Extracting Relevant Information about Reduct Sets from Data Tables

Mikhail Ju. Moshkov[1], Andrzej Skowron[2], and Zbigniew Suraj[3]

[1] Institute of Computer Science, University of Silesia
Będzińska 39, 41-200 Sosnowiec, Poland
moshkov@us.edu.pl
[2] Institute of Mathematics, Warsaw University
Banacha 2, 02-097 Warsaw, Poland
skowron@mimuw.edu.pl
[3] Chair of Computer Science, University of Rzeszów
Rejtana 16A, 35-310 Rzeszów, Poland
zsuraj@univ.rzeszow.pl

Abstract. The direct searching for relevant reducts in the set of all reducts of a given data table can be often computationally infeasible, especially for large data tables. Hence, there is a need for developing efficient methods for extracting relevant information about reducts from data tables which could help us to perform efficiently the inducing process of the high quality data models such as rule based classifiers. Such relevant information could help, e.g., to reduce the dimensionality of the attribute set. We discuss methods for generating relevant information about reduct sets from information systems or decision tables. In particular, we consider a binary relation on attributes satisfied for two given attributes if and only if there is no reduct consisting them both. Moreover, we prove that for any fixed natural k, there exists a polynomial in time algorithm which for a given decision table T and given k conditional attributes recognizes if there exists a decision reduct of T covering these k attributes. We also present a list of problems related to the discussed issues. The reported results create a step toward construction of a software library reducing the searching costs for relevant reducts.

Keywords: Rough sets, decision tables, decision reducts, geometry of reducts.

1 Introduction

In the rough set approach for decision making often are used different kinds of reducts such as reducts of information systems, decision reducts or local reducts. It is well known that the size of the set of all reducts of information (decision) systems can be huge relative to the number of attributes. Hence, the direct searching in such sets for relevant reducts can be often computationally infeasible, especially for large data tables. Hence, there is a need for developing efficient methods for extracting relevant information about reducts from data tables which could

J.F. Peters et al. (Eds.): Transactions on Rough Sets IX, LNCS 5390, pp. 200–211, 2008.

help us to induce the high quality data models such as rule based classifiers. This relevant information could help, e.g., to reduce the dimensionality of the attribute set. We present examples illustrating how such relevant information about reduct sets can be generated from information systems or decision tables. In particular, we consider a binary relation on attributes satisfied by two given attributes if and only if there does not exist reduct consisting them both. We illustrate how such a relation can be used in the reduct generation. Moreover, we prove that for any fixed natural k, there exists a polynomial in time algorithm which for a given decision table T and given k conditional attributes recognizes if there exists a decision reduct of the decision system T containing these k attributes. Using this algorithm one can, in particular, eliminate all single attributes which are not covered by any reduct of T. We also shortly discuss how the dependencies between attributes can be used in the reduct generation. Finally, we also present a list of problems related to the discussed issue.

We plan to develop a software library which could be used in preprocessing of the reduct generation.

The set of all decision reducts of a decision table T [5] contains rich information about the table T. Unfortunately, there is no polynomial algorithms for construction of the set of all reducts.

In this paper, we show that there are polynomial (in time) algorithms for obtaining of indirect but useful information about this set.

We show that for any fixed natural k, there exists a polynomial (in time) algorithm $\mathcal{A}_k$ checking, for a given decision table T and given k conditional attributes, if there exist a reduct for T covering these k attributes.

The information obtained on the basis of algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ can be represented in a simple graphical form. One can construct a graph with the set of vertices equal to the set of attributes covered by at least one reduct, and the set of edges equal to the set of all pairs of attributes which do not belong to any reduct. The degree of an attribute in this graph (the number of edges incident to this attribute) characterizes the attribute importance. The changes of this graph after adding of a new object to the decision table allow us to evaluate the degree of influence of this new object on the reduct set structure. In the paper, we consider such graphs for three real-life decision tables. Some properties of such graphs are studied in [2].

Note that there exist close analogies between results of this paper and results obtained in [1], where the following problem was considered: for a given positive Boolean function f and given subset of its variables it is required to recognize if there exists a prime implicant of dual Boolean function f^d containing these variables.

Another approach for efficient extracting from a given decision table T of indirect information about the set of all reducts and a graphical representation of the information was considered in [9]. It was shown that there exists a polynomial algorithm for constructing the so-called pairwise core graph for a given decision table T. The set of vertices of this graph is equal to the set of conditional attributes of T, and the set of edges coincides with the two element sets of

attributes disjoint with the core of T (i.e., the intersection of all reducts of T) and having non-empty intersection with any reduct of T. This example is a step toward a realization of a program suggested in early 90s by Andrzej Skowron in his lectures at Warsaw University to study geometry of reducts aiming at developing tools for investigating geometrical properties of reducts in the space of all reducts of a given information system. For example, the core of a given information system can be empty but in the reduct space can exist only a few subfamilies of reducts such that the intersection of each subfamily is non-empty.

Yet another discussed in the paper method for extracting information about the reduct set from a given data table can be based on the dependencies between attributes in a given information system.

This paper is structured as follows. In Section 2, we discuss the problem of existence of reducts including a given set of attributes. The graphical representation of some information about the set of reducts is considered in Section 3. In Section 4, we discuss shortly a possible application of dependencies in the reduct generation. In Section 5, we present conclusions and a list of problems for further study. In the appendix, we present a polynomial algorithm for one of problems listed in Section 5. This algorithm was found during the final editing of the paper.

This paper is an extended version of [3].

2 On Covering of k Attribute Sets by Reducts

A *decision table* T is a finite table in which each column is labeled by a *conditional attribute*. Rows of the table T are interpreted as tuples of values of conditional attributes on some objects. Each row is labeled by a *decision* which is interpreted as the value of the *decision attribute*[1].

Let A be the set of conditional attributes (the set of names of conditional attributes) of T. We will say that a conditional attribute $a \in A$ *separates* two rows if these rows have different values at the intersection with the column labeled by a. We will say that two rows are *different* if at least one attribute $a \in A$ separates these rows. Denote by $P(T)$ the set of unordered pairs of different rows from T which are labeled by different decisions.

A subset R of the set A is called a *test* (or *superreduct*) for T if for each pair of rows from $P(T)$ there exists an attribute from R which separates rows in this pair. A test R for T is called a *reduct* for T if each proper subset of R is not a test for T. In the sequel, we deal with decision reducts but we will omit the word "decision".

Let us fix a natural number k. We consider the following *covering problem for k attributes by a reduct*: for a given decision table T with the set of conditional attributes A, a subset B of the set A, and k pairwise different attributes $a_1, \ldots, a_k \in B$ it is required to recognize if there exist a reduct R for T such that

[1] We consider uniformly both consistent and inconsistent decision tables. However, in the case of inconsistent decision table, one can use also the so called generalized decision instead of the original decision [5,6,7].

$R \subseteq B$ and $a_1, \ldots, a_k \in R$, and if the answer is "yes" it is required to construct such a reduct. We describe a polynomial in time algorithm $\mathcal{A}_k$ for the covering problem (see Algorithm 1).

For $a \in A$, we denote by $P_T(a)$ the set of pairs of rows from $P(T)$ separated by a. For $a_1, \ldots, a_k \in A$ and $a_j \in \{a_1, \ldots, a_k\}$ let

$$P_T(a_j | a_1, \ldots, a_k) = P_T(a_j) \setminus \bigcup_{i \in \{1,\ldots,k\} \setminus \{j\}} P_T(a_i).$$

For $a_1, \ldots, a_k \in A$, let

$$\mathcal{P}_T(a_1, \ldots, a_k) = P_T(a_1 | a_1, \ldots, a_k) \times \ldots \times P_T(a_k | a_1, \ldots, a_k).$$

Assuming that $(\pi_1, \ldots, \pi_k) \in \mathcal{P}_T(a_1, \ldots, a_k)$, we denote by

$$D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$$

the set of attributes a from $B \setminus \{a_1, \ldots, a_k\}$ such that a separates rows in at least one pair of rows from the set $\{\pi_1, \ldots, \pi_k\}$. Note that

$$D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k) = \bigcup_{j=1}^{k} D_T(B, a_j, \pi_j).$$

Algorithm 1. Algorithm $\mathcal{A}_k$ for solving of the covering problem for k attributes by a reduct

Input : Decision table T with the set of conditional attributes A, $B \subseteq A$, and $a_1, \ldots, a_k \in B$.
Output: If there exists a reduct R for T such that $R \subseteq B$ and $a_1, \ldots, a_k \in R$, then the output is one of such reducts; otherwise, the output is "no".

construct the set $\mathcal{P}_T(a_1, \ldots, a_k)$;
for *any tuple* $(\pi_1, \ldots, \pi_k) \in \mathcal{P}_T(a_1, \ldots, a_k)$ **do**
  $R \longleftarrow B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$
  if R *is a test for* T **then**
    while R *is not a reduct for* T **do**
      select $a \in R$ such that $R \setminus \{a\}$ is a test for T;
      $R := R \setminus \{a\}$
    end
    return R;
    stop
  end
end
return "no" (in particular, if $\mathcal{P}_T(a_1, \ldots, a_k) = \emptyset$, then the output is "no")

Using algorithm $\mathcal{A}_k$ (see Algorithm 1) first the set $\mathcal{P}_T(a_1, \ldots, a_k)$ is constructed. Next, for each tuple $(\pi_1, \ldots, \pi_k) \in \mathcal{P}_T(a_1, \ldots, a_k)$ the set

$$D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$$

is constructed and it is verified if the set $B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$ is a test for T. It is clear that $|\mathcal{P}_T(a_1, \ldots, a_k)| \leq n^{2k}$, where n is the number of rows in T. Using this inequality and the fact that k is fixed natural number, one can prove that the algorithm $\mathcal{A}_k$ has polynomial time complexity[2]. Unfortunately, the algorithm $\mathcal{A}_k$ has a relatively high time complexity.

The considered algorithm is based on the following proposition:

Proposition 1. *Let T be a decision table with the set of conditional attributes A, $B \subseteq A$, and $a_1, \ldots, a_k \in B$. Then the following statements hold:*

1. *A reduct R for T such that $R \subseteq B$ and $a_1, \ldots, a_k \in R$ exists if and only if there exists a tuple $(\pi_1, \ldots, \pi_k) \in \mathcal{P}_T(a_1, \ldots, a_k)$ such that*

$$B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$$

   is a test for T.
2. *If the set $S = B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$ is a test for T then each reduct Q for T, obtained from S by removing from S of some attributes, has the following properties: $a_1, \ldots, a_k \in Q$ and $Q \subseteq B$.*

Proof. Let R be a reduct for T such that $a_1, \ldots, a_k \in R$ and $R \subseteq B$. It is clear that for each $a_j \in \{a_1, \ldots, a_k\}$ there exists a pair of rows π_j from $P(T)$ such that a_j is the only attribute from the set R separating this pair. It is clear that $(\pi_1, \ldots, \pi_k) \in \mathcal{P}_T(a_1, \ldots, a_k)$ and $R \subseteq B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$. Since R is a reduct for T, we conclude that $B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$ is a test for T.

Let us assume that there exists a tuple $(\pi_1, \ldots, \pi_k) \in \mathcal{P}_T(a_1, \ldots, a_k)$ such that the set $S = B \setminus D_T(B, a_1, \ldots, a_k, \pi_1, \ldots, \pi_k)$ is a test for T. Let Q be a reduct for T obtained by removing some attributes from S. It is also clear that $Q \subseteq B$. Let $j \in \{1, \ldots, k\}$. Since a_j is the only attribute from the test S separating rows from π_j, we have $a_j \in Q$. Thus, $a_1, \ldots, a_k \in Q$. □

3 Graphical Representation of Information about the Set of Reducts

Let T be a decision table with the set of conditional attributes A. Let $B \subseteq A$. Using polynomial algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ one can construct a graph $G(T, B)$. The set of vertices of this graph coincides with the set of attributes $a \in B$ for each of which there exists a reduct R for T such that $R \subseteq B$ and $a \in R$. Two different vertices a_1 and a_2 of $G(T, B)$ are linked by an edge if and only if there is no a reduct R for T such that $R \subseteq B$ and $a_1, a_2 \in R$. Let us denote by $G(T)$ the graph $G(T, A)$.

Note that there exists close analogy between the graph $G(T)$ and the so-called co-occurance graph [1] for positive Boolean function f. The set of vertices of this

[2] Note that k is treated as a constant for the algorithm $\mathcal{A}_k$.

graph is equal to the set of variables of f. Two different variables are linked by an edge if and only if f has a prime implicant containing these variables.

Now, we present the results of three experiments with real-life decision tables from [4] (the first example was considered in [2]).

Example 1. [2] Let us denote by T_Z the decision table "Zoo" [4] with 16 conditional attributes $a_1, \ldots, a_{16}$ (we ignore the first attribute "animal name") and 101 rows. Only attributes $a_1, a_3, a_4, a_6, \ldots, a_{14}, a_{16}$ are vertices of the graph $G(T_Z)$. The set of reducts for T_Z is represented in Table 1. The graph $G(T_Z)$ is depicted in Fig. 1. For example, any reduct containing a_7 is disjoint with $\{a_8, a_9, a_{14}\}$.

Example 2. Let us denote by T_L the decision table "Lymphography" [4] with 18 conditional attributes $a_1, \ldots, a_{18}$ and 148 rows. Each of the considered attributes is a vertex of the graph $G(T_L)$. The graph $G(T_L)$ is depicted in Fig. 2.

Table 1. The set of reducts for the decision table T_Z ("Zoo")

$\{a_3, a_4, a_6, a_8, a_{13}\}$	$\{a_3, a_6, a_8, a_9, a_{12}, a_{13}\}$	$\{a_3, a_6, a_8, a_{13}, a_{16}\}$
$\{a_3, a_4, a_6, a_9, a_{13}\}$	$\{a_1, a_3, a_6, a_7, a_{10}, a_{12}, a_{13}\}$	$\{a_3, a_6, a_9, a_{13}, a_{16}\}$
$\{a_3, a_6, a_8, a_{10}, a_{13}\}$	$\{a_3, a_4, a_6, a_7, a_{10}, a_{12}, a_{13}\}$	$\{a_4, a_6, a_8, a_{11}, a_{13}, a_{16}\}$
$\{a_1, a_3, a_6, a_9, a_{10}, a_{13}\}$	$\{a_1, a_6, a_8, a_{10}, a_{12}, a_{13}\}$	$\{a_4, a_6, a_9, a_{11}, a_{13}, a_{16}\}$
$\{a_1, a_3, a_6, a_8, a_{11}, a_{13}\}$	$\{a_1, a_6, a_9, a_{10}, a_{12}, a_{13}\}$	$\{a_3, a_6, a_7, a_{10}, a_{11}, a_{13}, a_{16}\}$
$\{a_1, a_3, a_6, a_9, a_{11}, a_{13}\}$	$\{a_1, a_3, a_6, a_{10}, a_{13}, a_{14}\}$	$\{a_1, a_6, a_8, a_{10}, a_{11}, a_{13}, a_{16}\}$
$\{a_3, a_6, a_8, a_9, a_{11}, a_{13}\}$	$\{a_3, a_4, a_6, a_{10}, a_{13}, a_{14}\}$	$\{a_1, a_6, a_9, a_{10}, a_{11}, a_{13}, a_{16}\}$
$\{a_1, a_3, a_6, a_8, a_{12}, a_{13}\}$	$\{a_3, a_6, a_8, a_{11}, a_{13}, a_{14}\}$	$\{a_3, a_6, a_7, a_{10}, a_{12}, a_{13}, a_{16}\}$
$\{a_4, a_6, a_8, a_{12}, a_{13}\}$	$\{a_3, a_6, a_8, a_{12}, a_{13}, a_{14}\}$	$\{a_3, a_6, a_{10}, a_{13}, a_{14}, a_{16}\}$
$\{a_1, a_3, a_6, a_9, a_{12}, a_{13}\}$	$\{a_1, a_6, a_{10}, a_{12}, a_{13}, a_{14}\}$	$\{a_1, a_6, a_{10}, a_{11}, a_{13}, a_{14}, a_{16}\}$
$\{a_4, a_6, a_9, a_{12}, a_{13}\}$	$\{a_4, a_6, a_{10}, a_{12}, a_{13}, a_{14}\}$	$\{a_4, a_6, a_{10}, a_{11}, a_{13}, a_{14}, a_{16}\}$

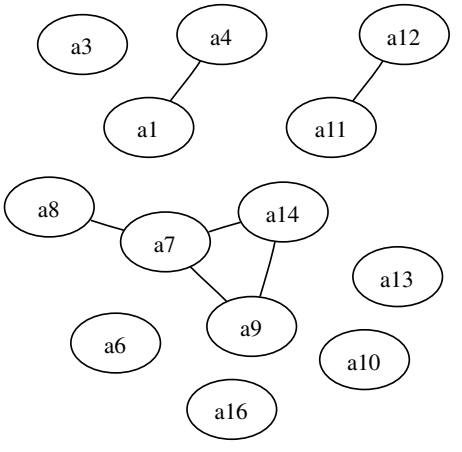

Fig. 1. Graph $G(T_Z)$ for the decision table T_Z ("Zoo")

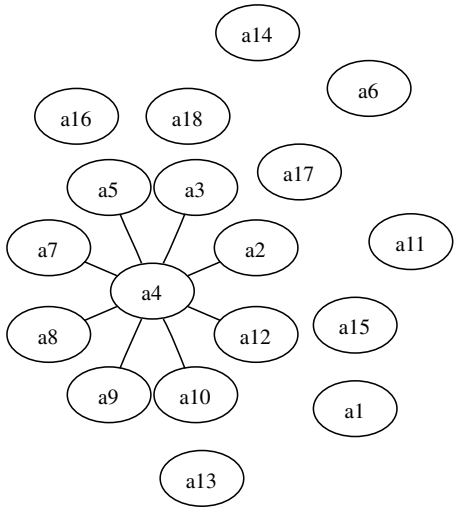

Fig. 2. Graph $G(T_L)$ for the decision table T_L ("Lymphography")

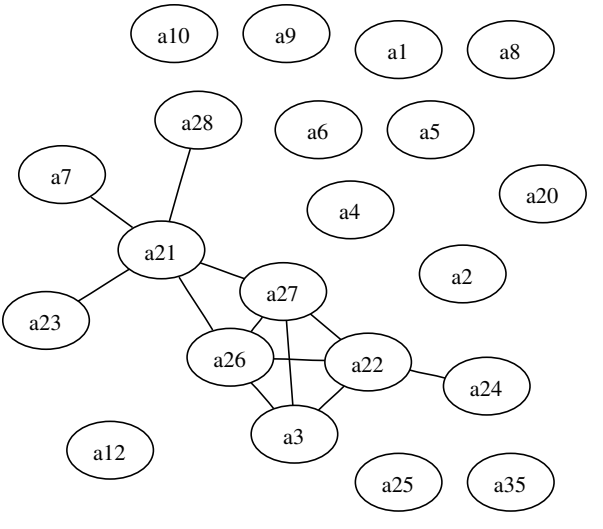

Fig. 3. Graph $G(T_S)$ for the decision table T_S ("Soybean-small")

In particular, one can observe from $G(T_L)$ that any reduct of T_L containing a_4 is disjoint with $\{a_2, a_3, a_5, a_7, a_8, a_9, a_{10}, a_{12}\}$.

Example 3. Let us denote by T_S the decision table "Soybean-small" [4] with 35 conditional attributes $a_1, \ldots, a_{35}$ and 47 rows. Only attributes $a_1, \ldots, a_{10}, a_{12}$

and $a_{20}, \ldots, a_{28}, a_{35}$ are vertices of the graph $G(T_S)$. The graph $G(T_S)$ is depicted in Fig. 3.

Some properties of graphs $G(T)$ are studied in [2].

It is shown in [2] that any undirected graph G can be represented as the graph $G(T)$ for an appropriate decision table T. However, the graph $G(T)$ can give us rich information about the set of reducts for a decision table T.

Proposition 2. *[2] Let A be a finite set of names of conditional attributes, and $G = (V, E)$ be an undirected graph, where $V \subseteq A$ is the set of vertices of G and E is the set of edges of G such that each edge of G is a two-element subset of V. Then there exists a decision table T with the set of names of conditional attributes A such that $G(T) = G$.*

Results of experiments considered in [2] show that there exists a correlation between the degrees[3] of attributes in $G(T)$ and the number of reducts of T covering these attributes (the last parameter is considered often as an attribute importance), and between changes of $G(T)$ and changes of the set of reducts for T after extending T by a new object (the changes in the set of reducts can be considered as a noticeable influence of updating of the decision table by the new object).

4 Using Dependencies in Generation of Reducts

Another important issue in data analysis is discovering dependencies between attributes in a given decision system $T = (U, C, D)$. Intuitively, a set of attributes D depends totally on a set of attributes C, denoted $C \Rightarrow D$, if the values of attributes from C uniquely determine the values of attributes from D. In other words, D depends totally on C, if there exists a functional dependency between values of C and D.

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

$$k = \gamma(C, D) = \frac{card(POS_C(D))}{card(U)}, \tag{1}$$

where

$$POS_C(D) = \bigcup_{X \in U/D} C_*(X),$$

called a *positive region* of the partition U/D with respect to C[4], is the set of all elements of U that can be uniquely classified to blocks of the partition U/D, by means of C.

If $k = 1$ we say that D *depends totally* on C, and if $k < 1$, we say that D *depends partially* (to *degree* k) on C. If $k = 0$ then the *positive region* of the partition U/D with respect to C is empty. The coefficient k expresses the ratio

[3] A degree of an attribute is the number of edges incident to this attribute.

[4] $C_*(X)$ denotes the C-lower approximation of X [6].

of all elements of the universe, which can be properly classified to blocks of the partition U/D, employing attributes C and will be called the *degree of the dependency.* Summing up: D is *totally* (*partially*) dependent on C, if *all* (*some*) elements of the universe U can be uniquely classified to blocks of the partition U/D, employing C. Observe, that (1) defines only one of possible measures of dependency between attributes (see, e.g., [8]).

Let us consider one very simple application of dependencies in reduct generation. One can also consider dependencies between conditional attributes in decision tables. In particular, if B, B' are subsets of conditional attributes in T then the dependency $B \Rightarrow B'$ is true in T if and only if the dependency $B \Rightarrow_1 B'$ holds in the decision system (U, B, B'). Then, in searching for decision reducts of T in which B should be included one can eliminate the attributes from B'. Obviously, if there exist two disjoint subsets B_1, B_2 of B such that $B_1 \Rightarrow B_2$ holds in T then there does not exist reduct covering B.

5 Conclusions

We have discussed some methods for generation of information from data tables which can be used in the reduct computation. In particular, we have shown that, for each natural k a polynomial algorithm $\mathcal{A}_k$ exists which for a given decision table and given k conditional attributes recognizes if there exist a decision reduct covering these k attributes. Results of computer experiments with two algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ are reported. Finally, we have shortly discussed applications of dependencies between conditional attributes in the reduct generation. In our project we are building a software library which could be helpful in solving different reduct generation problems. Methods from this library could be applied as some additional tools simplifying searching for relevant reducts.

Below we present a list of exemplary problems we would like to investigate in our further study. We are interested in computational complexity of these problems and algorithms (heuristics) for solving them.

The input for each problem is a data table T representing an information or decision system. By $RED(T)$ we denote the set of all reducts of T of a given kind (e.g., decision reducts).

- *Problem 1.* Let us consider a graph $G_{RED}(T)$ with nodes equal to elements of $RED(T)$. Two reducts are linked by an edge if and only if they have non-empty intersection. We would like to estimate the number of connected components of the graph $G_{RED}(T)$ [5].
- *Problem 2.* For given thresholds $tr, k > 0$ check if there exist at least k reducts with non-empty intersection consisting at least tr attributes.

[5] During the final editing of the paper we found a polynomial algorithm for this problem solving in the case when $RED(T)$ is the set of all decision reducts (see appendix). This algorithm has a relatively high time complexity. The problem of existence of more efficient algorithms is open.

- *Problem 3.* How many maximal (with respect to the number of elements) families of reducts from $RED(T)$ exist which satisfy the condition formulated in *Problem 2*?
- *Problem 4.* Find a maximal family of pairwise disjoint reducts in $RED(T)$.
- *Problem 5.* Let us consider a discernibility function for reducts defined by $dis_T(R) = |\{R' \in RED(T) : \ R \cap R' = \emptyset\}|$ for $R \in RED(T)$. Find bounds for $f_T(n) = \max\{dis_T(R) : \ R \in RED(T) \ \& \ |R| = n\}$.
- *Problem 6.* Let us consider a binary discernibility relation on subsets of attributes defined by $B \ DIS(T) \ C$ if and only if ($B \subseteq R$ and $R \cap C = \emptyset$) or ($C \subseteq R$ and $R \cap B = \emptyset$) for some $R \in RED(T)$, where B, C are subsets of the set of (conditional) attributes of T. What are the properties of $DIS(T)$?
- *Problem 7.* Let us consider a distance between reducts defined by

$$dist_T(R, R') = |R \setminus R'| + |R' \setminus R|,$$

  for $R, R' \in RED(T)$. Estimate the largest distance between reducts from $RED(T)$.
- *Problem 8.* Let us consider an incremental sequence of decision tables $T_i = (U_i, C, D)$ for $i = 1, 2, \ldots$, where $U_i \subseteq U_{i+1}$ for any i. We would like to develop methods for reasoning about changes between $RED(T_i)$ and $RED(T_{i+1})$.

In our further study we also would like to check if there exist efficient randomized algorithms for solution of the considered in the paper problem.

Acknowledgments

The research has been partially supported by the grant N N516 368334 from Ministry of Science and Higher Education of the Republic of Poland and by the grant "Decision support – new generation systems" of Innovative Economy Operational Programme 2008-2012 (Priority Axis 1. Research and development of new technologies) managed by Ministry of Regional Development of the Republic of Poland.

The authors are grateful to Marcin Piliszczuk for performing of experiments and for his suggestion allowing us to improve algorithms $\mathcal{A}_k$.

References

1. Boros, E., Gurvich, V., Hammer, P.L.: Dual subimplicants of positive Boolean functions. Optimization Methods and Software 10, 147–156 (1998)
2. Moshkov, M., Piliszczuk, M.: Graphical representation of information on the set of reducts. In: Yao, J., Lingras, P., Wu, W.-Z., Szczuka, M.S., Cercone, N.J., Ślęzak, D. (eds.) RSKT 2007. LNCS (LNAI), vol. 4481, pp. 372–378. Springer, Heidelberg (2007)
3. Moshkov, M., Skowron, A., Suraj, Z.: On covering attribute sets by reducts. In: Kryszkiewicz, M., Peters, J.F., Rybinski, H., Skowron, A. (eds.) RSEISP 2007. LNCS (LNAI), vol. 4585, pp. 175–180. Springer, Heidelberg (2007)

4. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases. University of California, Irvine, Department of Information and Computer Sciences (1998),
`http://www.ics.uci.edu/~mlearn/MLRepository.html`
5. Pawlak, Z.: Rough Sets – Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht (1991)
6. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Information Sciences 177(1), 3–27 (2007)
7. Pawlak, Z., Skowron, A.: Rough sets and Boolean reasoning. Information Sciences 177(1), 41–73 (2007)
8. Ślęzak, D.: Approximate entropy reducts. Fundamenta Informaticae 53, 365–387 (2002)
9. Wróblewski, J.: Pairwise cores in information systems. In: Ślęzak, D., Wang, G., Szczuka, M.S., Düntsch, I., Yao, Y. (eds.) RSFDGrC 2005. LNCS (LNAI), vol. 3641, pp. 166–175. Springer, Heidelberg (2005)

Appendix: Comparison of Graphs $G_{RED}(T)$ and $G'(T)$

In this section, we consider a polynomial algorithm for solving of *Problem 1* in the case of decision reducts.

Let T be a decision table. We denote by $RED(T)$ the set of all decision reducts for T. The set of nodes of the graph $G_{RED}(T)$ coincides with $RED(T)$. Two reducts are linked by an edge if and only if they have non-empty intersection. Graph $G'(T)$ is the complement of the graph $G(T)$. The set of nodes of the graph $G'(T)$ coincides with the set $A_{RED}(T) = \bigcup_{R \in RED(T)} R$ of all conditional attributes of T each of which belongs to at least one decision reduct for T. Two attributes are linked by an edge if and only if there exists a decision reduct for T containing both these attributes.

We show that graphs $G_{RED}(T)$ and $G'(T)$ have the same number of connected components.

Proposition 3. *Let T be a decision table. Then the number of connected components of the graph $G_{RED}(T)$ is equal to the number of connected components of the graph $G'(T)$.*

Proof. Let $B_1, \ldots, B_n$ be all connected components of the graph $G_{RED}(T)$. For $i = 1, \ldots, n$, we denote by A_i the set of attributes contained in reducts belonging to B_i. It is clear that $A_1 \cup \ldots \cup A_n = A_{RED}(T)$ and $A_i \cap A_j = \emptyset$ for any $i, j \in \{1, \ldots, n\}$, $i \neq j$. For $i = 1, \ldots, n$, we denote by $G'(T, A_i)$ the subgraph of the graph $G'(T)$ generated by nodes from A_i.

Let us prove that $G'(T, A_1), \ldots, G'(T, A_n)$ are all connected components of the graph $G'(T)$. To this end we must show that, for any $i, j \in \{1, \ldots, n\}$, $i \neq j$, the following statements hold:

1. Any two nodes from A_i are connected by a path in the graph $G'(T)$.
2. Any node from A_i and any node from A_j are not linked by an edge.

Let $|A_i| = 1$. Then the first statement holds. The unique attribute from A_i forms a reduct. This attribute can not belong to any other reduct. Therefore, the second statement holds too.

Let $|A_i| > 1$. Let us consider arbitrary different nodes a and a' from A_i. Then there are reducts R and R' from B_i such that $a \in R$ and $a' \in R'$. If $R = R'$ then a and a' are linked by an edge. Let us assume that $R \neq R'$. We consider a shortest path $\alpha = R, R_1, \ldots, R_m, R'$ in $G_{RED}(T)$ connecting the reducts R and R'. We set $R_0 = R$ and $R_{m+1} = R'$. For $t = 0, \ldots, m$, we choose an attribute $a_t \in R_t \cap R_{t+1}$. It is clear that $R_1, \ldots, R_m$ belong to B_i. Therefore, $a_t \in A_i$ for $t = 0, \ldots, m$. Let us consider the sequence

$$\beta = a, a_0, \ldots, a_m, a'.$$

Notice that it is possible that $a = a_0$ or $a_m = a'$. Since α is a shortest path connecting R and R', we have $a_0 \neq a_1 \neq a_2 \neq \ldots \neq a_m$. It is clear that $a, a_0 \in R_0 = R$, $a_0, a_1 \in R_1$, ..., $a_{m-1}, a_m \in R_m$, $a_m, a' \in R_{m+1} = R'$. Therefore, the sequence β forms a path connecting a an a' in $G'(T)$. Thus, the first statement holds.

Let us assume that there exist a node $a_1 \in A_i$ and a node $a_2 \in A_j$ which are linked by an edge. Then there exists a reduct R such that $a_1, a_2 \in R$, which is impossible since $R \in B_i$ and $R \in B_j$. Thus, the second statement holds. Hence, we obtain that $G'(T, A_1), \ldots, G'(T, A_n)$ are all connected components of the graph $G'(T)$. □

Proposition 3 allows us to solve *Problem 1* (for the case of decision reducts for a decision table T) in the following way: using polynomial algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ we construct the graph $G'(T)$ and find (in polynomial time) the number of connected components in this graph. The obtained number is equal to the number of connected components in the graph $G_{RED}(T)$.

Unfortunately, algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ have a relatively high time complexity. The problem of existence of more efficient algorithms for *Problem 1* solving is open.

Example 4. Let us consider decision tables T_Z ("Zoo"), T_L ("Lymphography") and T_S ("Soybean-small") discussed in Examples 1-3. Simple analysis of graphs $G(T_Z)$, $G(T_L)$ and $G(T_S)$, which are the complements of the graphs $G'(T_Z)$, $G'(T_L)$ and $G'(T_S)$, shows that each of graphs $G'(T_Z)$, $G'(T_L)$ and $G'(T_S)$ has exactly one connected component. Therefore, each of graphs $G_{RED}(T_Z)$, $G_{RED}(T_L)$ and $G_{RED}(T_S)$ has exactly one connected component.