7328	1.0
0.888889	25	0.7484	1.0

Table 2.11. Breast cancer data set with missing values. Strategy D: r=granule radius, mtrn=mean granular training sample size, macc=mean accuracy, mcov=mean covering, gb=LERS method 9 [8]

r	*mtrn*	*macc*	*mcov*	*gb*
0.555556	9	0.7057	1.0	0.6748
0.666667	16	0.7640	1.0	
0.777778	17	0.6824	1.0	
0.888889	25	0.7520	1.0	

A look at Tables 2.8–2.11 shows that granulated approach gives with Breast cancer data better results than obtained earlier with the LERS method. This strategy deserves therefore attention.

2.8 Applications of rough sets

A number of software systems for inducing classifiers were proposed based on rough set methodology, among them LERS by Grzymala–Busse ; TRANCE due to Kowalczyk; RoughFamily by Słowiński and Stefanowski; TAS by Suraj; PRIMEROSE due to Tsumoto; KDD-R by Ziarko; RSES by Skowron et al; ROSETTA due to Komorowski, Skowron et al; RSDM by Fernandez–Baizan et al; GROBIAN due to Duentsch and Gediga RoughFuzzyLab by Swiniarski. All these systems are presented in [30].

Rough set techniques were applied in many areas of data exploration, among them in exemplary areas:

`Processing of audio signals`: [4].

`Pattern recognition`: [36].

`Signal classification`: [41].

`Image processing`: [39].

`Rough neural computation modeling`: [26].

`Self organizing maps`: [19].

`Learning cognitive concepts`: [32].

2.9 Concluding remarks

Basic ideas, methods and results obtained within the paradigm of rough sets by efforts of many researchers, both in theoretical and application oriented aspects, have been recorded in this Chapter. Further reading, in addition to works listed in References, may be directed to the following monographs or collections of papers:
A. Polkowski L, Skowron, A (eds.) (1998) Rough Sets in Knowledge Discovery, Vols. 1 and 2, Physica Verlag, Heidelberg

B. Inuiguchi M, Hirano S, Tsumoto S (eds.) (2003) Rough Set Theory and Granular Computing, Springer, Berlin

C. Transactions on Rough Sets I. Lecture Notes in Computer Science (2004) 3100, Springer, Berlin

D. Transactions on Rough Sets II. Lecture Notes in Computer Science (2004) 3135, Springer Verlag, Berlin

E. Transactions on Rough Sets III. Lecture Notes in Computer Science (2005) 3400, Springer, Berlin

F. Transactions on Rough Sets IV. Lecture Notes in Computer Science (2005) 3700, Springer Verlag, Berlin

G. Transactions on Rough Sets V. Lecture Notes in Computer Science (2006) 4100, Springer, Berlin

H. Transactions on Rough Sets VI. Lecture Notes in Computer Science (2006) 4374, Springer, Berlin

References

1. Bazan JG (1998) A comparison of dynamic and non–dynamic rough set methods for extracting laws from decision tables. In: Polkowski L, Skowron A (eds.), Rough Sets in Knowledge Discovery 1. Physica, Heidelberg 321–365
2. Bazan JG, Synak P, Wróblewski J, Nguyen SH, Nguyen HS (2000) Rough set algorithms in classification problems. In: Polkowski L, Tsumoto S, Lin TY (eds.) Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems, Physica , Heidelberg 49–88
3. Brown MF (2003) Boolean Reasoning: The Logic of Boolean Equations, 2nd ed., Dover, New York
4. Czyżewski A, et al. (2004) Musical phrase representation and recognition by means of neural networks and rough sets, Transactions on Rough Sets I. Lecture Notes in Computer Science 3100, Springer, Berlin 254–278

5. Duda RO, Hart PE, Stork DG (2001) Pattern Classification, John Wiley and Sons, New York
6. Frege G (1903) Grundlagen der Arithmetik II, Jena
7. Grzymala–Busse JW (1992) LERS – a system for learning from examples based on rough sets. In: Słowiński R (ed.) Intelligent Decision Support: Handbook of Advances and Applications of the Rough Sets Theory. Kluwer, Dordrecht 3–18
8. Grzymala–Busse JW, Ming H (2000) A comparison of several approaches to missing attribute values in data mining, Lecture Notes in AI 2005, Springer, Berlin, 378–385
9. Grzymala–Busse JW (2004) Data with missing attribute values: Generalization of indiscernibility relation and rule induction, Transactions on Rough Sets I. Lecture Notes in Computer Science 3100, Springer, Berlin 78–95
10. Klösgen W, Żytkow J (eds.) (2002) Handbook of Data Mining and Knowledge Discovery, Oxford University Press, Oxford
11. Kryszkiewicz M (1999) Rules in incomplete information systems, Information Sciences 113:271–292
12. Kryszkiewicz M, Rybiński H (2000) Data mining in incomplete information systems from rough set perspective. In: Polkowski L, Tsumoto S, Lin TY (eds.) Rough Set Methods and Applications, Physica Verlag, Heidelberg 568–580
13. Leśniewski S (1916) Podstawy Ogólnej Teoryi Mnogosci (On the Foundations of Set Theory), in Polish. See English translation (1982) Topoi 2:7–52
14. Lin TY (2005) Granular computing: Examples, intuitions, and modeling. In: Proceedings of IEEE 2005 Conference on Granular Computing GrC05, Beijing, China. IEEE Press 40–44
15. Michalski RS, et al (1986) The multi–purpose incremental learning system AQ15 and its testing to three medical domains. In: Proceedings of AAAI-86, Morgan Kaufmann, San Mateo CA 1041–1045
16. Nguyen HS (1997) Discretization of Real Valued Attributes: Boolean Reasoning Approach, PhD Dissertation, Warsaw University, Department of Mathematics, Computer Science and Mechanics
17. Nguyen HS, Skowron A (1995) Quantization of real valued attributes: Rough set and Boolean reasoning approach, In: Proceedings 2^{nd} Annual Joint Conference on Information Sciences, Wrightsville Beach NC 34–37
18. Nguyen SH (2000) Regularity analysis and its applications in Data Mining, In: Polkowski L, Tsumoto S, Lin TY (eds.), Physica Verlag, Heidelberg 289–378
19. Pal S K, Dasgupta B, Mitra P (2004) Rough–SOM with fuzzy discretization. In: Pal SK, Polkowski L, Skowron A (eds.), Rough – Neural Computing. Techniques for Computing with Words. Springer, Berlin 351–372
20. Pawlak Z (1982) Rough sets, Int. J. Computer and Information Sci. 11:341–356
21. Pawlak Z (1991) Rough sets: Theoretical Aspects of Reasoning About Data. Kluwer, Dordrecht
22. Pawlak Z, Skowron A (1993) A rough set approach for decision rules generation. In: Proceedings of IJCAI'93 Workshop W12. The Management of Uncertainty in AI; also ICS Research Report 23/93, Warsaw University of Technology, Institute of Computer Science
23. Polkowski L (2002) Rough Sets. Mathematical Foundations, Physica Verlag, Heidelberg
24. Polkowski L (2004) Toward rough set foundations. Mereological approach. In: Proceedings RSCTC04, Uppsala, Sweden, Lecture Notes in AI 3066, Springer, Berlin 8–25

25. Polkowski L (2005) Formal granular calculi based on rough inclusions. In: Proceedings of IEEE 2005 Conference on Granular Computing GrC05, Beijing, China, IEEE Press 57–62
26. Polkowski L (2005) Rough–fuzzy–neurocomputing based on rough mereological calculus of granules, International Journal of Hybrid Intelligent Systems 2:91–108
27. Polkowski L (2006) A model of granular computing with applications. In: Proceedings of IEEE 2006 Conference on Granular Computing GrC06, Atlanta, USA. IEEE Press 9–16
28. Polkowski L, Artiemjew P (2007) On granular rough computing: Factoring classifiers through granular structures. In: Proceedings RSEISP'07, Warsaw, Lecture Notes in AI 4585, Springer, Berlin, 280–289
29. Polkowski L, Skowron A (1997) Rough mereology: a new paradigm for approximate reasoning, International Journal of Approximate Reasoning 15:333–365
30. Polkowski L, Skowron A (eds.) (1998) Rough Sets in Knowledge Discovery 2. Physica Verlag, Heidelberg
31. Polkowski L, Skowron A (1999) Towards an adaptive calculus of granules. In: Zadeh L A, Kacprzyk J (eds.) Computing with Words in Information/Intelligent Systems 1. Physica Verlag, Heidelberg 201–228
32. Semeniuk–Polkowska M (2007) On conjugate information systems: A proposition on how to learn concepts in humane sciences by means of rough set theory, Transactions on Rough Sets VI. Lecture Notes in Computer Science 4374:298–307, Springer, Berlin
33. Skowron A et al (1994) RSES: A system for data analysis.
Available: `http:\\logic.mimuw.edu.pl/~rses/`
34. Skowron A (1993) Boolean reasoning for decision rules generation. In: Komorowski J, Ras Z (eds.), Proceedings of ISMIS'93. Lecture Notes in AI 689:295–305. Springer, Berlin
35. Skowron A, Rauszer C (1992) The discernibility matrices and functions in decision systems. In: Słowiński R (ed) Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory. Kluwer, Dordrecht 311–362
36. Skowron A, Swiniarski RW (2004) Information granulation and pattern recognition. In: Pal S K, Polkowski L, Skowron A (eds.), Rough – Neural Computing. Techniques for Computing with Words. Springer, Berlin 599–636
37. Stefanowski J (2006) On combined classifiers, rule induction and rough sets, Transactions on Rough Sets VI. Lecture Notes in Computer Science 4374:329–350. Springer, Berlin
38. Stefanowski J, Tsoukias A (2001) Incomplete information tables and rough classification, Computational Intelligence 17:545–566
39. Swiniarski RW, Skowron A (2004) Independent component analysis, principal component analysis and rough sets in face recognition, Transactions on Rough Sets I. Lecture Notes in Computer Science 3100:392–404. Springer, Berlin
40. UCI Repository: `http://www.ics.uci.edu./~mlearn/databases/`
41. Wojdyłło P (2004) WaRS: A method for signal classification. In: Pal S K, Polkowski L, Skowron A (eds.), Rough – Neural Computing. Techniques for Computing with Words. Springer, Berlin 649–688
42. Wojna A (2005) Analogy–based reasoning in classifier construction, Transactions on Rough Sets IV. Lecture Notes in Computer Science 3700:277–374. Springer, Berlin

43. Wróblewski J (2004) Adaptive aspects of combining approximation spaces. In: Pal S K, Polkowski L, Skowron A (eds.), Rough – Neural Computing. Techniques for Computing with Words. Springer, Berlin 139–156
44. Zadeh LA (1979) Fuzzy sets and information granularity. In: Gupta M, Ragade R, Yaeger RR (eds.) Advances in Fuzzy Set Theory and Applications. North–Holland, Amsterdam 3–18