Feature Selection Using Rough Sets Theory

Maciej Modrzejewski

Institute of Computer Science, WUT
Nowowiejska 15/19
00-665 Warsaw, Poland

Abstract. The paper is related to one of the aspects of learning from examples, namely learning how to identify a class of objects a given object instance belongs to. In the paper a method of generating sequence of features allowing such identification is presented. In this approach examples are represented in the form of attribute-value table with binary values of attributes. The main assumption is that one feature sequence is determined for all possible object instances, that is next feature in the order does not depend on values of the previous features. The algorithm is given generating a sequence under these conditions. Theoretical background of the proposed method is rough sets theory. Some generalizations of this theory are introduced in the paper. Finally, a discussion of the presented approach is provided and results of functioning of the proposed algorithm are summarized. Direction of further research is also indicated.

1. Introduction.

One of the most important aspects of artificial intelligence (AI) is machine learning. During the last several years many methods and approaches have been proposed for this problem. Research in this field embraces inductive learning based on examples or on observations, discovery systems, neural nets learning, genetic algorithms learning and others [Mich91].

In our paper we deal with the inductive inference technique called learning from examples, more precisely with its subarea, from instance to class. The goal of learning is in this case identifying the class of objects to which a given instance of object belongs. The input data is a number of examples. In most applications examples are given as an attribute-value table. Attributes describe objects and directly reflect questions about object properties, which may be asked during process of object classification. Learning system must generate and represent somehow a way of making decisions concerning object class. The decisions, obviously, must be consistent with the provided examples.

There are two general methods of representing information related to decision making - a decision tree and a set of decision rules. For both representation methods many

algorithms have been proposed. The problem of generating minimal decision tree has received especially large attention. From the algorithms dealing with this problem the most notable is Quinlan's ID3 algorithm [Quin79] and its mutations. The common feature of the algorithms is that they produce trees in which query to be currently asked depends on the response to the preceding queries. We call such tree adaptive. Concerning the second representation, a well known algorithm of generating a set of decision rules from examples is e.g. AQ15 [HoMM86].

In both representations sequence of queries asked to classify object depends on the object instance. This means that generally it is different for different instances. In decision tree the sequence results directly from the tree and in the set of decision rules it depends on the algorithm of searching the set.

We investigate another possibility - generating "as good as possible" sequence of attributes, which would be applicable for all object instances. In this approach the features order is predetermined and questions are asked according to this order regardless of previous responses. This method may be related to the decision tree representation, however in this case a decision tree has specific properties. All nodes of a given level of the tree are assigned the same feature. Such tree will be called preset tree. To store a preset decision tree it is only necessary to store the set of attributes along with the information when to stop. This is an advantage over adaptive tree for which all possible paths must be stored. If the number of attributes and possible decisions is large, the number of paths in adaptive decision tree may be enormous and amount of memory needed to store all nodes of the tree may be prohibitively large. Then the adaptive method becomes inapplicable.

The above observation and the fact that the size of preset decision tree depends on the order of attributes motivate more thorough studying the possibility of generating an optimal order. In the following sections we present our considerations related to this problem. We present algorithm PRESET generating sequence of attributes. We propose an effective method of representing and storing a preset decision tree. Results of experiments of PRESET algorithm functioning are presented as well.

As a theoretical background we use the rough set theory, introduced by Pawlak [Pawl82]. The theory is suitable for the problem since it allows processing knowledge represented in a data table form, where objects are characterized by attributes. We use several notions of rough sets theory and operation of reduction of knowledge. This operation allows extracting most important properties which make different two objects belonging to two different classes. We also introduce some theoretical enhancements necessary to deal with our problem.

In the section 2 we present basics of the rough sets theory and in the section 3 our enhancements of this theory. The section 4 contains the algorithm for ordering attributes and in the section 5 we discuss the proposed method.

2. Main concepts of the rough sets theory.

Before presenting our investigations we first review basics of the rough sets theory, following [Pawl91].

Information systems.
Rough sets theory allows dealing with some type of knowledge related to a set of objects. In this approach the knowledge is a collection of facts concerning objects. The facts are represented in a data table form. Rows of the table correspond to the objects and columns to attributes describing the objects. Entries in a row represent knowledge about object corresponding to that row. The knowledge is expressed by values of attributes. A data table as above is called an *information system.*
Formally, an information system S is a 4-tuple $S = (U, Q, V, f)$, where

U - is a nonempty, finite set of objects, called the universe;
Q - is a finite set of attributes;
$V = \bigcup V_q$, where V_q is a domain of attribute q;
f - is an information function assigning a value of attribute for every object and every attribute, i.e.
$f : U \times Q \rightarrow V$, such that
for every $x \in U$ and for every $q \in Q$ $f(x,q) \in V$.

Indiscernibility relation.
For any set $P \subseteq Q$ of attributes a relation, called *indiscernibility relation* and denoted *IND* is defined as follows:

$$IND(P) = \{(x,y) \in U \times U : f(x,a) = f(y,a) \text{ for every } a \in P\}.$$

If $(x,y) \in IND(P)$, then x and y are called indiscernible with respect to P.
The indiscernibility relation is an equivalence relation over U. Hence, it partitions U into equivalence classes - sets of objects indiscernible with respect to P. Such partition (classification) is denoted by $U/IND(P)$.

Approximations of sets.
For any subset of objects $X \subseteq U$ and subset of attributes $P \subseteq Q$ the P-lower (denoted $\underline{PX}$) and P-upper (denoted $\overline{PX}$) *approximations* of X are defined as follows:

$$\underline{PX} = \bigcup \{ Y \in U/IND(P) : Y \subseteq X \} ;$$

$$\overline{PX} = \bigcup \{ Y \in U/IND(P) : Y \cap X \neq \emptyset \} .$$

A set for which $\underline{PX} = \overline{PX}$ is called an exact set, otherwise it is called rough (with respect to P).

Dependency of attributes.
A measure of dependency of two sets of attributes $P, R \subseteq Q$ is introduced in rough sets theory. The measure is called a *degree of dependency* of P on R and denoted $\gamma_R(P)$. It is defined as

$$\gamma_R(P) = \frac{\text{card}(POS_R(P))}{\text{card}(U)}, \text{ where}$$

$$POS_R(P) = \bigcup_{X \in U/IND(P)} \underline{R}X .$$

The set $POS_R(P)$ is called a positive region of classification $U/IND(P)$ (or in short a positive region of P) for the set of attributes R. Informally speaking, the set $POS_R(P)$ contains those objects of U which may be classified as belonging to one of the equivalence classes of $IND(P)$, employing attributes from the set R. The coefficient $\gamma_R(P)$ expresses numerically the percentage of objects which can be properly classified. For any two sets of attributes P, $R \subseteq Q$

$$0 \leq \gamma_R(P) \leq 1$$

and we say that P depends to degree $\gamma_R(P)$ on R.

Significance of attributes.

The coefficient γ is used to define an important for our investigations notion of *significance of an attribute*. The significance of an attribute $a \in R$, $R \subseteq Q$ is a measure expressing how important the attribute a is in R, regarding classification $U/IND(P)$. The significance is denoted σ_a^R and defined as follows:

$$\sigma_a^R(P) = \gamma_R(P) - \gamma_{R-\{a\}}(P).$$

Let us notice that such defined significance is relative in its nature since it depends on both set P and R. Therefore, an attribute may have different significance for different classifications and in different "contexts" (sets R in the definition above). However, we can also talk about an absolute significance of an attribute. For that purpose we take the whole set of attributes Q as the sets R and P in the definition. Then, i.e. for $R = P = Q$

$$\sigma_a^Q(Q) = \gamma_Q(Q) - \gamma_{Q-\{a\}}(Q),$$

and taking into account that $\gamma_Q(Q) = 1$,

$$\sigma_a^Q(Q) = 1 - \gamma_{Q-\{a\}}(Q).$$

Reduction of attributes.

Let $S = (U, Q, V, f)$ be an information system and let $P \subseteq Q$. Set P is called *independent* in S if for every $T \subset P$ $IND(P) \subset IND(T)$. Set $R \subseteq P \subseteq Q$ is a *reduct* of P if it is independent and $IND(R) = IND(P)$. This means that any reduct R of a set P classifies objects equally well as P does and attributes from $P-R$ are superfluous regarding distinguishing of objects.

Reducts are minimal in the sense that they cannot be reduced more (no attribute may be removed from a reduct without destroying its property of independence). The concept of reducts is one of the most important concepts in rough sets theory and it is used in most applications of the theory. There exist well defined and checked in practice algorithms which allow finding reducts effectively [SkRa91].

The example below illustrates the presented concepts.

Example 1.
Information system and indiscernibility relation.
An example information system is shown in Table 1 below.

	a	b	c	d
x_1	A	Z	4	4
x_2	B	X	3	2
x_3	C	X	2	2
x_4	C	Y	2	1
x_5	A	Y	4	1
x_6	B	Z	3	2
x_7	B	Y	3	3
x_8	C	Y	2	3

Table 1.

In this system:
$U = \{x_1, \ldots, x_8\}$, $Q = \{a, b, c, d\}$,
$(x_4, x_8) \in IND(\{a, b, c\})$,
$(x_2, x_3) \in IND(\{b, d\})$,
$(x_2, x_6) \in IND(\{a, d\})$ and so on.

Approximations.
Let us find the lower and upper approximations of set $X = \{x_1, x_2, x_4\}$ for $P = \{b, d\}$. Since $U/IND(P) = \{ \{x_1\}, \{x_2,x_3\}, \{x_4,x_5\}, \{x_6\}, \{x_7\ x_8\} \}$, then $\underline{P}X = \{x_1\}$ and $\overline{P}X = \{x_1, x_2, x_3, x_4, x_5\}$.
Therefore X is rough with respect to P.

Dependency.
Let us compute $POS_R(Q)$ for $R = \{a, b, c\}$. In this case $U/IND(R) = \{ \{x_1\}, \{x_2\}, \{x_3\}, \{x_4,x_8\}, \{x_5\}, \{x_6\}, \{x_7\} \}$. Since $U/IND(Q) = \{ \{x_1\}, \ldots, \{x_8\} \}$, then $POS_R(Q) = \{x_1, x_2, x_3, x_5, x_6, x_7\}$ and $\gamma_R(Q) = 0.75$.

Significance of attributes.
Using the above result we can conclude that
$\sigma_d^Q(Q) = 1 - \gamma_{\{a,b,c\}}(Q) = 0.25$.
Significance of other attributes in the system for $R = P = Q$ is given below:
$\sigma_a^Q(Q) = \sigma_c^Q(Q) = 0$,
$\sigma_b^Q(Q) = 0.25$.

Reducts.
There are two reducts of the set Q: $R_1 = \{a, b, d\}$ and $R_2 = \{b, c, d\}$. We can easily verify that both these sets and only they satisfy conditions stated in the definition of a reduct. To confirm that R_1 is a reduct we note that $U/IND(R_1) = U/IND(Q)$ and for every $q \in R$ $U/IND(R_1-\{q\}) \neq U/IND(R_1)$. After removing e.g. a we obtain $U/IND(\{b, d\}) \neq U/IND(Q)$.
□

3. Weighted information systems.

To solve the problem of ordering attributes, we introduce generalizations of some concepts of the rough sets theory. In this section we present notions used in further considerations.

We enhance modeling power of information systems by introducing weights of objects. To each object in a system its weight - a natural number - is assigned. The weights represent importance of objects in the system. Formally, in this case an information system becomes the *weighted information system WS* defined as follows:

$$WS = < U, Q, V, f, w >,$$

where U, Q, V and f are defined in the same way as in the definition of information system and w is a complete function assigning weights to objects:

$$w : U \rightarrow \mathbf{N},$$

where $\mathbf{N}$ stands for the set of natural numbers.

We call w a *weighting function* and $w(x)$ - a *weight* of an object $x \in U$. Then we take into account the weights in considerations regarding the system. We introduce definitions reflecting the presence of the weights. They modify meaning of some notions in classical rough sets theory. The definitions are shown below:

Weighted quality of classification.
Having two sets of attributes P, $R \subseteq Q$ we denote the weighted quality of classification $U/IND(R)$ by the set P as $W\gamma P(R)$ and define it as

$$W\gamma_P(R) = \frac{WPOS_P(R)}{WU}, \text{ where}$$

where $WPOS_P(R)$ is a sum of weights of objects constituting the positive region of classification $POS_P(R)$, i.e.

$$WPOS_P(R) = \Sigma (w(x) : x \in POS_P(R))$$

and WU is a total sum of weights of objects in the system, i.e.

$$WU = \Sigma (w(x) : x \in U).$$

Weighted significance of an attribute.
Let $P, R \subseteq Q$ and $a \in P$. By a weighted significance $W\sigma_a^P(R)$ of an attribute a in P, with respect to the classification $U/IND(R)$, we mean the value

$$W\sigma_a^P(R) = W\gamma_P(R) - W\gamma_{P-\{a\}}(R).$$

Let us note that in the simplest case, when $P = R = Q$ in the definitions above, the following equalities hold:

i) $WPOS_Q(Q) = WU.$

ii) $W\gamma_Q(Q) = 1.$

iii) $W\sigma_a^Q(Q) = 1 - W\gamma_{Q-\{a\}}(Q)$ for each $a \in Q.$

We will call $W\sigma_a^Q(Q)$ the *absolute weighted significance* of an attribute. For the sake of simplicity we will denote it as $W\sigma_a$.

Using the above modified notions we may model frequent in real life situation when objects are not homogeneous. In such case the important fact may be not only how many objects but also which of them are becoming indistinguishable when removing attributes. Therefore the weights assigned to objects should be interpreted as an importance of distinguishing these objects from the others in the system. The weighted significance of an attribute expresses how much of such importance of objects we lose removing this attribute.

Example 2.
Let us modify the information system from Example 1 by adding weights to the objects. The new, weighted system is shown in Table 2 below:

	$w(x)$	a	b	c	d
x_1	1	A	Z	4	4
x_2	2	B	X	3	2
x_3	3	C	X	2	2
x_4	1	C	Y	2	1
x_5	2	A	Y	4	1
x_6	3	B	Z	3	2
x_7	1	B	Y	3	3
x_8	2	C	Y	2	3

Table 2.

Now we compute the absolute weighted significance of the attributes with weights of objects as shown in the column $w(u)$:

$W\sigma_a = W\sigma_c = 0$,
$W\sigma_b = 1\frac{1}{3}$,
$W\sigma_d = 0.2$.

Comparing these values with results obtained in Example 1 we can notice that according to assigned weights attributes b and d are no longer equally significant. However loss of knowledge (expressed by unweighted significance) is the same for both of them, the loss of importance of objects (expressed by weighted significance) is higher for b and therefore removing this attribute is most "harmful" for the system. This results from the fact that without the attribute b objects x_2 and x_6 become indiscernible and we lose weights of total value 5, while without d weights of total value 3 are lost (sum of weights of x_4 and x_8, which become indiscernible in that case).
□

4. Algorithm to find sequence of attributes.

In this section we show a solution of the problem of ordering attributes to make the process of identifying objects most efficient. According to initial assumptions the problem is reduced to finding the proper permutation of attributes. This permutation should lead to minimal preset decision tree. A minimal tree is the one having minimal cost, i.e. sum of lengths of all paths from root to leaves.

We assume that the initial data provided is an information system of the form consistent with the definition given in the section 2. The other very important assumption is that attributes in the system have binary domain.

The first, preliminary step for generating an optimal sequence of attributes is determining a set of attributes which is to be ordered. In our approach this is achieved by identifying reducts of the set of all attributes. Then one reduct is chosen for the purpose of ordering attributes. Attributes not included in this reduct are removed from the system. The remaining attributes are independent, that is all of them are necessary for distinguishing objects.

The algorithm PRESET is proposed to solve the problem of ordering attributes. In view of the presented assumptions the initial point for the algorithm is an information system $S = < X, Q, V, p >$ with independent set of attributes having binary domain. To find a sequence of attributes we construct weighted information system $WS = <U, Q, V, f, w>$, in which

$U = X/IND(Q)$, i.e. objects in WS are equivalence classes of relation $IND(Q)$ in S; this means that no two objects in U are indiscernible or in other words $IND(Q)$ is empty over U,

Q is the set of attributes for which the order will be determined,
V = {value_0, value_1}, this means that attributes have binary domain; actual values depend on the system (these may be {0, 1}, {TRUE, FALSE}, {yes, no} and so on),
f is a function with values equal to values of p:
$f : U \times Q \rightarrow V : f(u,q) = p(x,q)$, where x is an arbitrary object belonging to u,
$w(u) = 2$ for every $u \in U$.

Let us denote a sequence of attributes generated by the algorithm by $\mathfrak{S}$. On the beginning the sequence $\mathfrak{S}$ is empty.

The algorithm PRESET is as follows:

1. Check cardinality of the set of attributes in WS.

1a. If the cardinality is 1, add the attribute to the sequence $\mathfrak{S}$ and finish the algorithm, the sequence $\mathfrak{S}$ is the sequence searched for, in the **reverse order**;

1b. Otherwise compute value of absolute weighted significance for each attribute in WS.
Proceed to step 2.

2. Choose the attribute having the lowest value of the significance (if there is one such attribute) or any attribute from the set of attributes having the lowest value (if there are more than one of them).
Proceed to step 3.

3. Add the chosen attribute (let it be q_i) to the sequence $\mathfrak{S}$.
Proceed to step 4.

4. Construct diminished weighted system $WS' = \langle U', Q', V, f', w' \rangle$, in which
$Q' = Q - \{q_i\}$,
$U' = U/IND(Q')$,
$f' : U' \times Q' \rightarrow V : f'(x',q') = f(x,q')$, where x is an arbitrary object belonging to u',
$w' : U' \rightarrow \{1, 2\} : w'(u') = \begin{cases} 2 \text{ if } card(u') = 1 \text{ and } w(u) = 2 \text{ for } u \in u' \\ 1 \text{ otherwise} \end{cases}$

Let $WS = WS'$, proceed to step 1.

■

Example 3:
Consider information system representing knowledge about some animals. The system in its weighted form is shown in Table 3.

	w	Warm_Blood	Can_Fly	Has_Fur	Lives_in_Water
Elephant (E)	2	yes	no	no	no
Shark (S)	2	no	no	no	yes
Bat (B)	2	yes	yes	yes	no
Python (P)	2	no	no	no	no
Hawk (H)	2	yes	yes	no	no
Dolphin (D)	2	yes	no	no	yes

Table 3.

For the sake of simplicity we abbreviate the attributes in the system by single letters: *B* for Warm_Blood, *C* for Can_Fly, *F* for Has_Fur and *W* for Lives_in_Water.

The system from Table 3 satisfies initial conditions for the algorithm PRESET. We start with computing weighted absolute significance of all attributes:

$W\sigma_B = W\sigma_W = \frac{8}{9}$.

$W\sigma_C = W\sigma_F = \frac{1}{2}$.

Therefore in this stage we can choose either attribute *C* or *F*. Let choose attribute *F*: $\mathfrak{S} = \langle F \rangle$.

Now we construct the new system with attribute *F* removed. It is shown in Table 4.

	w	*B*	*C*	*W*
{ E }	2	yes	no	no
{ S }	2	no	no	yes
{ B, H }	1	yes	yes	no
{ P }	2	no	no	no
{ D }	2	yes	no	yes

Table 4.

Absolute weighted significance of the attributes is now as follows:

$W\sigma_B = W\sigma_W = \frac{8}{9}$.

$W\sigma_C = \frac{1}{2}$.

Hence now we are obliged to attach attribute *C* to the sequence: $\mathfrak{S} = \langle F, C \rangle$.

The new form of the system is shown below, in the table 5.

	w	*B*	*W*
{ B, H, E }	1	yes	no
{ S }	2	no	yes
{ P }	2	no	no
{ D }	2	yes	yes

Table 5.

Significance of attributes:

$W\sigma_B = W\sigma_W = 1.$

Both attributes *B* and *W* are now equivalent. Let choose *W*: $\mathfrak{F} = \langle F, C, W \rangle$.
There remains only one attribute, which is added to the sequence, i.e. $\mathfrak{F} = \langle F, C, W, B \rangle$ and the algorithm ends.

The order searched for is therefore $\langle B, W, C, F \rangle$ and the sequence of questions to identify animal species should be as follows:
1. Is the animal warm-blooded?
2. Does the animal live in water?
3. Can the animal fly?
4. Does the animal have fur?

This sequence reflects the optimal strategy for reaching an answer about animal species, under condition that questions asked are always the same. Obviously, in three cases (Shark, Python and Dolphin) the answer is known already after two questions, case of Elephant requires three questions and in the case of Bat and Hawk all four questions are needed. The decision tree for the above sequence is shown in Figure 2.

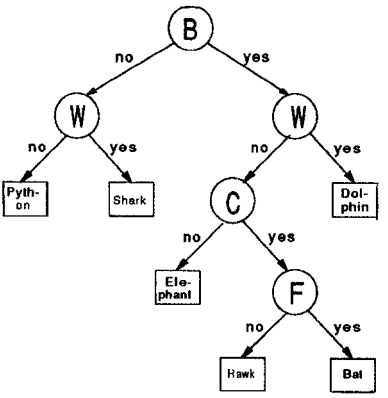

Figure 2.

Of 24 possibilities (all permutations of the four-element set of attributes) 4 produce minimal preset trees, i.e. trees with minimal sum of path lengths, which in this case is 17. We have found one of them. The remaining would have been obtained if we had made different choices in the first and third stages of the algorithm. More precisely, other sequences leading to minimal preset trees are: <*B*, *W*, *F*, *C*>, <*W*, *B*, *C*, *F*> and <*W*, *B*, *F*, *C*>.
□

5. Discussion of the proposed method.

A trivial solution of the problem of finding the optimal order of attributes is the exhaustive method, i.e. generating all possible permutations and selecting the best one. Nevertheless, it is impossible to use practically the exhaustive method since the number of permutations grows rapidly with the increase of number of attributes. Algorithm PRESET takes only a small fraction of time of exhaustive method.

However, our algorithm does not function well in all cases. In 100 experiments, when a set of examples has been generated randomly with various number of attributes (from 4 to 8) and objects, the algorithm arrived at correct result in 83 cases. We found that wrong results are obtained more probably if in early stage of the algorithm there are many attributes having the same value of significance and a decision must be made which one of them should be chosen. Error value, measured as the difference between generated and optimal tree costs did not exceed in conducted experiments 1.5% of the optimal tree cost.

The possible explanation of the above drawback is that the problem of finding optimal binary decision tree is known to be NP-complete [HyRi76]. Our problem, though formulated in different way, is similar, and is supposed to be of the class of NP-complete problems. We cannot yet prove or negate this statement, but if it is true, a polynomial algorithm cannot solve the problem. Algorithm PRESET is polynomial since amount of computation in each step of the algorithm depends on the number of object comparisons and significance evaluations. Assume that there are n objects (examples) and m attributes in a system. Number of comparisons needed to evaluate one significance is equal to the sum of all natural numbers from 1 to n-1, that is $\frac{n^2-n}{2}$. This must be repeated m times and the complexity is $O(\frac{m(n^2-n)}{2})$.

As was mentioned in the introduction, an advantage of preset decision tree is its memory requirements. Storing adaptive decision tree requires amount of memory needed to store all intermediate and terminal nodes. Assume that 4 memory units are devoted for an intermediate node (1 for an attribute, 2 for links to subtrees and 1 for node qualifier) and 2 units for a terminal node (1 for object identifier and 1 for node

qualifier). Since there are as many terminal nodes as examples and the least number of intermediate nodes for binary tree is n-1, at least $6n$-4 memory units are needed to represent an adaptive tree.

In our approach we need to store attributes in appropriate order and information when to stop. Assuming that for storing an attribute we need 1 memory unit, the first part requires as many units as there are attributes. Since under our assumptions $n > m+1$, in the worst case $m = n$-1 and we need n-1 memory units for storing sequence of attributes (the best case would be $\log_2 n$). The method of representation of the second part is not obvious and we present it briefly below. Assume that tree root is placed at level 0. For each example we store the value 2^{m-l}, where l is a tree level of the terminal node respective to this example. In our example system describing animals, objects Shark, Python and Dolphin would be assigned value 4, object Elephant value 2 and objects Bat and Hawk value 1. This representation requires 2 memory units for each example (1 for the above value and 1 for example identifier) and $2n$ units for all examples. Therefore maximal amount of memory needed in our representation is $3n$-1 and is approximately twice less than the memory needed to represent the respective adaptive tree. The actual gain is in most cases higher since the number of attributes is less than n-1. There exists a simple algorithm for generating such representation as well as an algorithm of making questions using this information for stopping. Due to the lack of space they will not be presented here.

6. Conclusions.

In this paper we have studied a problem of sequencing attributes in a information system to obtain an optimal preset decision tree. We have presented a polynomial algorithm solving the problem in many cases. We find the results encouraging and justifying further research. In particular, we intend to enhance the algorithm PRESET for the case of multivalued attributes. We would also like to identify precisely the cases when the algorithm produces non-optimal sequences.

Our algorithm seems to have wide area of applications, spanning from building expert systems, querying databases etc. to testing combinational circuits.

We think that the approach of preset sequence of attributes can combine advantages of simplicity and unambiguity of decision procedure in tree representation with power, intuitiveness and representation efficiency of a set of decision rules.

Acknowledgment

The author wish to thank Prof. A. Skowron of Warsaw University and Prof. M. Moshkov of Nizhniy Novgorod Technical University for their valuable comments and inspiring suggestions.

References

[HyRi76] Hyafil L., Rivest R.L., *Constructing optimal binary decision trees is NP-complete*, Information Processing Letters, Vol. 5, No. 1,May 1976, pp. 15-17.

[HoMM86] Hong J., Mozetic I., Michalski R.S., *AQ-15: Incremental Learning of Attribute-Based Descriptions from Examples, The Method and User's Guide*, Report ISG 86-5, UIUCDCS-F-86-949, Dept. of Computer Science, University of Illinois, Urbana, 1986.

[Mich91] Michalski R.S., *Inferential Learning Theory: A Conceptual Framework for Characterizing Learning Processes*, Report P91-13 MLI 91-6, Center for Artificial Intelligence, George Mason University, Fairfax, 1991.

[Pawl82] Pawlak Z., *Rough Sets*, International Journal of Computer and Information Sciences, Vol. 11, No. 5, 1982, pp. 341-356.

[Pawl91] Pawlak Z., *Rough Sets, Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, 1991.

[Quin79] Quinlan R., *Discovering rules from large collection of examples*, in D. Mitchie (editor), *Expert systems in the microelectronic age*, Edinburgh University Press, Edinburgh 1979.

[SkRa91] Skowron A., Rauszer C., *The Discernibility Matrices and Functions in Information Systems*, ICS WUT Research Report 1/91, Warsaw, 1991.