Rough Sets and Association Rule Generation

Hung Son Nguyen & Sinh Hoa Nguyen

Institute of Mathematics, Warsaw University
Banaacha Str. 2, 02-095, Warsaw Poland.
e-mail: son@ mimuw.edu.pl; hoa@ mimuw.edu.pl

Abstract. ASSOCIATION RULE (see [1]) extraction methods have been developed as the main methods for mining of real life data, in particular in Basket Data Analysis. In this paper we present a novel approach to generation of association rule, based on Rough Set and Boolean reasoning methods. We show the relationship between the problems of association rule extraction for transaction data and relative reducts (or α-reducts generation) for a decision table. Moreover, the present approach can be used to extract association rules in general form. The experimental results show that the presented methods are quite efficient. Large number of association rules with given support and confidence can be extracted in a short time.

1 Introduction

Many problems in Data Mining can be solved by applying the Rough Set theory (see [14, 18, 23, 24]). Rough Sets theory is based on dealing with indiscernibility and discernibility between objects described by finite number of features. In [6, 24, 8] one can find results proving that Rough Set theory offers many interesting methods for many important Data Mining tasks like: rule induction, concept description (granularity), pattern extraction, discretization, symbolic value grouping.

Association rules (see [1]) are one of the important tools for data analysis, in particular in Basket Data Analysis see [4]. Association rules describe the relationship between attributes. An example of association rule is as follows:

More than 90% of customers, who bought article A and article B, also bought article C and article D and article E...

This is a simplest form of association rules. One can see that in this formula, all attributes (articles A, B, C, D, E) have only one value ("bought"). This limitation does not allow to analyze the data bases with multi-valued or continuous attributes. In this paper we consider a generalized form of association rules where descriptors can have different forms (e.g. (attribute = value) or (attribute ∈ Value_Set)).

Many association rule extraction methods have been proposed (see e.g. [1, 21]). Any extraction method consists of two main steps, i.e., template extraction (or large item set extraction [1]) and association rule generation from templates.
We consider two optimization problems. First is related to optimal template extraction (i.e., the long templates supported by large number of objects). The second is related to the problem of shortest representative association rule generation (see e.g. [7]) from a given template.

In this paper we present a novel approach to association rule extraction based on rough set theory and boolean reasoning. We show the correspondence between the reduct finding problem in rough set theory and the problem of searching for representative association rules from template. We present some theoretical results related to optimization problems (described above) as well as numerous strategies for producing semi-optimal association rules from data.

The paper is organized as follows. In Section 2 we introduce some basic notions of rough sets theory, boolean reasoning, templates and association rules. In Section 3 we recall some results related to template generation problem. In Section 4 we introduce the notion of association rule as strongly related to the notion of template, known from rough sets theory. We show the NP-hardness of the problem of finding the optimal approximate (in sense of a confidence threshold) association rule corresponding to a given template. In Section 5 we show how optimal approximate association rules can be searched for as minimal approximate reducts, by using an appropriate decision table representation. We present two algorithms for association rule extraction from template. In Section 6 describe some directions for future research.

2 Basic notions

An information system is a pair $S = (U, A)$, where $U$ is a non-empty, finite set called the universe and $A$ is a non-empty, finite set of attributes. Each $a \in A$ corresponds to function $a : U \to V_a$, where $V_a$ is called the value set of $a$. Elements of $U$ are called situations, objects or rows, interpreted as, e.g., cases, states, patients, observations.

In the paper we also consider a special case of information systems called decision tables. In a decision table $S = (U, A \cup \{d\})$, $d \notin A$ is a distinguished attribute called decision. The elements of $A$ are called conditional attributes (conditions).

2.1 Rough set preliminaries

With any subset of attributes $B \subseteq A$, we define an information vector for any object $x \in U$ by

$$\text{inf}_B(x) = \{(a, a(x)) : a \in B\}$$

An equivalence relation called the $B$-indiscernibility relation [14], denoted by $IND(B)$, is defined by

$$IND(B) = \{(x, y) \in U \times U : \text{inf}_B(x) = \text{inf}_B(y)\}$$

Objects $x, y$ satisfying relation $IND(B)$ are indiscernible by attributes from $B$. By $[x]_{IND(B)}$ we denote the equivalence class of $IND(B)$ defined by $x$. A
minimal subset $B$ of $A$ (with regard to inclusion) such that $\text{IND}(A) = \text{IND}(B)$ is called a reduct of $A$.

If $\mathcal{A} = (U, A)$ is an information system, $B \subseteq A$ is a set of attributes and $X \subseteq U$ is a set of objects, then the sets

$$B^\text{low} = \{x \in U : [x]_{\text{IND}(B)} \subseteq X\} \quad \text{and} \quad B^\text{up} = \{x \in U : [x]_{\text{IND}(B)} \cap X \neq \emptyset\}$$

are called $B$-lower and $B$-upper approximation of $X$ in $\mathcal{A}$, respectively.

If $\mathcal{A} = (U, A \cup \{d\})$ is a decision table and $B \subseteq A$ then we define a function

$$\partial_B : U \rightarrow 2^{|1, \ldots, r(d)|},$$

called the generalized decision in $\mathcal{A}$, by

$$\partial_B(x) = \{i : \exists x' \in U \ [\langle x' \text{IND}(B)x \rangle \wedge (d(x') = i)]\} = d \left([x]_{\text{IND}(B)}\right)$$

A decision table $\mathcal{A}$ is called consistent (deterministic) if $\text{card}(\partial_A(x)) = 1$ for any $x \in U$, otherwise $\mathcal{A}$ is inconsistent (non-deterministic).

The set of attributes $B \subseteq A$ is called "relative reduct" or simply reduct of decision table $\mathcal{A}$ if and only if

1. $\partial_B(x) = \partial_A(x)$ for all object $x \in U$.
2. any proper subset of $B$ does not satisfy the previous condition.

i.e., $B$ is a minimal subset (with respect to the inclusion relation $\subseteq$) of attributes satisfying the property $\forall x \in U \partial_B(x) = \partial_A(x)$.

There are two problems related to the notion of "reduct", which have been intensively explored in rough set theory by many researchers (see e.g. [2, 20, 19]). The first problem is related to searching for "shortest reducts" (i.e. reduct with minimal cardinality). The second problem is related to searching for all reducts. It has been shown that the first problem is NP-hard (see [18]) and second is at least NP-hard. Some heuristics has been proposed for those problems. Here we present the approach based on Boolean reasoning as proposed in [18].

2.2 Boolean reasoning approach

Many problems in rough set theory (e.g. reduct finding, rule extraction, discretization [12, 17]) has been successively solved by Boolean reasoning approach. This simple method is based on encoding the investigated optimization problem $\pi$ by a corresponding Boolean function $f_\pi$ in such a way that any prime implicant of $f_\pi$ states a solution of $\pi$. We illustrate this approach by the reduct problem.

Given a decision table $\mathcal{A} = (U, A \cup \{d\})$, where $U = \{u_1, u_2, \ldots, u_n\}$, $A = \{a_1, \ldots, a_k\}$. By discernibility matrix of the decision table $\mathcal{A}$ we denote the $(n \times n)$ matrix

$$M(\mathcal{A}) = [C_{i,j}]_{i,j=1}^n$$

such that $C_{i,j}$ is the set of attributes discerning $u_i$ and $u_j$. Formally:

$$C_{i,j} = \begin{cases} \{a_m \in A : a_m(u_i) \neq a_m(u_j)\} & \text{if } d(x_i) \neq d(x_j) \\ \emptyset & \text{otherwise.} \end{cases}$$
One can also define the *discernibility function* $f_A$ as a Boolean function:

$$f_A(a_1^*, ..., a_k^*) = \bigwedge_{i,j} \left( \bigvee_{a_m \in C_{i,j}} a_m^* \right)$$

where $a_1^*, ..., a_k^*$ are Boolean variables corresponding to attributes $a_1, ..., a_k$. One can show that prime implications of $f_A(a_1^*, ..., a_k^*)$ correspond exactly to reducts in $A$.

Intuitively, the set $B \subseteq A$ of attributes is called "consistent with $d$" (or $d$-consistent) if $B$ has nonempty intersection with any nonempty set $C_{i,j}$ i.e.

$$B \text{ is consistent with } d \iff \forall_{i,j}(C_{i,j} = \emptyset) \lor (B \cap C_{i,j} \neq \emptyset)$$

The set of attributes is reduct if it is minimal (with respect to inclusion) among $d$-consistent set of attributes.

Moreover, in some applications (see [16,15]), instead of reducts we prefer to use their approximations called $\alpha$-reducts, where $\alpha \in [0,1]$ is a real parameter. The set of attributes is called $\alpha$-reduct if it is minimal (with respect to inclusion) among the sets of attributes $B$ such that

$$\text{disc degree}(B) = \frac{|\{C_{i,j} : B \cap C_{i,j} \neq \emptyset\}|}{|\{C_{i,j} : C_{i,j} \neq \emptyset\}|} \geq \alpha$$

When $\alpha = 1$, the notions of $\alpha$-reduct and normal reduct are the same. One can show that for a given $\alpha$, the problems of searching for shortest $\alpha$-reducts and for all $\alpha$-reducts are also NP-hard [13].

Boolean reasoning approach also offers numerous approximate algorithms for solving the problem of prime implication extraction from boolean functions. We illustrate the simplest heuristics for this problem, called *greedy algorithm*, by the example of minimum reduct searching problem. In many applications this algorithm seems to be quite efficient.

**Semi-minimal reduct finding**

1. Set $R := \emptyset$;
2. Insert to $R$ the attribute $a$ which occurs most frequently in the discernibility matrix $M(A)$
3. For all $C_{i,j}$
   - if $(a \in C_{i,j})$ then set $C_{i,j} := \emptyset$;
4. If there exists nonempty element in $M(A)$ then go to Step 2 else go to 5
5. Remove from $R$ all unnecessary attributes.

The modified version of greedy algorithm for shortest $\alpha$-reduct generation from given decision table $A_j^T$, could be as follows:
Semi-minimal α-reduct finding

1. Let \( k = \lceil \alpha \cdot |S| \rceil \) be the minimal number of members of \( M(\mathbb{A}) \), which can have the empty intersection with the resultant reduct \( R \).
2. \( R := \varnothing \);
3. Insert to \( R \) the attribute \( a \) which occurs most frequently in discernibility matrix \( M(\mathbb{A}) \);
4. For all \( C_{i,j} \)
   - if \( (a \in C_{i,j}) \) then set \( C_{i,j} := \varnothing \);
5. If there exist more than \( k \) nonempty element in \( M(\mathbb{A}) \) then go to Step 3
   else go to Step 7;
7. Remove from \( R \) all unnecessary attributes.

2.3 Templates as patterns in data

Let \( \mathbb{A} = (U, A) \) be an information table. By descriptors (or simple descriptors) we mean the terms of form \( (a = v) \), where \( a \in A \) is an attribute and \( v \in V_a \) is a value in the domain of \( a \) (see [10]). By template we mean the conjunction of descriptors:

\[
T = D_1 \land D_2 \land \ldots \land D_m
\]

where \( D_1, \ldots, D_m \) are either simple or generalized descriptors. We denote by \( \text{length}(T) \) the number of descriptors being in \( T \).

For the given template with length \( m \):

\[
T = (a_1 = v_1) \land \ldots \land (a_m = v_m)
\]

the object \( u \in U \) is said to be satisfy the template \( T \) if and only if \( \forall_j a_j(u) = v_j \).

In this way the template \( T \) describes the set of objects having the common property: "their values on attributes \( a_1, \ldots, a_m \) are equal to \( v_1, \ldots, v_m \), respectively". In this sense one can use templates to describe the regularity in data, i.e., patterns - in Data Mining or granules - in Soft Computing.

Templates, except length, are also characterized by their support. The support of a template \( T \) is defined by

\[
\text{support}(T) = |\{u \in U : u \text{ satisfies } T\}|
\]

From description point of view we prefer the long templates with large support.

In some applications we consider the special kind of templates called decision templates or decision rules. Almost all objects satisfying a decision template should belong to one decision class. Usually we denote decision templates by formulas of the form

\[
T \Rightarrow (d = v_i)
\]

where \( v_i \) is an arbitrary value from domain \( V_d \) of the decision attribute \( d \). The decision template quality is measured by the precision function:

\[
\text{precision}(T \Rightarrow (d = v_i)) = \frac{\text{support}(T \land (d = v_i))}{\text{support}(T)}
\]
In [11] we investigated a generalized form of templates, which allow to increase the number of supporting objects from data set. Such patterns, called generalized templates, are defined by conjunction of clauses (descriptors) of form \((\text{attribute} \in \text{value set})\) (see [11]).

### 2.4 Association rules

Association rules and their generations can be defined in many ways (see [1, 7]). Hereafter, according to the presented notation, association rules can be defined as implication of the form

\[ P \Rightarrow Q \]

where \(P\) and \(Q\) are different simple templates, i.e. the formulas of the form

\[ (a_{i_1} = v_{i_1}) \land \ldots \land (a_{i_n} = v_{i_n}) \implies (a_{j_1} = v_{j_1}) \land \ldots \land (a_{j_k} = v_{j_k}) \] \hspace{1cm} (1)

The presented form can be called generalized association rules, because association rules are originally defined by formulas \(P \Rightarrow Q\) where \(P\) and \(Q\) are the sets of items (i.e. goods or articles in markets) (see [1]) e.g.

\[ \{ A, B \} \Rightarrow \{ C, D, E \}. \]

One can see that this form can be obtained from Equation (1) by replacing values on descriptors by "1" i.e.: 

\[ (A = 1) \land (B = 1) \implies (C = 1) \land (D = 1) \land (E = 1). \]

Usually, for a given information table \(A\), the quality of the association rule \(R = P \Rightarrow Q\) can be evaluated using two measures called support and confidence with respect to \(A\). The support of the rule \(R\) is defined by the number of objects from \(A\) satisfying the condition \((P \land Q)\) i.e.

\[ \text{support}(R) = \text{support}(P \land Q) \]

The second measure – confidence of \(R\) – is the ratio between the support of \((P \land Q)\) and the support of \(P\) i.e.

\[ \text{confidence}(R) = \frac{\text{support}(P \land Q)}{\text{support}(P)} \]

Large effort has been done to solve a typical task related to associated rules which is formulated as follows:

**For a given information table \(A\), an integer \(s\), and a real number \(c \in (0; 1)\), find as much as possible association rules \(R = P \Rightarrow Q\) such that \(\text{support}(R) \geq s\) and \(\text{confidence}(R) \geq c\).**

All existing association rule generation methods (see e.g. [1]) consists of two main steps:
1. Generate as much "good templates" as possible $T = D_1 \wedge D_2 \ldots \wedge D_k$ such that $\text{support}(T) \geq s$ and $\text{support}(T \wedge D) < s$ for any descriptor $D$ (i.e. maximal templates among those which are supported by more than $s$ objects).

2. For any "good template" $T$, search for a partition $T = P \wedge Q$ such that:
   (a) $\text{support}(P) < \text{support}(T)$
   (b) $P$ is the shortest template satisfying the above condition.

In this paper we show that the second module can be solved using rough set methods.

3 Optimal templates

In previous papers (see e.g. [10]) we have shown that some optimization problems related to optimal (simple) template generation either are NP-hard or determination of their complexity class is still unsolved problem. In [10] we have shown the following theorems:

**Theorem 1.** Given an information system $\mathbb{A} = (A, U)$ and positive integer $L$. The optimization problem of searching for a template $T$ (if any) of length $L$ with maximal support is NP-hard.

In practice we do not have the length $L$ of templates hence, we must modify the optimization criterion. Possible functions are the following:

$$\text{quality}(T) = \text{support}(T) \times \text{length}(T)$$

or

$$\text{quality}(T) = \alpha \cdot \text{support}(T) + \beta \cdot \text{length}(T) \quad \text{for some } \alpha, \beta \in \mathbb{R}.$$

From the computational complexity point of view, the problems of searching for template with maximal quality (for a given information system $\mathbb{A} = (U, A)$) is either NP-hard or still open.

In [11] we considered more general form of descriptors, namely $(\text{variable} \in \text{value.set})$, where value.set is a subset of the variable domain. We have present methods for generalized pattern extraction.

Using machine learning terminology, we distinguish two models of pattern extraction techniques called unsupervised methods and supervised methods, with regard to the means of utilizing the information about decision classes. The unsupervised methods try to extract the sufficiently simple (generalized) templates describing relatively large set of objects. In the meantime, the supervised methods aspire to discover the templates, that describe the regularity of objects from one (or several) decision class and discern them from another decision classes.

We present a general scheme for template generation algorithms. Then we will discuss in details the most important methods of optimal template extraction from data.
We consider the following general scheme of greedy algorithms called the *template lengthening strategy*. Before running the algorithm we have to determine the quality function defined for an arbitrary template. The quality of the template $T$, say $\text{quality}(T)$, estimates the fitness of $T$ to the specific application.

At the beginning, the template variable $T$, which we are looking for, is initialized as an *empty template*. In the successive steps, $T$ is extended by adding the descriptors that maximize the quality of the resulting template.

But, it is impossible to determine the influence of the single descriptor to the quality of the resulting template in every iteration. Hence, for any temporary template $T$, we define the *fitness* of a descriptor $D$ to the template $T$ (i.e. function $\text{fitness}_T(D)$). The fitness of the descriptor reflects its potential possibility to create a good template.

For every descriptor $a \in S_a$, the fitness function reflects a chance, that a new template defined by conjunction $T \land (a \in S_a)$ is optimal. For example, it can be defined by

$$\text{fitness}_T(D) = \text{quality}(T \land D) - \text{quality}(T)$$

The searching algorithm is presented in the scheme below:

<table>
<thead>
<tr>
<th>Template Lengthening Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $i := 0; T_i = \emptyset$.</td>
</tr>
<tr>
<td>2. while ($A \neq \emptyset$)</td>
</tr>
<tr>
<td>3. <strong>Choose</strong> the attribute $a \in A$ and the corresponding value set $S_a \subset V_a$ such that $(a \in S_a)$ is the best descriptor according to $\text{fitness}_T(\cdot)$;</td>
</tr>
<tr>
<td>4. $T_{i+1} := T_i \land (a \in S_a); i := i + 1;$.</td>
</tr>
<tr>
<td>5. <strong>Remove</strong> the attribute $a$ from the attribute set $A$;</td>
</tr>
<tr>
<td>6. endwhile.</td>
</tr>
<tr>
<td>7. <strong>Return</strong> the template $T_{\text{best}}$ with maximal quality.</td>
</tr>
</tbody>
</table>

Any algorithm based on the presented scheme should at the beginning be provided two factors: i.e. **the estimation of the quality of templates** and **the strategy of searching for the best descriptor** ($a \in S_a$) of the given attribute $a$ (the Step 3).

This scheme can be easily modified to generate more than one template. Instead of choosing only one descriptor in Step 3, one can store several (say $k$, where $k$ is a parameter) best descriptors.

The second strategy is called the *local strategy* following an observation: "any template is a part of information vector of an object from given information system". Hence, one can restrict the set of all descriptors to the set of descriptors defined by an object called "generator". This strategy is used by methods proposed in [1] where the generator is the object containing the value "bought" on all attributes (articles). In case of templates with arbitrary values, the problem is how to choose the best generators from the set of all objects. In [10,11] we proposed some methods to solve this problem.
4 From templates to optimal association rules

Let us assume that the template $T$, which is supported by at least $s$ objects, has been found using one of the algorithms presented in the previous section. We assume that $T$ consists of $m$ descriptors i.e.

$$ T = D_1 \land D_2 \land \ldots \land D_m $$

where $D_i$ (for $i = 1, \ldots, m$) is a descriptor of the form $(a_i = v_i)$ for some $a_i \in A$ and $v_i \in V_a$. We denote the set of all descriptors occurring in the template $T$ by $DESC(T)$ i.e.

$$ DESC(T) = \{D_1, D_2, \ldots, D_m\} $$

Any set of descriptors $P \subseteq DESC(T)$ defines an association rule

$$ R_P = \{ \bigwedge_{D_i \in P} D_i \Rightarrow \bigwedge_{D_j \notin P} D_j \} $$

The function confidence of association rule $R_P$ can be redefined as

$$ confidence (R_P) = \frac{support(T)}{support(\bigwedge_{D_i \in P} D_i)} $$

i.e. the ratio of the number of objects satisfying $T$ to the number of objects satisfying all descriptors from $P$. The length of the association rule $R_P$ is number of descriptors from $P$.

In practice, we would like to find as many as possible association rules with satisfactory confidence (i.e. $confidence (R_P) \geq c$ for a given $c \in (0; 1]$). The following property holds for confidence of association rules:

$$ P_1 \subseteq P_2 \quad \Rightarrow \quad confidence (R_{P_1}) \leq confidence (R_{P_2}) \quad (2) $$

This property says that if association rule $R_P$ generated from descriptor set $P$ has satisfactory confidence then association rule generated from any superset of $P$ also has satisfactory confidence.

For a given confidence threshold $c \in (0; 1]$ and given set of descriptors $P \subseteq DESC(T)$, the association rule $R_P$ is called $c$-representative if

1. $confidence (R_P) \geq c$;
2. For any proper subset $P' \subset P$ we have $confidence (R_{P'}) < c$.

From Equation (2) one can see that instead of searching for all association rules, it is enough to find all $c$-representative rules. Moreover, every $c$-representative association rule covers a family of association rules. The shorter is association rule $R$, the bigger is the set of association rules covered by $R$. First at all, we show the following theorem:

**Theorem 2.** For an fixed real number $c \in (0; 1]$ and a template $T$, the problem of searching for a shortest $c$-representative association rule for a given table $A$ from $T$ (Optimal $c$-Association Rules Problem) is NP-hard.
Proof. Obviously, Optimal $c$-Association Rules Problem belongs to NP. We show that the Minimal Vertex Covering Problem (which is NP-hard, see e.g. [5]) can be transformed to the Optimal $c$-Association Rules Problem.

Let the graph $G = (V, E)$ be an instance of Minimal Vertex Cover Problem, where $V = \{v_1, v_2, ..., v_n\}$ and $E = \{e_1, e_2, ..., e_m\}$. We assume the every edge $e_i$ is represented by two-element set of vertices i.e. $e_i = \{v_{i_1}, v_{i_2}\}$. We construct the corresponding information table (or transaction table) $A(G) = (U, A)$ for Optimal $c$-Association Rules Problem as follows:

1. The set $U$ consists of $m$ objects corresponding to $m$ edges of the graph $G$ and $k + 1$ objects added for some technical purposes i.e.

   
   $U = \{x_1, x_2, ..., x_k\} \cup \{x^*\} \cup \{u_{e_1}, u_{e_2}, ..., u_{e_m}\}$

   
   where $k = \left\lceil \frac{c}{1-c} \right\rceil$ is a constant derived from $c$.

2. The set $A$ consists of $n$ attributes corresponding to $n$ vertices of the graph $G$ and 1 attribute added for some technical purposes i.e.

   
   $A = \{a_{v_1}, a_{v_2}, ..., a_{v_n}\} \cup \{a^*\}$

   
   The value of attribute $a \in A$ over the object $u \in U$ is defined as follows:

   (a) if $u \in \{x_1, x_2, ..., x_k\}$ then

   
   $a(x_i) = 1$ for any $a \in A$.

   (b) if $u = x^*$ then then for any $j \in \{1, ..., n\}$:

   
   $a_{v_j}(x^*) = 1$ and $a^*(x^*) = 0$.

   (c) if $u \in \{u_{e_1}, u_{e_2}, ..., u_{e_m}\}$ then for any $j \in \{1, ..., n\}$:

   
   $a_{v_j}(u_e) = \begin{cases} 0 & \text{if } v_j \in e_i \\ 1 & \text{otherwise} \end{cases}$ and $a^*(u_e) = 1$

The illustration of our construction is presented in Figure 1.

We will show that any set of vertices $W \subseteq V$ is minimal covering set for the graph $G$ if and only if the set of descriptors

$P_W = \{(a_{v_j} = 1) : \text{for } v_j \in W\}$

defined by $W$ encodes the shortest $c$-representative association rule for $A(G)$ from the template

$T = (a_{v_1} = 1) \land ... \land (a_{v_n} = 1) \land (a^* = 1)$.

The first implication ($\Rightarrow$) is obvious. We show that implication ($\Leftarrow$) also holds:

The only objects satisfying $T$ are $x_1, ..., x_k$, hence we have $\text{support}(T) = k$. Let $P \Rightarrow Q$ be an optimal $c$-confidence association rule derived from $T$. Then we have $\frac{\text{support}(T)}{\text{support}(P)} \geq c$, hence

$\text{support}(P) \leq \frac{1}{c} \cdot \text{support}(T) = \frac{1}{c} \cdot k = \frac{1}{c} \cdot \left\lceil \frac{c}{1-c} \right\rceil \leq \frac{1}{1-c} = \frac{c}{1-c} + 1$
Example
Let us consider the Optimal c-Association Rules Problem for $c = 0.8$. We illustrate the proof of Theorem 2 by the graph $G = (V,E)$ with five vertices $V = \{v_1, v_2, v_3, v_4, v_5\}$ and six edges $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$. First we compute $k = \left\lceil \frac{6}{1-c} \right\rceil = 4$. Hence, the information table $A(G)$ consists of six attributes $\{a_{v_1}, a_{v_2}, a_{v_3}, a_{v_4}, a_{v_5}, a^*\}$ and $(4 + 1) + 6 = 11$ objects $\{x_1, x_2, x_3, x_4, x^*, u_{e_1}, u_{e_2}, u_{e_3}, u_{e_4}, u_{e_5}, u_{e_6}\}$. The information table $A(G)$ constructed from the graph $G$ is presented in the next figure:

![Graph](image)

**Fig. 1.** The construction of information table $A(G)$ from the graph $G = (V,E)$ with five vertices and six edges for $c = 0.8$.

Because $support(P)$ is an integer number, we have

$$support(P) \leq \left\lceil \frac{c}{1-c} \right\rceil + 1 = \left\lceil \frac{c}{1-c} \right\rceil + k = 4 + 4 = 8$$

Thus, there is at most one object from the set $\{x^*\} \cup \{u_{e_1}, u_{e_2}, ..., u_{e_6}\}$ satisfying the template $P$. We consider two cases:

1. **The object $x^*$ satisfies $P$**; then the template $P$ can not contain the descriptor $(a^* = 1)$ i.e.

   $$P = (a_{v_{i_1}} = 1) \land ... \land (a_{v_{i_4}} = 1)$$

   and there is no object from $\{u_{e_1}, u_{e_2}, ..., u_{e_6}\}$ which satisfies $P$ i.e. for any edge $e_j \in E$ there exists a vertex $v_i \in \{v_{i_1}, ..., v_{i_4}\}$ such that $a_{v_i}(u_{e_j}) = 0$ (which means that $v_i \notin e_j$). Hence the set of vertices $W = \{v_{i_1}, ..., v_{i_4}\} \subseteq V$ is a solution of the Minimal Vertex Cover Problem.

2. **An object $u_{e_j}$ satisfies $P$**; then $P$ consists of the descriptor $(a^* = 1)$ thus

   $$P = (a_{v_{i_1}} = 1) \land ... \land (a_{v_{i_4}} = 1) \land (a^* = 1)$$

   Let us assume that $e_j = \{v_j, v_{j_2}\}$. We consider two templates $P_1, P_2$ obtained from $P$ by replacing the last descriptor by $(a_{v_{j_1}} = 1)$ and $(a_{v_{j_2}} = 1)$.
respectively, i.e.

\[ P_1 = (a_{v_1} = 1) \land \ldots \land (a_{v_i} = 1) \land (a_{v_j} = 1) \]

\[ P_2 = (a_{v_1} = 1) \land \ldots \land (a_{v_k} = 1) \land (a_{v_j} = 1) \]

One can prove that both templates are supported by exactly \( k \) objects: \( x_1, x_2, \ldots, x_l \) and \( x^* \). Hence, similarly to the previous case, the two sets of vertices \( W_1 = \{ v_{i_1}, \ldots, v_{i_l}, v_{j_1} \} \) and \( W_2 = \{ v_{i_1}, \ldots, v_{i_l}, v_{j_2} \} \) state the solutions of the Minimal Vertex Cover Problem.

We showed that any instance \( I \) of Minimal Vertex Cover Problem can be transformed to the corresponding instance \( I' \) of the Optimal \( c \)-Association Rule Problem in polynomial time and any solution of \( I \) can be obtained from solutions of \( I' \). The reasoning states that Optimal \( c \)-Association Rule Problem is NP-hard.

Since the problem of searching for shortest representative association rules is NP-hard, the problem of searching for all association rules must be also at least NP-hard because this is the more complex problem. Having all association rules one can easily find the shortest representative association rule. Hence we have the following:

**Theorem 3.** The problem of searching for all (representative) association rules from a given template is at least NP-hard unless \( P \neq NP \).

The NP-hardness of presented problems forces us to develop efficient approximate algorithms solving them. In the next section we show that they can be developed using rough set methods.

5 Searching for Optimal Association Rules by rough set methods

For solving the presented problem, we show that the problem of searching for optimal association rules from a given template is equivalent to the problem of searching for local \( \alpha \)-reducts for a decision table, which is well known problem in Rough set theory.

We construct the new decision table \( A|_T = (U, A|_T \cup d) \) from the original information table \( A \) and the template \( T \) as follows:

- \( A|_T = \{ a_{D_1}, a_{D_2}, \ldots, a_{D_m} \} \) is a set of attributes corresponding to the descriptors of the template \( T \)

\[
    a_{D_i}(u) = \begin{cases} 
    1 \text{ if the object } u \text{ satisfies } D_i, \\
    0 \text{ otherwise.} 
    \end{cases} 
\]

(3)

- the decision attribute \( d \) determines if a given object satisfies template \( T \) i.e.

\[
    d(u) = \begin{cases} 
    1 \text{ if the object } u \text{ satisfies } T, \\
    0 \text{ otherwise.} 
    \end{cases} 
\]

(4)
The following theorems describe the relationship between association rules problem and reduct searching problem.

**Theorem 4.** For a given information table \( \mathbb{A} = (U, \mathcal{A}) \) and a template \( \textbf{T} \), the set of descriptors \( \textbf{P} \) is reduct in \( \mathbb{A}_{|\textbf{T}} \) if and only if the rule

\[
\bigwedge_{D_i \in \textbf{P}} D_i \Rightarrow \bigwedge_{D_j \notin \textbf{P}} D_j
\]

is 100%-representative association rule from \( \textbf{T} \).

**Proof.** Any set of descriptors \( \textbf{P} \) is reduct in the decision table \( \mathbb{A}_{|\textbf{T}} \) if and only if every object \( u \) with decision 0 is discerned from objects with decision 1 by one of the descriptors from \( \textbf{P} \) (i.e. there is at least one 0 in the information vector \( \text{inf}_D(u) \)). Thus \( u \) does not satisfy the template \( \bigwedge_{D_i \in \textbf{P}} D_i \). Hence

\[
\text{support} \left( \bigwedge_{D_i \in \textbf{P}} D_i \right) = \text{support}(\textbf{T})
\]

The last equality means that

\[
\bigwedge_{D_i \in \textbf{P}} D_i \Rightarrow \bigwedge_{D_j \notin \textbf{P}} D_j
\]

is 100%-confidence association rule for table \( \mathbb{A} \). \( \square \)

Analogously, one can show the following

**Theorem 5.** For a given information table \( \mathbb{A} = (U, \mathcal{A}) \), a template \( \textbf{T} \), a set of descriptors \( \textbf{P} \subseteq \text{DESC}(\textbf{T}) \), the rule

\[
\bigwedge_{D_i \in \textbf{P}} D_i \Rightarrow \bigwedge_{D_j \notin \textbf{P}} D_j
\]

is a \( \alpha \)-representative association rule obtained from \( \textbf{T} \) if and only if \( \textbf{P} \) is \( \alpha \)-reduct of \( \mathbb{A}_{|\textbf{T}} \), where \( \alpha = 1 - \frac{s-1}{n-1} \), \( n \) is the total number of objects from \( U \) and \( s = \text{support}(\textbf{T}) \). In particular, the problem of searching for optimal association rules can be solved by \( \alpha \)-reduct finding problem.

**Proof.** Assume that \( \text{support}(\bigwedge_{D_i \in \textbf{P}} D_i) = s + \epsilon \), where \( s = \text{support}(\textbf{T}) \). Then we have

\[
\text{confidence} \left( \bigwedge_{D_i \in \textbf{P}} D_i \Rightarrow \bigwedge_{D_j \notin \textbf{P}} D_j \right) = \frac{s}{s + \epsilon} \geq c \quad (5)
\]

This condition is equivalent to

\[
e \leq \left( \frac{1}{c} - 1 \right) s
\]
Hence one can evaluate the discernibility degree of $P$ by

$$disc\_degree(P) = \frac{e}{n-s} \leq \left(\frac{\frac{1}{s} - 1}{n-s}\right) = \frac{n\left(\frac{1}{s} - 1\right)}{n-s} = 1 - \alpha$$

Thus

$$\alpha = 1 - \frac{n\left(\frac{1}{s} - 1\right)}{n-s}$$

Searching for minimal $\alpha$-reducts is well known problem in rough sets theory. One can show, that the problem of searching for shortest $\alpha$-reducts is NP-hard [13] and the problem of searching for the all $\alpha$-reducts is at least NP-hard. However, there exist many approximate algorithms solving the following problems:

1. Searching for shortest reduct (see e.g. [18])
2. Searching for $k$ short reducts (see e.g. [20])
3. Searching for all reducts (see e.g. [2, 18])

These algorithms are efficient from computational complexity point of view and on the other hand, in practical applications, the reducts generated by them are quite closed to the optimal one.

In Section 5.2 we represent some heuristics for those problems in term of association rule generation.

5.1 The Example

The following example illustrates the main idea of our method. Let us consider the following information table $A$ with 18 objects and 9 attributes.

Assume that the template

$$T = (a_1 = 0) \land (a_3 = 2) \land (a_4 = 1) \land (a_6 = 0) \land (a_8 = 1)$$

has been extracted from the information table $A$. One can see that $support(T) = 10$ and $length(T) = 5$. The new constructed decision table $A|_T$ is presented in Table 2.

The discernibility function for decision table $A|_T$ can be explained as follows

$$f(D_1, D_2, D_3, D_4, D_5) = (D_2 \lor D_4 \lor D_6) \land (D_1 \lor D_3 \lor D_4) \land (D_2 \lor D_3 \lor D_4)$$
$$\land (D_1 \lor D_2 \lor D_3 \lor D_4) \land (D_1 \lor D_3 \lor D_5)$$
$$\land (D_2 \lor D_3 \lor D_5) \land (D_3 \lor D_4 \lor D_5) \land (D_1 \lor D_5)$$

After simplification the condition presented in Table 2 we obtain six reducts for the decision table $A|_T$.

$$f(D_1, D_2, D_3, D_4, D_5) = (D_2 \land D_5) \lor (D_4 \land D_5) \lor (D_1 \land D_2 \land D_3) \lor$$
$$\quad (D_1 \land D_2 \land D_4) \lor (D_1 \land D_2 \land D_5) \lor (D_1 \land D_3 \land D_4)$$
\[
\begin{array}{c|cccc}
A & a_1 & a_2 & a_3 & a_4 \\
\hline
u_1 & 0 & 1 & 1 & 2 \\
u_2 & 0 & 2 & 1 & 0 \\
u_3 & 0 & 2 & 1 & 0 \\
u_4 & 2 & 1 & 0 & 1 \\
u_5 & 1 & 2 & 2 & 1 \\
u_6 & 0 & 1 & 2 & 1 \\
u_7 & 0 & 2 & 1 & 0 \\
u_8 & 0 & 2 & 1 & 0 \\
u_9 & 1 & 2 & 2 & 0 \\
u_{10} & 0 & 3 & 2 & 0 \\
u_{11} & 0 & 2 & 1 & 0 \\
u_{12} & 0 & 2 & 2 & 2 \\
u_{13} & 0 & 2 & 1 & 0 \\
u_{14} & 0 & 2 & 1 & 0 \\
u_{15} & 0 & 2 & 1 & 0 \\
u_{16} & 1 & 2 & 1 & 0 \\
u_{17} & 0 & 2 & 1 & 0 \\
u_{18} & 0 & 2 & 1 & 0 \\
\end{array}
\]

\[
\begin{array}{c|cccc}
A' & D_1 & D_2 & D_3 & D_4 \\
\hline
u_1 & 1 & 0 & 1 & 0 \\
u_2 & 1 & 1 & 1 & 1 \\
u_3 & 1 & 1 & 1 & 1 \\
u_4 & 1 & 1 & 1 & 1 \\
u_5 & 0 & 1 & 0 & 0 \\
u_6 & 1 & 0 & 0 & 0 \\
u_7 & 0 & 0 & 0 & 1 \\
u_8 & 1 & 1 & 1 & 1 \\
u_9 & 1 & 1 & 1 & 1 \\
u_{10} & 1 & 1 & 1 & 1 \\
u_{11} & 0 & 1 & 0 & 0 \\
u_{12} & 1 & 0 & 0 & 0 \\
u_{13} & 1 & 1 & 1 & 1 \\
u_{14} & 1 & 1 & 0 & 0 \\
u_{15} & 1 & 1 & 1 & 1 \\
u_{16} & 1 & 1 & 1 & 1 \\
u_{17} & 1 & 1 & 1 & 1 \\
u_{18} & 0 & 1 & 1 & 0 \\
\end{array}
\]

(a) Table 1. (a) The example of information table A and template T support by 10 objects (b) the new decision table $A'|T$ constructed from A and template T

\[
\begin{array}{c|c|c|c|c|c|c}
M(A'|T) & u_1, u_3, u_4, u_6, u_9 & u_{10}, u_{13}, u_{15}, u_{16}, u_{17} \\
\hline
u_1 & D_2 \lor D_4 \lor D_5 \\
u_2 & D_1 \lor D_3 \lor D_4 \\
u_3 & D_2 \lor D_3 \lor D_4 \\
u_4 & D_1 \lor D_3 \lor D_4 \\
u_5 & D_1 \lor D_2 \lor D_3 \lor D_4 \\
u_6 & D_1 \lor D_2 \lor D_3 \lor D_4 \\
u_7 & D_1 \lor D_2 \lor D_3 \lor D_4 \\
u_8 & D_1 \lor D_3 \lor D_4 \\
u_9 & D_1 \lor D_4 \\
u_{10} & D_1 \lor D_2 \lor D_3 \lor D_4 \\
u_{11} & D_1 \lor D_3 \lor D_4 \\
u_{12} & D_2 \lor D_3 \lor D_4 \\
u_{13} & D_1 \lor D_3 \lor D_4 \\
u_{14} & D_3 \lor D_4 \\
u_{15} & D_1 \lor D_3 \lor D_4 \\
u_{16} & D_5 \\
u_{17} & D_1 \lor D_3 \lor D_4 \\
u_{18} & D_1 \lor D_5 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c|c}
100\%-representative rules & D_3 \land D_5 & D_1 \land D_2 \land D_4 \land D_5 \\
\hline
D_1 \land D_2 \land D_3 & D_4 \land D_5 & D_1 \land D_2 \land D_3 \\
D_1 \land D_2 \land D_4 & D_3 \land D_5 \\
D_1 \land D_2 \land D_4 & D_2 \land D_3 \\
D_1 \land D_3 \land D_4 & D_2 \land D_3 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c|c}
90\%-representative rules & D_1 \land D_2 & D_3 \land D_4 \land D_5 \\
\hline
D_1 \land D_2 \land D_3 & D_3 \land D_4 \land D_5 \\
D_1 \land D_3 \land D_4 & D_3 \land D_4 \land D_5 \\
D_1 \land D_3 \land D_5 & D_3 \land D_4 \land D_5 \\
D_2 \land D_3 \land D_4 & D_1 \land D_2 \land D_3 \\
D_2 \land D_3 \land D_4 & D_1 \land D_2 \land D_3 \\
D_3 \land D_4 \land D_5 & D_1 \land D_2 \land D_3 \\
D_3 \land D_4 \land D_5 & D_1 \land D_2 \land D_3 \\
\end{array}
\]

Table 2. The simplified version of its discernibility matrix $M(A'|T)$; Representative association rules with (100\%)-confidence and representative association rules with at least (90\%)-confidence
Thus, we have found from template $\mathbf{T}$ six association rules with (100\%)-confidence (see Table 2).

For $c = 90\%$, we would like to find $\alpha$-reducts for the decision table $A|_{\mathbf{T}}$, where

$$\alpha = 1 - \frac{1 - \frac{1}{k}}{\frac{1}{k} - 1} = 0.86$$

Hence we would like to search for a set of descriptors that covers at least

$$[(n - s)(\alpha)] = [8 \cdot 0.86] = 7$$

elements of discernibility matrix $\mathcal{M}(A|_{\mathbf{T}})$. One can see that the following sets of descriptors:

$$\{D_1, D_2\}, \{D_1, D_3\}, \{D_1, D_4\}, \{D_1, D_5\}, \{D_2, D_3\}, \{D_2, D_5\}, \{D_3, D_4\}$$

have nonempty intersection with exactly 7 members of the discernibility matrix $\mathcal{M}(A|_{\mathbf{T}})$. In Table 2 we present all association rules achieved from those sets.

In Figure 2 we present the set of all 100\%-association rules (light gray region) and 90\%-association rules (dark gray region). The corresponding representative association rules are represented in bold frames.

**Fig. 2.** The illustration of 100\% and 90\% representative association rules
5.2 The approximate algorithms

As we can see in the previous example, the problem is to find the representative association rules encoded by subsets of the descriptor set in a lattice (see Figure 2). In general, there are two searching strategies: bottom-up and top-down. The top-down strategy starts with the whole descriptor set and tries to go down through the lattice. In every step we reduce the most superfluous subsets keeping the subsets which most probably can be reduced in the next step. Almost all existing methods realize this strategy (see [Agrawal,...]). The advantage of those methods is as follows:

1. It generates all association rules during searching process.
2. It is easy to implement them for parallel or concurrent computer.

But this process can take very long computation time because of NP-hardness of the problem (see Theorem 3).

The rough set based method realizes the bottom-up strategy. We start with the empty set of descriptors. Here we describe the modified version of greedy heuristics for the decision table $A_T$. In practice we do not construct this additional decision table. The main problem is to compute the occurrence number of descriptors in the discernibility matrix $M(A_T)$. For any descriptor $D$, this number is equal to the number of "0" occurring in the column $a_D$ represented by this descriptor and it can be computed using simple SQL queries of form

```
SELECT COUNT ... WHERE ...
```

We present two algorithms: the first finds almost shortest $c$-representative association rule.

---

**Short representative association rule**

**input:** Information table $A$, template $T$, minimal confidence $c$.

**output:** short $c$-representative association rule.

1. Set $P := \emptyset$; $U_P := U$; $\text{min\_support} := |U| - \frac{1}{c} \cdot \text{support}(T)$;
2. Choose a descriptor $D$ from $\text{DESC}(T) \setminus P$ which is satisfied by the smallest number of objects from $U_P$;
3. Set $P := P \cup \{D\}$;
4. $U_P := \text{Satisfy}(P)$; (i.e. set of objects satisfying all descriptors from $P$)
5. If $|U_P| > \text{min\_support}$ then go to Step 2 else stop;

After the algorithm stops we do not have any guarantee that the descriptor set $P$ is $c$-representative. But one can achieve it by removing from $P$ (which is in general small) all unnecessary descriptors.

The second algorithm finds $k$ short $c$-representative association rules where $k$ and $c$ are parameters given by the user.
**Short representative association rules**

**Input:** Information table $\mathbb{A}$, template $T$, minimal confidence $c \in [0, 1]$, number of representative rules $k \in \mathbb{N}$.

**Output:** $k$ short $c$-representative association rules $R_{P_1}, \ldots, R_{P_k}$.

1. For $i := 1$ to $k$
   
   Set $P_i := \emptyset$; $U_{P_i} := U$;

2. Set $\text{min\_support} := |U| - \frac{1}{c} \cdot \text{support}(T)$;

3. $\text{Result\_set} := \emptyset$; $\text{Working\_set} := \{P_1, \ldots, P_k\}$;

4. $\text{Candidate\_set} := \emptyset$;

5. for $(P_i \in \text{Working\_set})$ do
   
   Chose $k$ descriptors $D^i_1, \ldots, D^i_k$ from $\text{DESC}(T) \setminus P_i$ which is satisfied by smallest number of objects from $U_{P_i}$;
   
   Insert $P_i \cup \{D^i_1\}, \ldots, P_i \cup \{D^i_k\}$ to the $\text{Candidate\_set}$;

6. Select $k$ descriptor sets $P'_1, \ldots, P'_k$ from the $\text{Candidate\_set}$ (if exist) which are satisfied by smallest number of objects from $U$.

7. Set $\text{Working\_set} := \{P'_1, \ldots, P'_k\}$;

8. for $(P_i \in \text{Working\_set})$ do
   
   Set $U_{P_i} := \text{support}(P_i)$;
   
   if $|U_{P_i}| < \text{min\_support}$ then
     
     Move $P_i$ from $\text{Working\_set}$ to the $\text{Result\_set}$
   
   else GO TO Step 4;

9. if $(|\text{Result\_set}| > k \text{ or } \text{Working\_set} \text{ is empty})$ then STOP
   
   else GO TO Step 4;

---

**Fig. 3.** The illustration of the "$k$ short representative association rules" algorithm
6 Conclusions

We have presented two main results related to the problem of searching for shortest representative association rules from given template (OAR problem). First is related to determining the computational complexity of OAR problem. We have shown that OAR problem is NP-hard (derived from Minimal Vertex Covering problem, see Section 4). This result states that one can not expect any polynomial algorithm solving OAR problem unless \( P = NP \), and it is necessary to find out a good approximate algorithm solving OAR problem for large data bases. The second result concerns a correspondence between OAR and \( \alpha \)-reduct finding problem (well known in rough set theory). We have shown that for a given table \( A \) and a template \( T \) one can construct a new decision table \( A_T \) such that every \( \alpha \)-representative association rule obtained from \( T \) corresponds to one \( \alpha \)-reduct of \( A_T \) where \( \alpha \) is a coefficient derived from \( c, T \) and \( A \) (see Section 5). The approximate algorithms for OAR problem (based on heuristics for reduct finding) are presented in Section 5.2.

Acknowledgement

This work was partially supported by Research Program of European Union - ESPRIT-CRIT2 No. 20288.

References


\(^{1}\) shortening of Optimal Association Rule