*Fundamenta Informaticae 30 (1997) 345–358*
*IOS Press*

# Decision Algorithms: a Survey of Rough Set - Theoretic Methods

**Andrzej Skowron**
*Institute of Mathematics*
*Warsaw University*
*Warsaw, Poland*
*e-mail: skowron@mimuw.edu.pl*

**Lech Polkowski**
*Institute of Mathematics*
*Warsaw University of Technology*
*Warsaw, Poland*
*e-mail:mltpolk@plwatu21.bitnet*

**Abstract.** In this paper we present some strategies for synthesis of decision algorithms studied by us. These strategies are used by systems of communicating agents and lead from the original (input) data table to a decision algorithm. The agents are working with parts of data and they compete for the decision algorithm with the best quality of object classification.

We give examples of techniques for searching for new features and we discuss some adaptive strategies based on the rough set approach for the construction of a decision algorithm from a data table. We also discuss a strategy of clustering by tolerance.

## 1. Introduction

The aim of synthesis of decision algorithms from a given decision system (data table) is to discover a set of decision rules from the knowledge about classified objects which will properly classify new (so far unseen) objects. The problem of extracting decision rules from experimental data is extensively studied (see [4], [5], [9], [10], [16], [18], [20], [23]). The rough set methods developed so far are not always sufficient for extracting laws from decision tables; one of the reasons is that these methods are not taking into account the fact that part of the reduct set is *chaotic* i.e. not stable with respect to randomly chosen samples of the decision table and the other is that the chosen features may not be adequate for object classification. The steps we will describe below leading from a given decision table to a decision algorithm are based on developments in rough set approach and they are used in a dynamic adaptive procedure better predisposed to recognize the chaotic nature of data and to classify objects.

We also discuss some aspects of a general cooperative intelligent agent system searching for adaptive strategies for decision algorithm synthesis.

Due to the high complexity of the decision algorithm synthesis from experimental data the problem has to be decomposed into some smaller tasks. For example this decomposition is done by considering a family of samples (subtables) of a given data table and applying to any sample a general fixed strategy for the synthesis. Any sample with a strategy is attached to an intelligent agent. During the process the agents communicate exchanging their findings (discoveries). This communication process can lead to extracting the relevant features for object classification [2], [4], [5]. We propose to base the mechanism of selecting best features on the so called dynamic reducts and rules. The communications between agents can lead, like in genetic algorithms [7], or other adaptive systems [10] to the elimination of some agents (producing decision algorithms with low quality of classification) from the existing population or adding some new agents to this population (born from the most promising agents).

We concentrate in our discussion on examples of new strategies based on the rough set approach which lead to improvement in the quality of object classifications or are recently investigated for this purpose and now are implemented and tested. Any agent acts according to the following general scheme:

**step 0**. Reading a data table: $\mathbb{A} := \mathit{INPUT}\,()$.

**step 1**. Transforming the table $\mathbb{A}$ and messages $\mathbb{M}$ communicated by other agents (concerning data filtration and/or feature extraction) into a new table:

$$\mathbb{A} := F(\mathbb{A}, \mathbb{M})$$

where $F$ is a composition $F_2 \circ F_1$ of a preprocessing operator $F_1$ (performing data filtration) and a feature extraction operator $F_2$ (providing the list of features for object classification).
In this step some other tasks are also performed like:

(i) storing the discoveries of the agent;

(ii) exchanging messages with other agents;

(iii) voting on creation of new agents and/or on elimination of some others.

**step 2**. Synthesis of a decision algorithm $D$ on the basis of strategies receiving as input: the data table $\mathbb{A}$, information about the agent discoveries and messages $\mathbb{M}$ communicated by other agents concerning their current findings about local decision algorithms including their quality of classification.

In this step also some other tasks are also performed like (i), (iii) mentioned in *step 1*.

**step 3**. Reading new data:

$$\mathbb{B} := \mathit{INPUT}\,()$$

(New examples are stored in a data table $\mathbb{B}$).

**step 4**. Testing $D$ on new data $\mathbb{B}$:

$$b := \mathit{TEST}\,(D, \mathbb{B})$$

where $b$ is a variable representing the quality of classification. Merging tables $\mathbb{A}$ and $\mathbb{B} : \mathbb{A} := Merge(\mathbb{A}, \mathbb{B})$.
In this step also some other tasks are also performed like (i)-(iii) mentioned in *step 1*.

**step 5**. **if** $b <$*threshold* **then go to** step 2 (or *step 1* when e.g. several attempts to obtain a better classification quality by going back to *step 2* failed) where *threshold* represents a prescribed acceptance level of the classification quality.

**step 6**. **go to** *step 3*

In the sequel we discuss some examples of specific solutions based on the rough set approach in the above general scheme (assuming the family of samples (subtables) to be fixed).

## 2. Preprocessing — Data Filtration

The preprocessing of the input data table consists in data filtration according to the techniques of analytical morphology [19], [21]. We recall here very briefly the main lines of this approach. Nevertheless we assume that the reader is familiar with the basic notions of the rough set theory [13], [20].

Now we recall the definition of decision rules. Let $\mathbb{A} = (U, A \cup \{d\})$ be a decision table and let $V = \bigcup\{V_a : a \in A\} \cup V_d$. The atomic formulas over $B \subseteq A \cup \{d\}$ and $V$ are expressions of the form $a = v$, called *descriptors* over $B$ and $V$, where $a \in B$ and $v \in V_a$. The set $\mathcal{F}(B, V)$ of formulas over $B$ and $V$ is the least set containing all atomic formulas over $B$ and $V$ and closed with respect to the classical propositional connectives $\vee$ (disjunction), $\wedge$ (conjunction). Let $\tau \in \mathcal{F}(B, V)$. Then by $\tau_{\mathbb{A}}$ we denote the meaning of $\tau$ in the decision table $\mathbb{A}$, i.e. the set of all objects in $U$ with property $\tau$, defined inductively as follows:

1. if $\tau$ is of the form $a = v$ then $\tau_{\mathbb{A}} = \{u \in U : a(u) = v\}$;
2. $(\tau \wedge \tau')_{\mathbb{A}} = \tau_{\mathbb{A}} \cap \tau'_{\mathbb{A}}$; $(\tau \vee \tau')_{\mathbb{A}} = \tau_{\mathbb{A}} \cup \tau'_{\mathbb{A}}$.

The set $\mathcal{F}(A, V)$ is called the set of *conditional formulas of* $\mathbb{A}$.

A *decision rule* of $\mathbb{A}$ is any expression of the form

$$\tau \Longrightarrow d = v \quad \text{where} \quad \tau \in \mathcal{F}(A, V) \quad \text{and} \quad v \in V_d.$$

The decision rule $\tau \Longrightarrow d = v$ of $\mathbb{A}$ is *true in* $\mathbb{A}$, symbolically $\tau \Longrightarrow_{\mathbb{A}} d = v$, iff $\tau_{\mathbb{A}} \subseteq (d = v)_{\mathbb{A}}$; if $\tau_{\mathbb{A}} = (d = v)_{\mathbb{A}}$ then we say that the rule is $\mathbb{A}$-*exact*. If $\mathbb{A} = (U, A \cup \{d\})$ is a decision table then by $\mathbb{A}_{\partial}$ we denote the decision table $(U, A \cup \{d\})$ where $\partial = \partial_A$. Let us observe that any decision rule $\tau \Longrightarrow \partial_A = \theta$ where $\tau \in \mathcal{F}(A, V)$ and $\phi \neq \theta \subseteq V_d$ valid in $\mathbb{A}_{\partial}$ and having examples in $\mathbb{A}$, i.e. satisfying $\tau_{\mathbb{A}} \neq \phi$ determines a distribution of objects satisfying $\tau$ among elements of $\theta$. This distribution is defined by

$$\mu_i(\mathbb{A}, \tau, \theta) = \frac{|Y \cap X_i|}{|Y|}, \quad \text{for} \quad i \in \theta$$

where $Y = \tau_{\mathbb{A}}$. Any decision rule $\tau \Longrightarrow \partial_A = \theta$ valid in $\mathbb{A}_{\partial}$ and having examples in $\mathbb{A}$ with $|\theta| > 1$ is called *non-deterministic*, otherwise it is *deterministic*. The number $n(\mathbb{A}, \tau, i) = |\tau_{\mathbb{A}} \cap X_i|$ is called the *number of examples supporting* $\tau$ *in the* $i$-th *decision class* $X_i$. Let us observe that for any formula $\tau$ over $A$ and $V$ with $\tau_{\mathbb{A}} \neq \phi$ there exists exactly one subset $\theta_\tau$ of $\Theta$ such that $\tau \Longrightarrow_{\mathbb{A}} \partial_A = \theta_\tau$ and $\mu_i(\mathbb{A}, \tau, \theta_\tau) > 0$ for any $i \in \theta_\tau$. In the sequel, we write $\mu_i(\mathbb{A}, \tau)$ instead of $\mu_i(\mathbb{A}, \tau, \theta_\tau)$.

Let $k$ be a real number from the interval $(0, 1]$. Let $\mathbb{A} = (U, A \cup \{d\})$ be a decision table where $A = \{a_1, \ldots, a_m\}$ and let $\tau$ be a formula over $A$ and $V$. If $\alpha = \{(a_1, v_1), \ldots, (a_m, v_m)\}$, then $\bigwedge \alpha$ denotes the formula $(a_1 = v_1) \wedge \ldots \wedge (a_m = v_m)$. If $\tau$ is a formula over $\mathbb{A}$ and $V$ then by $\mathbb{A}_\tau$ we denote the restriction of $\mathbb{A}$ to the set of all objects from $U$ satisfying $\tau$, i.e. $\mathbb{A}_\tau = (\tau_{\mathbb{A}}, A \cup \{d\})$. By $T(\mathbb{A}, \tau, k)$ we denote the conjunction of the following conditions:

(i) $\tau_{\mathbb{A}} \neq \emptyset$

(ii) for any $\alpha \in INF(\mathbb{A}_\tau)$:

  $(*)$ $\max_i \mu_i(\mathbb{A}, \bigwedge \alpha \wedge \tau) > k$

  and

  $(**)$ there exists exactly one $i_o$ with the following property:
  $\mu_{i_o}(\mathbb{A}, \bigwedge \alpha \wedge \tau) = \max_i \mu_i(\mathbb{A}, \bigwedge \alpha \wedge \tau)$.

For any formula $\tau$ over $A$ and $V$ and a threshold $k$ satisfying the condition $T(\mathbb{A}, \tau, k)$ we define a $\tau$-*approximation function in* $\mathbb{A}$ *with threshold* $k$

$$F(\mathbb{A}, \tau, k) : INF(\mathbb{A}_\tau)|B \longrightarrow INF(\{d\}, \tau_{\mathbb{A}}, V)$$

by $F(\mathbb{A},\tau,k)(\alpha) = \{(u,i_0) : u \in \tau_{\mathbb{A}}\}$ for any $\alpha \in \mathit{INF}(\mathbb{A}_\tau)|B$, where $\mu_{i_0}(\mathbb{A},\bigwedge \alpha \wedge \tau) = \max\{\mu_i(\mathbb{A},\bigwedge \alpha \wedge \tau) : i \in \Theta_\tau\}$ and $B$ is the set of conditions occurring in $\tau$.

In the choice of approximation functions one can also take into account the *critical level of examples* $l$ by demanding that

$$\max_i n(\mathbb{A},\bigwedge \alpha \wedge \tau, i) > l$$

for $\alpha \in \mathit{INF}(\mathbb{A}_\tau)|B$ where $B$ is the set of conditions occuring in $\tau$ and $n(\mathbb{A},\bigwedge \alpha \wedge \tau, i)$ is the number of examples satisfying the formula $\bigwedge \alpha \wedge \tau$ and having the decision value $i$.

The data filtration process is performed with a set $\mathcal{F}$ of approximation functions. The particular problem is how to choose a proper subset of approximation functions from a given set $\mathcal{F}$ of approximation functions and in what order to apply them to get the proper filtration of $\mathbb{A}$. To solve this problem we apply genetic algorithms. The other, and prior, problem is to determine the set $\mathcal{F}$. Here the following procedure can be applied.

For any $a \in A' \subseteq A$, where $A'$ is a randomly chosen sample set of conditions, apply the methods for decision rules synthesis (see [14], [20]) for the decision table $\mathbb{A}_a = (U, \{A - A'\}) \cup \{a\})$ (in particular apply also for synthesis of rules the so called $\varepsilon$-reducts [4], [19], [20]). The output of this step is a set of decision rules of the form:

$$\tau \Longrightarrow \partial_{A-A'} = \Theta_\tau$$

where $\partial_{A-A'}$ is the generalized decision corresponding to the condition $a \in A'$ and $\tau$ is a formula over $A - A'$ and $V$. Now some strategies should be applied to get from these decision rules a “global” decision rule for $a$

$$\tau_0 \Longrightarrow \partial_{A-A'} = \Theta$$

such that $T(\mathbb{A}_a, \tau_0, k)$ holds and $\max_i n(\mathbb{A}_a, \bigwedge \alpha \wedge \tau_0, i) > l$ for $\alpha \in \mathit{INF}((\mathbb{A}_a)_{\tau_0}|B)$ where $B$ is the set of conditions occuring in $\tau_0$ and $(\mathbb{A}_a)_{\tau_0}$ is the restriction of $\mathbb{A}_a$ to the set of objects satisfying $\tau_0$.

From the global decision rules for conditions in $\mathbb{A}$ the approximation functions are built. In this way we obtain a set $\mathcal{F}$ of approximation functions.

One can distinguish in the above procedure the two strategies. The first one, say $S$, produces from an actual decision table $\mathbb{A}$ for any $a \in A'$ a sequence of decision rules of the following form:

$$\tau_1 \Longrightarrow \partial_{A-\{a\}} = \Theta_1, \ldots, \tau_p \Longrightarrow \partial_{A-\{a\}} = \Theta_p$$

satisfying $T(\mathbb{A}_a, \tau_i, k)$ for $i = 1, \ldots, p$.

These rules are next used to generate approximation functions by the second strategy, say $H$. The strategy $H$ produces from the above sequences a new sequence

$$(s_1, \ldots, s_r)$$

where $s_i$ is a set of non—conflicting decision rules of the form $\tau \Longrightarrow \partial_{A-\{a\}} = \Theta$ for some $a \in A$, $\Theta \subseteq V_d$ and formula $\tau$ over $A - \{a\}$ and $V$. The approximation functions corresponding to the decision rules from $s_i$ are chosen by $H$ for simultaneous application at the $i$-the step of transformation of the actual decision table $\mathbb{A}$.

In this way our procedure has parameters $S, H, \mathbb{A}, \delta$ (by assumption $l$ and $k$ are fixed). The proper values for $k$ and $l$ should be chosen by making experiments with $\mathbb{A}$.

One may not expect that the solutions $S$ and $H$ for the above filtration problems are of polynomial time complexity with respect to the size of $\mathbb{A}$ and $\delta$ because the strategies $S$ and $H$ are based, e.g. on procedures for decision rules generation and these procedures are based on the reduct set generation [20]. Nevertheless, one can build efficient heuristics for solving the filtration problems for practical applications.

We apply the above construction to decision tables derived from a given decision table $\mathbb{A} = (U, A \cup \{d\})$. These decision tables are of the form $\mathbb{B} = (U, B \cup \{c\})$ and they are constructed from information systems $\mathbb{B} = (U, B \cup C)$, where $B, C \subseteq A$ and $B \cap C = \emptyset$ by representing $C$ by means of one decision attribute $c$.

The result of the competitive process of analytic filtration of data is a new data table $\mathbb{A}' = (U, A' \cup \{d\})$ where $A' = \{a'_1, \ldots, a'_m\}$ and $V_{a'_i} \subseteq V_{a_i}$ for $i = 1, \ldots, m$. For $\delta \in (0, 1]$, we say that $\mathbb{A}'$ is a *$\delta$-filtration of* $\mathbb{A}$ iff

(i) $\partial_A = \partial_{A'}$
(ii) $|\{[u]_{A'} : [u]_{A'} \subseteq Z_\Theta\}| < \delta |\{[u]_A : [u]_A \subseteq Z_\Theta\}|$

for any $\Theta \subseteq V_d$ with $\emptyset \neq Z_\Theta = \partial_A^{-1}(\Theta)$.
One can also consider another version of this definition with condition (ii) substituted by a weaker condition specifying that $\partial_A$ and $\partial_{A'}$ should be sufficiently close with respect to some distance function e.g. some iterative tolerance metric.

The process of defining the new attributes $a'_1, \ldots, a'_m$ by analytic filtration can be regarded as a specific instance of the general process of feature extraction. The analytic filtration preprocessing leads to new features about which one may assume that they are more robust i.e. noise — resistant then the original (measurable) attributes. The preprocessed data table $\mathbb{A}'$ undergoes in turn the process of feature extraction by dedicated techniques discussed shortly below.

## 3. Feature Extraction

We may define features as functions (attributes) on objects derived from the existing attributes. In this respect one may realize that a given data table presents not only a small fragment of the reality as it classifies a tiny fraction of objects but also it employs a tiny fraction of possible attributes. The purpose of feature extraction is to obtain a set of attributes with better classifying properties with respect to new objects. An important criterion for quality of feature extraction is the reduction of dimensionality and size of the classification space. Our perceiving of the data structure determines the set of possible features. In this set we look for the relevant features. We would like to point out to the fact that the process of synthesis of adaptive decision algorithms should allow for adaptive search for proper (from the classification point of view) representation of object structure (knowledge representation). For example, searching for a proper representation of structure in the logical framework requires finding a proper syntax and semantics of a logical language. We discuss below applications of multi-modal logics to this problem [5], [23]. There are many feature extracting techniques which we divide tentatively into the following groups:

- feature extraction by means of logical languages,
- feature extraction by means of dependencies discovery,
- feature extraction by clustering,
- feature extraction by statistical tools,
- feature extraction by topological methods,
- feature extraction by morphological filtering,
- feature extraction by algebraic, difference or differential equations,
- feature extraction by algebraic or integral transforms,
- feature extraction by syntactic and linguistic methods.

We discuss here examples of these techniques constructed with application of rough set methods. The description of the others one can find e.g. in [5], [11], [20], [23].

Any process of feature extraction applied to the preprocessed data table $\mathbb{A}'$ results in a new information system $\mathbb{A}'' = (U, A'')$. From this new information system a decision algorithm is constructed by applying some strategies.

## 3.1. Feature extraction by discovery of dependencies

This technique consists in finding near-to-functional relationships in data tables and is discussed in [21].

One can observe that the approximation functions applied for filtration of decision tables can also be used in searching for new features. Let us assume that our system is searching for the approximation functions

$$F : \mathit{INF}(\mathbb{A})|B \longrightarrow \mathit{INF}(C, V)$$

with the following property: if $F(u) = v$ where $u$ and $v$ are pieces of the information about a classified object then there is a strong evidence (measured e.g. by some threshold $k$) that the object belongs to a distinguished set of decision classes. If it is possible to discover this kind of approximation function then one can add as a new condition (feature, classifier) to the decision table the binary attribute $a_F$ defined by:

$$a_F(x) = 1 \quad \text{iff} \quad F(\{(a, a(x)) : a \in B\}) = \{(a, a(x)) : a \in C\}$$

Adding to the decision table several features of the above form, distinguishing a particular set of decision classes with sufficiently large evidence (related to the value of the threshold $k$) one can expect to get efficient classification mechanism. We implement now this kind of mechanism for feature extraction.

In [21], we presented a detailed introduction to Mathematical Morphology and a higher-level version of Mathematical Morphology called Analytical Morphology, aimed at filtering data tables without any apriorical geometric structure. The important difference which makes Analytical Morphology a high-level abstraction of standard morphology is that analytical morphology filters relations over subsets of a geometric set modifying these relations with preserving the geometric support instead of modifying the supporting set.

## 3.2. Feature extraction by clustering

Clustering may be defined informally as an attempt at imposing a distance measure (function) $d$ on objects of some collection $U$ in such a way that the collection $U$ can be represented as a union $C_1 \cup C_2 \cup \ldots \cup C_k$ of subcollections $C_i \subseteq U$, $i = 1, 2, \ldots, k$, forming clusters i.e. $d$-distances among objects within any cluster $C_i$ are relatively small compared to distances between distinct clusters. The vague nature of this informal description reflects the ambiguity inherent to clustering: as a rule, neither the distance function nor the size of clusters nor the degree to which they may overlap are defined clearly; they are subject to some judicious choice. This choice can be based on different criteria: among techniques for clustering one may distinguish the main three groups viz. 1. based on minimization of weighted least squares functionals 2. based on hierarchical clustering and 3. based on graph — theoretical methods [3].

No matter which technique we do apply for clustering, it is important to realize that we interpret clusters as collections of objects similar to one another, while objects from distinct clusters are perceived by us as not being similar. Clusters therefore define a similarity relation on objects from the given collection; it is seldom that this similarity has more properties besides reflexivity and symmetry; clusters form a covering of the universe with intricate overlapping relationships as there is a trade-off between the crispness of clustering and the adaptive quality of clusters-based decision algorithms.
This cluster-defined similarity is therefore a tolerance relation [3], [22]. We may restrict ourselves to the simplest case in which a clustering $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ on the universe $U$ determines a tolerance relation $\tau_{\mathcal{C}}$ on $U$ by the condition

$$x \tau_{\mathcal{C}} y \quad \text{iff} \quad x, y \in C_i \quad \text{for some} \quad i = 1, 2, \ldots, k.$$

It is the important property of any tolerance relation $\tau$ that it determines a metric (distance function) $d_\tau$ on $U$ by the formula

$$d_\tau(x,y) = \min\{n : \text{ there exists a sequence } x_0, x_1, \ldots, x_n \text{ such that } x_0 = x, x_n = y \text{ and } x_i \tau x_{i+1}\} \text{ in case } x \neq y \text{ and } d_\tau(x,x) = 0.$$

We will call $d_\tau$ the *tolerance iterate metric.*
Clusters $C_i$ are therefore defined metrically by the conditions:

(i) $d_{\tau_C}$-*diameter* $C_i = 1$ for any $i = 1, 2, \ldots, k$ and
(ii) $d_{\tau_C}$-$dist(C_i, C_j) = \min\{\{\max dist(x, C_j) : x \in C_i\}, \max\{dist(x, C_i) : x \in C_j\}\} \geq 2$ for any $i \neq j$.

We obtain the following conclusion:
*any clustering $\mathcal{C}$ on a set $U$ of objects can be represented in the form $\mathcal{C}(\tau, k, m)$ where $\tau$ is a tolerance on $U, k, m$ are natural numbers with $1 \leq k < m$ and*

(iii) $d_\tau$-*diameter* $C_i \leq k$ for any $C_i \in \mathcal{C}$;
(iv) $d_\tau$-*dist* $(C_i, C_j) \geq m$ for any pair $C_i, C_j \in \mathcal{C}$.

The above observation is the cornerstone of our clustering strategy: we will search for tolerance relations $\tau$ whose iterate metric will satisfy a) and b) Actually, we will search for tolerance relations $\tau$ satisfying a stronger condition $b'$) viz. for some $m$

(iv)' $d_\tau$-*Dist*$(C_i, C_j) = \min\{d_\tau(x,y) : x \in C_i, y \in C_j)\} \geq m$
for any pair $C_i, C_j \in \mathcal{C}$.

Finding such a tolerance $\tau$ would mean that we have found a similarity of objects relation whose iterates up to $k$ separate objects into certain attractors (clusters) while iterates of higher degrees separate the clusters.

We discuss below the two basic problems into which the clustering by tolerance problem splits: the problem of clustering with a given tolerance relation as well as the problem of searching for a proper tolerance relation.

## 3.3 Clustering with a fixed distance function (tolerance):

Let us observe that the least squares functional clustering [11] does not attempt at reflecting the dependencies among objects as it is based on an objective functional and therefore it can be of use rather for clustering within any given decision class then for clustering of objects globally. Other clustering methods [11] like clustering based on hierarchical merging [11] are also oriented towards imposing a structure on objects rather than reflecting an inherent structure in the set of objects or are also based on an objective global distance function [11].

Now we will discuss problems concerning clustering when the distance function $d$ is a tolerance iterate metric $d_\tau$ for a given tolerance relation $\tau$.

We assume that a fixed tolerance $\tau$ is given hence the iterate distance function $d_\tau$ is fixed as the distance function with respect to which we are going to discuss our clustering strategy. The clustering strategy $\mathcal{C}$ will call some parameters which we are going to discuss briefly.

## 3.4 Preprocessing: the training example set size

In this problem we would like to apply minimal absorbent sets. Consider a decision table $\mathbb{A} = (U, A \cup \{d\})$; the generalized decision $\partial_\mathbb{A}$ introduces on the universe $U$ a *generalized decision tolerance* $\tau_{\partial_\mathbb{A}}$ defined as follows:

$$x \tau_{\partial_\mathbb{A}} y \quad \text{if and only if} \quad \partial_\mathbb{A}(x) \cap \partial_\mathbb{A}(y) \neq \emptyset.$$

Objects $x$ and $y$ are therefore tolerant if and only if their generalized decisions intersect.

## 3.5 The training set of examples

For any object $x \in U$, we denote by $\tau_{\partial_\mathbb{A}}(x)$ the tolerance set of $x$ with respect to the tolerance $\tau_{\partial_\mathbb{A}}$. We apply to the family $\{\tau_{\partial_\mathbb{A}}(x) : x \in U\}$ the procedure of generating minimal aborbent sets [22] i.e. minimal sets of objects such that any other element is at a distance at most one from this set. This procedure [22] can be built by applying boolean reasoning [6]. Let $MIN\_ABS(\tau_{\partial_{\mathbb{A}}})$ be the family of all these absorbents. Our first parameter will be a set $X \in MIN\_ABS(\tau_{\partial_{\mathbb{A}}})$. The set $X$ will be a training set of examples on which a clustering strategy will be tested; observe that by definition of an absorbent set, for any object $x \in U$ there exists an object $y \in X$ with $x\tau_{\partial_{\mathbb{A}}}y$ i.e. any object is at the distance at most 1 from $X$ with respect to the iterate distance function $d_{\tau_{\partial_{\mathbb{A}}}}$.

## 3.6 Iteration number scaling

Given a training set $X$, we are going to apply a clustering with respect to the fixed distance $d_\tau$.

There are two aspects of the iteration number scaling problem:

- crisp or fuzzy scaling
- uniform or non-uniform scaling.

The scaling problem concerns the *iterate interval* $I = [1, N]$ where $N$ is the *iterate upper bound* (see the next section). Setting $I$ means that we are assuming that the clusters will be determined by iterates $\tau^k$ of $\tau$ with $k \leq N$.

Our problem now is to decide how to scale the iterate interval $I$: we have to decide about the number of iterations which we regard as e.g. *small* i.e. the objects at this distance we will regard as close, repectively the number of iterates we regard as *medium, large*, etc.

This amounts to selecting a covering $\mathcal{U}$ of $I$ : $\mathcal{U} = \{I_1, I_2, \ldots, I_k\}$ where each $I_j$ is a sub-interval of $I$. We have two cases.

**Case 1.** The case of crisp scaling. In this case the covering $\mathcal{U}$ is a partition of the interval $I$ into disjoint sub-intervals $I_1, I_2, \ldots, I_k$. To each sub-interval $I_j$ a label $\alpha_j$ is attached indicating the degree of closeness of objects whose distance value belongs to $I_j$.

**Case 2.** The case of fuzzy scaling. In this case the covering $\mathcal{U}$ consists of overlapping intervals $I_1, I_2, \ldots, I_k$. To each sub-interval $I_j$ a label $\alpha_j$ is attached indicating the degree of closeness of objects whose distance value belongs to $I_j$. For any interval $I_j$, the fuzzy membership function $\mu_j$ is assigned i.e. for any $k \in I$, the value $\mu_j(k)$ indicates the degree in which $k$ belongs to $I_j$. Once we decide on the scaling the interval $I$, we have to decide whether our scaling of iterations be uniform or non-uniform.

**Case 3.** The uniform scaling. In this case we select a scaling according to either Case 1 or Case 2 and apply it at any element of the training set.

**Case 4.** The non-uniform scaling. In this case, we partition the training set $X$ into disjoint subsets, say, $X_1, X_2, \ldots, X_q$. For any subset $X_i$, we select a pair ($\mathcal{U}^i = \{I^i_1, I^i_2, \ldots, I^i_{k_i}\}$, $\alpha^i = \{\alpha^i_1, \alpha^i_2, \alpha^i_{k_i}\}$) according to Case 1 or a triple ($\mathcal{U}^i = \{I^i_1, I^i_2, \ldots, I^i_{k_i}\}$, $\alpha^i = \{\alpha^i_1, \alpha^i_2, \alpha^i_{k_i}\}$, $\mu^i = \{\mu^i_1, \mu^i_2, \ldots, \mu^i_{k_i}\}$) according to Case 2.

We obtain in this way a separate scaling for any subset $X_i$ of the training set $X$.

## 3.7 Bounds on iteration number

The iteration upper bound $N$ has to be decided on the basis of experiments with various training sets.

## 3.8 Adaptive strategies based on tolerance reducts

One can apply strategies based on dynamic reducts and rules (see the next section) to the case of tolerance and relative tolerance reducts [20] to define a dynamic counterpart of these notions.

## 3.9 Clustering quality

Clustering quality will be evaluated in a twofold perspective: from the structural perspective and from the adaptive perspective.

## 3.10 Structure quality evaluation

Structure quality is evaluated on the basis of the following parameters:

- minimal cluster size
- minimal distance between clusters
- maximal cluster diameter
- number of clusters.

### 3.11 Adaptive quality evaluation

Adaptive quality is evaluated on the basis of performance of the decision algorithm based on the new decision table constructed by means of feature extraction by clustering.

### 3.12 Searching for a proper tolerance (interactive, semi-automatic, automatic)

The searching for the proper tolerance is a crucial and the most difficult task related to decision algorithm synthesis based on tolerance. Let us restrict our considerations to the tolerances defined by weighted distances between information vectors $u, v$:

$$d(u,v) = \sum_i w_i |u_i - v_i|$$

where $w_i$ is a weight of the i-th attribute and $w_i \geq 0$.

Because of high computational complexity of the searching problem for the proper weights one should apply some heuristics. They can use the ideas of e.g. simulated annealing [1], genetic algorithms [7] or neural networks.

We sketch here the basic idea of the simulated annealing approach this problem. Any choice of weights $\{w_i\}$ determines the state of the system. The transition relation can be defined by choosing a local perturbation of weights. The resulting new state (the new tolerance relation) is tested. This testing stage consists of applying the clustering strategy based on the new tolerance for synthesis of decision algorithm which is next tested on unseen objects. If the test results are satisfactory then the new state received by the local perturbation of weights is evaluated as better than the previous state and is taken as the current state of the process of simulated annealing. If the test results are not satisfactory then the new state is selected as a current state with probability given by the Boltzmann distribution. Iterating the process, we determine a sequence of states that is a sequence of tolerance relations. One can experimentally choose the number of iterations. Once the iteration process is completed, we decrease (slightly) the control parameter (corresponding to the temperature) and repeat the iteration. This procedure is repeated until the chosen in advance stop criterion is reached. The obtained final tolerance relation is taken as the best possible tolerance. Clearly, much more work should be done to apply this idea for specific data tables. Although the scheme is universal, the choice of specific parameters of the simulated annealing process will depend on the particular data table and these parameters should be tuned to fit any concrete case.

In the research project on adaptive searching for a proper tolerance we also plan to apply genetic algorithms and neural networks.

## 4. Adaptive Strategies for Decision Algorithm Construction from Information Systems

Given a new information system (decision table) $\mathbb{A}''$, one applies to $\mathbb{A}''$ some strategies to produce decision rules. The primary technique offered by the rough set theory has been here the reduct generation. Reducts offer the same classificational ability as the whole system and with smaller set of attributes. However, as already stressed this approach does not suffice and is implemented by additional tools. We discuss examples of strategies based on [20]:

- reduct approximation,

- dynamic reducts and rules,
- voting strategies,
- on boundary region thinning.

We will discuss below these topics in more detail.

## 4.1. Approximation of reducts

One group of methods for extracting laws from decision tables can be obtained on the basis of two notions, namely the discernibility matrix [20] and the reduct approximation [20]. Several strategies for searching for a subset of the set of discernibility matrix entries sufficient for the generation of laws encoded in the decision table are implemented in our system for objects classification.

One of the techniques for the reduct approximation can be based on the positive region approximation. The following algorithm computes this kind of the reduct approximation.

**Algorithm**

For $R \in RED(\mathbb{A}, d)$:

```
step 1: calculate positive regions $POS_{R-\{a\}}$ for all $a \in R$.
step 2: choose one attribute $a_0$ from the reduct $R$ satisfying the condition:
    $\forall a \in R \;\; POS_{R-\{a_0\}} \geq POS_{R-\{a\}}$.
step 3: if $POS_{R-\{a_0\}} > k \cdot N$ (e.g. $k = 0.9$) then
    begin
        $R := R - \{a_0\}$
        go to step 1
    end
step 4: now, we received a new set of attributes ($R$) called the S-reduct.
```

$S$-reducts can help to extract interesting laws from decision tables. Applying reduct approximation instead of reducts we slightly decrease the quality of classification of objects from training set but we expect to receive more general rules with the higher quality of the classification for new objects.

## 4.2. Boundary region thinning

Objects are classified on the basis of information about them. In a given table, we associate with any information vector the distribution of objects corresponding to this vector into decision classes. When this probability distribution is non - uniform, we can regard objects corresponding to small values of this distribution as in a sense abnormal or noisy objects. The generalized decision for a given information vector can be then modified by removing from it the decision values corresponding to these small values of the probability distribution. The decision rules generated for the modified generalized decision can give a better quality of classification of new yet unseen objects. Various techniques of this approach called boundary region thinning have been proposed [20], [24]. Boundary region thinning gives a new decision table to which the discussed by us methods of synthesis of decision rules can be applied.

## 4.3. Dynamic reducts and rules

We now show an example of communication among cooperating agents working on synthesis of adaptive decision algorithms. In this example, the information sharing among agents leads to extraction, from the subtables processed by agents of the most stable reducts called *dynamic reducts.*

The underlying idea of dynamic reducts stems from the observation that reducts generated from the information system are not stable in the sense that they are sensitive to changes in the information system introduced by removing a randomly chosen set of objects. The notion of a dynamic reduct encompasses the stable reducts i.e. reducts that are the most frequent reducts in random samples created by subtables of the given decision table [4], [20]. We show here how to compute dynamic reducts from reduct approximations and how to generate dynamic rules from dynamic reducts. The dynamic reducts have shown their utility in various experiments with data sets of various kinds e.g. market data [8], monk's problems [4] or handwritten digits recognition [5], [23]. The quality of unseen objects classification by decision rules generated from dynamic reducts increases especially when the data are very noisy as is the case e.g. with market data [8].

To capture the fact that some reducts are chaotic, we consider random samples forming subtables of a given decision table $\mathbb{A} = (U, A \cup \{d\})$; we will call a *subtable* of $\mathbb{A}$ any information system $\mathbb{B} = (U', A \cup \{d\})$ such that $U' \subseteq U$.

For a family $\mathcal{F}$ of subtables of $\mathbb{A}$, we denote by $DR(\mathbb{A}, \mathcal{F})$ the set $RED(\mathbb{A}, d) \cap \bigcap_{\mathbb{B} \in \mathcal{F}} RED(\mathbb{B}, d)$; elements of the set $DR(\mathbb{A}, \mathcal{F})$ are called *$\mathcal{F}$-dynamic reducts* of $\mathbb{A}$.

It follows immediately from the definition of a dynamic reduct that a relative reduct of $\mathbb{A}$ is an $\mathcal{F}$-dynamic reduct iff it is a relative reduct of any subtable from $\mathcal{F}$.

The notion of a relative reduct may still be a little restrictive; to make it a bit more flexible, we introduce a weaker notion of an $(\mathcal{F}, \varepsilon)$-dynamic reduct with $\varepsilon$ a real number from the unit interval $[0, 1]$.

The set $DR_\varepsilon(\mathbb{A}, \mathcal{F})$ of *$(\mathcal{F}, \varepsilon)$-dynamic reducts* is defined by

$$DR_\varepsilon(\mathbb{A}, \mathcal{F}) = \left\{C \in RED(\mathbb{A}, d) : \frac{|\{\mathbb{B} \in \mathcal{F} : C \in RED(\mathbb{B}, d)\}|}{|\mathcal{F}|} \geq 1 - \varepsilon\right\}$$

The following properties of $(\mathcal{F}, \varepsilon)$-dynamic reducts follow from the definition

**Proposition 4.1.**

(i) If $\mathcal{F} = \{\mathbb{A}\}$, then $DR(\mathbb{A}, \mathcal{F}) = RED(\mathbb{A}, d)$;
(ii) If $\mathcal{F} \subseteq \mathcal{F}'$, then $DR_\varepsilon(\mathbb{A}, \mathcal{F}) \subseteq DR_\varepsilon(\mathbb{A}, \mathcal{F}')$;
(iii) If $\varepsilon \leq \varepsilon'$, then $DR_\varepsilon(\mathbb{A}, \mathcal{F}) \subseteq DR_{\varepsilon'}(\mathbb{A}, \mathcal{F})$;
(iv) $DR(\mathbb{A}, \mathcal{F}) \subseteq DR_\varepsilon(\mathbb{A}, \mathcal{F})$ for any $\varepsilon \geq 0$;
(v) $DR(\mathbb{A}, \mathcal{F}) = DR_0(\mathbb{A}, \mathcal{F})$; □

For $C \in RED(\mathbb{B}, d)$, the number $1 - inf\{\varepsilon \geq 0 : C \in DR_\varepsilon(\mathbb{A}, \mathcal{F})\}$ is called the *stability coefficient* of $C$ *relative* to $\mathcal{F}$.

### 4.4 Methods and techniques for dynamic reduct computation

The processes of decision rules generation based on reduct sets have high computational complexity: the problem of computing a minimal reduct is NP-hard [20] and therefore we are forced to apply some approximation algorithms to obtain knowledge about reduct sets.

One possibility is to use approximation algorithms that do not give an optimal solution but use short computing time e.g. algorithms based on simulated annealing and Boltzmann machines, genetic algorithms and algorithms using neural networks. We plan to use these algorithms in experiments for generation of a large number of reducts. The other possibility is to use standard computational techniques but on modified information systems. Here we can pursue the following courses of action:

- conceptual clustering of values of attributes or groups of attributes
- conceptual clustering of objects
- extracting new attributes from existing decision tables.

## 4.5 Reducts of subtables

We present one technique for computing the dynamic reducts [4], [20]. The experiments with different data sets have shown that this type of dynamic reducts allows to generate decision rules with better quality of classification of new objects than the other methods. The method consists in the following.

step 1: a random set of subtables is taken from the given table; for example:

10 samples with the size of 90% of the decision table,
10 samples with the size of 80% of the decision table,
10 samples with the size of 70% of the decision table,
10 samples with the size of 60% of the decision table,
10 samples with the size of 50% of the decision table.

step 2: reducts for all of these tables are calculated; for example reducts for any of the 50 randomly chosen tables.
step 3: reducts with the stability coefficients higher than a fixed threshold are extracted.

These reducts selected in step 3 are regarded as true dynamic reducts.

## 4.6 Reduct approximation techniques (S-reducts)

One can compute dynamic reducts using approximations of reducts instead of reducts to generate dynamic reducts.

## 4.7 Decision rules analysis

If a set of dynamic reducts (with the stability coefficients greater than a given threshold) has been computed then it is necessary to decide how to compute the set of decision rules.

We have implemented several methods. The first one is based on the $(F, \varepsilon)$-dynamic core of $A$, i.e. on the set $\bigcup DR_{\varepsilon}(\mathbb{A}, \mathcal{F})$. We apply the methods based on boolean reasoning [6] presented in [14], [20] to generate decision rules (with minimal number of descriptors) from conditional attributes belonging to the dynamic core. The second one is based on the decision rule set construction for any chosen dynamic reduct. The final decision rule set is equal to the union of all these sets. In our experiments we have received a bit better results of tests applying the second method. If an unseen object has to be classified it is first matched against all decision rules from the constructed decision rule set. Next the final decision is predicted by applying some strategy predicting the final decision from all "votes" of decision rules. The simplest strategy we have tested was the majority voting i.e. the final decision is the one supported by the majority of decision rules. One can also apply fuzzy methods to predict the proper decision, but it is necessary to do this very carefully because most of these methods need a notion of a distance between objects and values of decision.

### 4.8 Dynamic rules

An idea of dynamic reducts can be adapted for a new method of dynamic rules computation. From a given data table a random set of subtables is chosen.

For example:

10 samples with the size 90% of the decision table,
10 samples with the size 80% of the decision table,
10 samples with the size 70% of the decision table,
10 samples with the size 60% of the decision table,
10 samples with the size 50% of the decision table.

Thus we receive 50 new decision tables. Then the decision rule sets for all these tables are calculated. In the next step the rule memory is constructed where all rule sets are stored. Intuitively, the dynamic rule is appearing in all (or almost all) of experimental subtables. The decision rules can be also computed from so called local reducts used to generate decision rules with minimal number of descriptors [14].

### 4.9 Complexity problems and some approximation solutions

The processes of decision rules' generation from decision tables are based on the reduct set computation. The time of reduct set computation can be too long when the decision table has too many: attributes or different values of attributes or objects. The reason is that in general the size of the reduct set can be exponential with respect to the size of the decision table and the problem of computing a minimal reduct is NP-hard [19]. Therefore we were forced to apply some approximation algorithms to obtain some knowledge about reduct sets. We can pursue the following courses of action: conceptual clustering values of attributes or groups of attributes; conceptual clustering objects; synthesis of new attributes from decision table and joining of attributes into groups [4].

Several experiments performed with different data tables (see [4]) are showing that our strategies for decision algorithm synthesis are increasing the quality of unseen object classification.

## Summary

We have presented some techniques based on rough sets which can help to construct decision algorithms with a high quality of classification. The discussed techniques have been tested on several data tables and they proved to be useful (see e.g. [4], [5]). The throughout analysis of these techniques and their improvement is caried out in our recent research projects.

## References

[1] Aarts E., Korst J.: Simulated Annealing and Boltzmann Machines, *Wiley*, New York 1989

[2] Almuallim H., Dietterich T.G.: Efficient Algorithms for Identifying Relevant Features, *Proc. of the Ninth Canadian Conference on Artificial Intelligence*, University of British Columbia, Vancouver, May 11-15, 1992, 38-45

[3] Anderberg M.R.: Cluster Analysis for Applications, *Academic Press*, New York 1973

[4] Bazan J., Skowron A., Synak P.:Dynamic Reducts as a Tool for Extracting Laws from Decision Tables, in: Methodologies for Intelligent Systems. *Proc. 8th International*

*Symposium ISMIS'94*, Charlotte, NG, October 1994, LNAI **vol. 869**, *Springer Verlag* 1994, 346-355

[5] Bazan J., Son H.N., Trung T.N., Skowron A., Synak P.: Some Logic and Rough set Applications for Classifying Objects, *ICS Research Report* 38/94, Warsaw University of Technology 1994.

[6] Brown E.M.: Boolean Reasoning. Dordrecht: Kluwer 1990

[7] Goldberg D.E.: Genetic Algorithms in Search Optimization and Machine Learning, *Addison-Wesley*, Reading MA, 1989

[8] Market Data, manuscript from *Hughes Research Laboratories*

[9] Kodratoff Y., Michalski R.: Machine Learning: An *Artificial Intelligence Approach*, Vol.3. San Mateo: Morgan Kaufmann 1990

[10] Michalski R., Tecuci G.: *Machine Learning. A* Multistrategy Approach vol.IV, Morgan Kaufmann 1994

[11] Nadler M., Smith E.P.: Pattern Recognition Engineering, *Wiley*, New York 1993

[12] Pawlak, Z.: Rough Sets, *International Journal of Information and Computer Science* 11 (1982),344-356

[13] Pawlak Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Dordrecht: Kluwer 1991

[14] Pawlak Z., Skowron A.: *A* Rough Set Approach for Decision Rules Generation. *ICS Research Report* 23/93, Warsaw University of Technology 1993, *Proc. of the IJCAI'93 Workshop: The Management of Uncertainty in AI*, France 1993

[15] Polkowski L.T.: Mathematical Morphology of Rough Sets, *Bull. Acad. Polon. Sci.*, Ser. Sci. Math. 41(3) (1993), 242-273.

[16] *Proceedings of the International Workshop on Rough Sets and Knowledge Discovery*, RSKD'93, Banff, October 11-15, Canada 1993, 101-104

[17] *Proceedings of the International Workshop on Rough Sets and Soft Computing*, RSSC'94, San Jose, California, November 10- 12, 1994, also in: Lin, T.Y., Wildberger, A.M. (eds.), *Soft Computing*, Simulation Councils, San Diego.

[18] Quinlan J. R.: C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, California 1993

[19] Skowron A.: Extracting Laws from Decision Tables: A Rough Set Approach, *Computational Intelligence*, 11(2), 1995, 371-388.

[20] Skowron A. and Polkowski L.: Synthesis of Decision Systems from Data Tables, in: (T.Y. Lin, N. Cercone (eds.)), *Rough Sets and Data Mining*, Kluwer, Dordrecht, 1997, 259-299.

[21] Skowron A., Polkowski L.: Analytical Morphology: Mathematical Morphology of Decision Tables, *Fundamenta Informaticae*, 27, 1996, 255-271.

[22] Tentush I.: On Minimal Absorbent Sets for some Types of Tolerance Relations, *Bull. Acad.Polon.Sci.*, Ser. Sci. Tech., 43(1), 1995, 79-88.

[23] Trung N. T. and Son N. H.: An Approach to the Handwriting Digit Recognition Problem Based on Modal Logic, *ICS Research Report* 44/93, Warsaw University of Technology 1993, M. Sc. Thesis, Warsaw University 1993

[24] Ziarko W.: Variable Precision Rough Set Model, *Journal of Computer and System Sciences*, 46(1), 1993, 39-59.