Discovery of Interesting Action Rules

Zbigniew W. Raś[1,2], Angelina A. Tzacheva[1], and
Li-Shiang Tsay[1]

[1] UNC-Charlotte, Department of Computer Science,
Charlotte, N.C. 28223, USA
[2] Polish Academy of Sciences, Institute of Computer Science,
Ordona 21, 01-237 Warsaw, Poland

Abstract. There are two aspects of interestingness of rules, objective and subjective measures ([7], [1], [15], [16]. Objective measures are data-driven and domain-independent. Generally, they evaluate the rules based on their quality and similarity between them. Subjective measures are user-driven, domain-dependent, and include unexpectedness, novelty and actionability [7], [1], [15], [16]. Liu [7] defines a rule as actionable one, if user can do an action to his/her advantage based on that rule. In [12] it was assumed that actionability has to be expressed in terms of changes in values of certain attributes which are used in an information system. They introduced a new class of rules (called action rules) which are constructed from certain pairs of association rules extracted from the same information system. Conceptually similar definition of an action rule was proposed independently by [4]. Action rules have been investigated further in [14], [13], [11], and [18].
In order to construct action rules it is required that attributes in a database are divided into two groups: stable and flexible. Flexible attributes are used in a decision rule as a tool for making hints to a user what changes within some of their values are needed to reclassify a group of objects from one decision class into another one. In this paper, we give a strategy for constructing all action rules from a given information system and show that action rules constructed by system $DEAR$, presented in [13], cover only a small part of all action rules. Clearly, we are not interested in all action rules as we are not interested in extracting all possible rules from an information system. Classical strategies like $See5$, $LERS$, $CART$, $Rosetta$, $Weka$ are discovering rules which classification part is either the shortest or close to the shortest. This approach is basically ruling out all other classification rules unless their are surprising rules [17]. In this paper, we introduce the notion of a cost of an action rule and define interesting action rules as rules of the smallest cost. We give a strategy showing how interesting action rules can be generated from action rules discovered system $DEAR$.

1 Introduction

There are two aspects of interestingness of rules that have been studied in data mining literature, objective and subjective measures (see [7], [1], [15], [16]. Objective measures are data-driven and domain-independent. Generally, they evaluate

the rules based on their quality and similarity between them. Subjective measures, including unexpectedness, novelty and actionability, are user-driven and domain-dependent.

A rule is actionable if user can do an action to his/her advantage based on that rule [7]. This definition is not only subjective but we may even formulate actions which involve attributes outside the information system schema. In order to decrease the number of such actions, the definition of a new class of rules (called action rules) constructed from certain pairs of classification rules has been given in [12]. Independently, another formal definition of an action rule was proposed in [4]. These rules have been investigated further in [13], [11], [19], [18].

To give an example justifying the need of action rules, let us assume that a number of customers have stopped buying products at one of the grocery stores. To find the cause of their decision, possibly the smallest and the simplest set of rules describing all these customers is extracted from the customer database. For instance, let us assume that *[Nationality, European]* $\wedge$ *[Milk Products, Kefir]* $\longrightarrow$ *[Profit, Excellent]* is such a rule. Assume also that from the same database, a rule *[Nationality, European]* $\longrightarrow$ *[Profit, Average]* representing the remaining customers has been extracted. At the same time, we know that the grocery store stopped ordering *kefir* about a month ago. Now, by comparing these two rules, we can easily find out that the grocery store manager should start ordering kefir again if he does not want to loose more European customers. Action rule, constructed from these two rules, is represented by the expression: *[Nationality, European]* $\wedge$ *[Milk Products,* $\longrightarrow$ *Kefir]* $\longrightarrow$ *[Profit, Average* $\longrightarrow$ *Excellent]*. It should be read as: *If Europeans will buy Kefir, then they should shift from the average group of customers to the excellent one.* Ordering kefir by the grocery store manager is an example of its implementation.

Formal definition of an action rule and the algorithm to construct them from classification rules was proposed in [12]. This algorithm was implemented as one of the modules in *DEAR* [13] system.

In this paper, we present a new method to construct action rules directly from an information system. At the same time, we show that any action rule is naturally associated with a pair of classification rules from which it can be constructed following the strategy given in [12]. So, to have the certainty that strategy proposed in [12] generates all action rules from a given information system S, we need to develop an algorithm which generates all classification rules (not only rules which classification part is the shortest). The easiest way to achieve that goal is to modify one of the classical rule discovery methods. We have chosen system $LERS$ for that purpose because it requires quite simple modification (terms producing classification rules are marked in $LERS$ but in our algorithm they remain unmarked and they are still used in the process of classification rules construction).

Following the proposed strategy, all action rules can be constructed from all possible classification rules. Independently from the method of constructing

them (either directly from an information system or using pairs of extended classification rules), their number is not only too large but also both methods are too expensive (in terms of time complexity). It leaves us with a similar problem initially faced in association rules mining. It was partially solved by proposing a small subset of association rules, called representative rules, jointly with a very simple strategy of generating all remaining association rules from them. It was proved that this strategy is sound and complete [6].

Following similar approach, we could try to find a small subset of action rules and look for a strategy to generate from them all remaining action rules. As we mentioned before, the strategy presented in [12] discovers only a small subset of action rules. In this paper we propose a sound strategy of discovering a small part of the remaining action rules classified as interesting rules on the basis of their cost (cheaper means more interesting). Clearly, the question of identifying a small subset of action rules from which all the remaining action rules can be constructed is open.

2 Information System and Action Rules

An information system is used for representing knowledge. Its definition, presented here, is due to [8].

By an information system we mean a pair $S = (U, A)$, where:

- U is a nonempty, finite set of objects,
- A is a nonempty, finite set of attributes, i.e. $a : U \longrightarrow V_a$ is a function for any $a \in A$, where V_a is called the domain of a.

Objects, for instance, can be interpreted as customers. Attributes can be interpreted as features, offers made by a grocery store, characteristic conditions etc.

In this paper we only consider the special case of information systems called decision tables [8]. In any decision table together with the set of attributes a partition of that set into conditions and decisions is given. Additionally, we assume that the set of conditions is partitioned into stable conditions and flexible conditions. For simplicity reason, we assume that there is only one decision attribute. *Date of birth* is an example of a stable attribute. *Interest rate* on any customer account is an example of a flexible attribute (dependable on a bank). We adopt the following definition of a decision table:

By a decision table we mean an information system $S = (U, A_1 \cup A_2 \cup \{d\})$, where $d \notin A_1 \cup A_2$ is a distinguished attribute called decision. The elements of A_1 are called stable conditions, whereas the elements of A_2 are called flexible conditions.

As an example of a decision table we take $S = (\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}, \{a, c\} \cup \{b\} \cup \{d\})$ represented by Table 1. Attribute a is stable, $\{b, c\}$ is the set of

	a	b	c	d
x_1	0	S	0	L
x_2	0	R	1	H
x_3	0	S	0	L
x_4	0	R	1	H
x_5	2	P	2	L
x_6	2	P	2	L
x_7	2	S	2	H
x_8	2	S	2	H
x_9	2	S	0	L

Table 1. Decision System S

flexible attribute and d is a decision attribute. Also, we assume that H denotes a *high* profit and L denotes a *low* one.

In order to induce rules in which the THEN part consists of the decision attribute d and the IF part consists of attributes belonging to $A_1 \cup A_2$, subtables $(U, B \cup \{d\})$ of S where B is a d-reduct [8] in S are used for rules extraction. By $L(r)$ we mean all attributes listed in the IF part of a rule r. For example, if $r = [(a, 2) * (b, S) \longrightarrow (d, H)]$ is a rule then $L(r) = \{a, b\}$. By $d(r)$ we denote the decision value of that rule. In our example $d(r) = H$. If r_1, r_2 are rules and $B \subseteq A_1 \cup A_2$ is a set of attributes, then the equation $r_1/B = r_2/B$ means that the conditional parts of rules r_1, r_2 restricted to attributes B are the same. Now, if $r_1 = [(b, S) * (c, 2) \longrightarrow (d, H)]$, then $r_1/\{b\} = r/\{b\}$.

Any certain rule is optimal in S, if by dropping an attribute value listed in its conditional part we get a rule which is no longer certain one. Similar definition of optimality can be used for classification rules if we only assume that the assumption about their certainty is replaced by a threshold value for their minimal confidence.

Now, let us assume that $(a, v \longrightarrow w)$ denotes the fact that the value of attribute a has been changed from v to w for a number of objects in S. Similarly, the term $(a, v \longrightarrow w)(x)$ means that the attribute value $a(x) = v$ has been changed to $a(x) = w$ for the object x.

Let $S = (U, A_1 \cup A_2 \cup \{d\})$ be a decision table and certain rules r_1, r_2 are extracted from S. Assume that B_1 is a maximal subset of A_1 such that $r_1/B_1 = r_2/B_1$, $d(r_1) = k_1$, $d(r_2) = k_2$ and $k_1 \leq k_2$. Also, assume that $(b_1, b_2, ..., b_p)$ is a list of all attributes in $L(r_1) \cap L(r_2) \cap A_2$ on which r_1, r_2 differ in values and $r_1(b_1) = v_1$, $r_1(b_2) = v_2$,..., $r_1(b_p) = v_p$, $r_2(b_1) = w_1$, $r_2(b_2) = w_2$,..., $r_2(b_p) = w_p$.

By (r_1, r_2)-action rule r in S we mean the expression:

$$r = [[(b_1, v_1 \longrightarrow w_1) \wedge (b_2, v_2 \longrightarrow w_2) \wedge ... \wedge (b_p, v_p \longrightarrow w_p)] \Rightarrow [(d, k_1 \longrightarrow k_2)]].$$

By an action rule in S we mean any (r_1, r_2)-action rule r in S, where r_1, r_2 are classification rules extracted from S. Clearly, if r is an action rule in S, then more than one pair of classification rules extracted from S may be used to define r.

Similarly to the notation adopted for rules extracted from S, we assume that $L(r) = \{b_1, b_2, ..., b_p\}$. Two definitions related to support of an action rule are proposed. They both take an optimistic approach. We list them below.

We say that object x *supports* action rule r in S, if there is an object $y \in U$ such that:

- $(\forall i \leq p)[[b_i \in L(r)] \longrightarrow [b_i(x) = v_i]]$,
- $(\forall i \leq p)[[b_i \in L(r)] \longrightarrow [b_i(y) = w_i]]$,
- $(\forall b \in [A_1 - L(r)])[b(x) = b(y)]$,
- $d(x) = k_1$ and $d(y) = k_2$.

We say that object x *supports* (r_1, r_2)-action rule r in S, if there is an object $y \in U$ such that:

- x supports classification rule r_1,
- y supports classification rule r_2,
- $(\forall b \in [A_1 - L(r)])[b(x) = b(y)]$,
- $d(x) = k_1$ and $d(y) = k_2$.

Clearly, the second definition is much stronger since it requires from objects x, y to support classification rules r_1, r_2, correspondingly. The first definition requires only the support restricted to attributes in $\{b_i : 1 \leq i \leq p\} \cup \{d\}$. It does not relate to classification rules used to construct an action rule.

By the support of (r_1, r_2)-action rule r in S, denoted by $Sup_{S_{r_1,r_2}}(r)$, we mean the number of objects in S supporting (r_1, r_2)-action rule r.

By the support of action rule r in S, denoted by $Sup_S(r)$, we mean the number of objects in S supporting r. Clearly $Sup_{S_{r_1,r_2}}(r) \leq Sup_S(r)$, for any r_1, r_2.

By the confidence of action rule r in S, denoted by $Conf_S(r)$, we mean $[Sup_S(r)/Sup_S(\{v_i : 1 \leq i \leq p\})]$.

By the confidence of (r_1, r_2)-action rule r in S, denoted by $Conf_{S_{r_1,r_2}}(r)$, we mean $[Sup_{S_{r_1,r_2}}(r)/Sup_S(r_1)]$.

Let us go back again to the example of a decision table S 2. Assume for now that only attributes b, d are flexible in S, where $V_b = \{S, R, P\}$ and $V_d = \{L, H\}$. To generate all possible action rules directly from S which may reclassify objects from class L to H, we have to consider all combinations of changes

involving values of flexible attributes. In our simplified example, they are: $S \rightarrow R$, $S \rightarrow P$, $R \rightarrow S$, $P \rightarrow S$, $R \rightarrow P$, $P \rightarrow R$. In general, if $\{b_1, b_2, ..., b_p\}$ is the list of all flexible attributes in S excluding a binary decision attribute d and $card(V_{b_i}) = m_i$, $(i = 1, 2, ..., p)$, then we have $2^p - 1$ different classes of action rules to be considered. Each class is labelled by a subset of $\{b_1, b_2, ..., b_p\}$. Let $\{b_{i_1}, b_{i_2}, ..., b_{i_q}\}$ be one of such subsets. The number of possible action rules associated with that subset is equal to $2^q \cdot \prod_{j=1}^{q}[m_{i_j} \cdot (m_{i_j} - 1)]$. Clearly, having such a huge number of possible action rules we can not generate them one by one from the decision table. So, we have to look for alternate methods.

Let us assume that $v_i, w_i \in V_{b_i}$, $i \in \{1, 2, .., p\}$ and $k_i \in V_d$, where $i = 1, 2$. The general idea in action rules construction is to identify a set of objects satisfying properties $\{v_1, v_2, .., v_p, k_1\}$ and a set of objects satisfying properties $\{w_1, w_2, .., w_p, k_2\}$. The goal of an action rule is to reclassify objects satisfying the first set of properties to the class k2 which is more preferable state. To achieve that, we may require to change v_1 to w_1, change v_2 to w_2, and change v_p to w_p, then the expectation is that k_1 will change to k_2. To materialize such an expectation, we need an association between elements in $\{v_1, v_2, , v_p\}$ and k_1 and another association between elements in $\{w_1, w_2, , w_p\}$ and k_2. Otherwise, we wouldn't succeed. Basically, two classification rules have to be hidden in an action rule. One classification rule is linked with the left hand side of an action rule and the other one with its right hand side. To generate classification rules we have to use some knowledge discovery strategy. But, the classical KD strategies generate only the shortest or close to the shortest classification rules. Therefore, we have to develop a new algorithm for discovering all classification rules. To do that we will modify $LERS$ strategy given by [5].

3 Discovering All Action Rules

In this section, we show how to modify $LERS$ algorithm so it does not extract only the shortest classification rules but all of them. The original $LERS$ algorithm is placing a mark on the inclusion $[t^* \subseteq a_i^*]$, if the corresponding classification rule $[t \rightarrow a_i]$ satisfies the threshold for minimal confidence and support. It means, after finding that rule, the term t is not extended further to guarantee that the discovered classification rules are the shortest. The modified classification rules mining algorithm (called *E-LERS*) allows the extension of terms already marked. Since we are not placing any marks on constructed terms, the algorithm generates all classification rules. We start with an algorithm, called *S-LERS*, which is a modification of $LERS$.

Let us assume that $S = (X, A \cup \{d\}, V)$ is a decision system, where X is a set of objects, $A = \{a[i] : 1 \leq i \leq I\}$ is a set of classification attributes, $V_i = \{a[i, j] : 1 \leq j \leq J(i)\}$ is a set of values of attribute $a[i]$, $i \leq I$. We also assume that d is a decision attribute, where $V_d = \{d[m] : 1 \leq m \leq M\}$.

Finally, we assume that $a(i_1, j_1) \cdot a(i_2, j_2) \cdot ... \cdot a(i_r, j_r)$ is denoted by term $[a(i_k, j_k)]_{k \in \{1,2,...,r\}}$, where all $i_1, i_2, ..., i_r$ are distinct integers and $j_p \leq J(i_p)$,

$1 \leq p \leq r$.

Algorithm for Extracting Classification Rules from Incomplete Decision System S

S-LERS$(S, \lambda_1, \lambda_2, L(D))$;

$S = (X, A, V)$ - incomplete decision system,
λ_1 - threshold for minimum support,
λ_2 - threshold for minimum confidence,
$L(D)$ set of rules discovered from S by *S-LERS*.

begin
$i := 1$;
while $i \leq I$ **do**
begin
$j := 1$; $m := 1$
while $j \leq J[i]$ **do**
begin
if $sup(a[i,j] \rightarrow d(m)) < \lambda_1$ **then** $mark[a[i,j]^\star \preceq d(m)^\star]$;
if $sup(a[i,j] \rightarrow d(m)) \geq \lambda_1$ and $conf(a[i,j] \rightarrow d(m)) \geq \lambda_2$ **then**
begin
$mark[a[i,j]^\star \preceq d(m)^\star]$;
$output[a[i,j] \rightarrow d(m)]$
end
$j := j + 1$
end
end
$I_k := \{i_k\}$; /i_k - index randomly chosen from $\{1, 2, ..., I\}$/
for all $j_k \leq J(i_k)$ **do** $a[(i_k, j_k)]_{i_k \in I_k} := a(i_k, j_k)$;
for all i, j **such that** both rules
$a[(i_k, j_k)]_{i_k \in I_k} \rightarrow d(m)$,
$a[i,j] \rightarrow d(m)$ are not marked and $i \notin I_k$ **do**
begin
if $sup(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j] \rightarrow d(m)) < \lambda_1$
then $mark[(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j])^\star \preceq d(m)^\star]$;
if $sup(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j] \rightarrow d(m)) \geq \lambda_1$ and
$conf(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j] \rightarrow d(m)) \geq \lambda_2$ **then**
begin
$mark[(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j])^\star \preceq d(m)^\star]$;
$output[a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j] \rightarrow d(m)]$
end
else
begin
$I_k := I_k \cup \{i\}$;
$a[(i_k, j_k)]_{i_k \in I_k} := a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i,j]$
end

end

Again, let us consider the decision system S represented by Table 2 and let us discover rules from it using the above algorithm.

It can be easily checked the the following certain classification rules are generated:

- $r_1 = [(B, P) \rightarrow (D, L)]$; Sup($r_1$)=2
- $r_2 = [(C, 0) \rightarrow (D, L)]$; Sup($r_2$)=2
- $r_3 = [(A, 0) \wedge (B, S) \rightarrow (D, L)]$; Sup($r_3$)=2
- $r_4 = [(B, S) \wedge (C, 1) \rightarrow (D, L)]$; Sup($r_4$)=1
- $r_5 = [(B, R) \rightarrow (D, H)]$; Sup($r_5$)=2
- $r_6 = [(B, S) \wedge (C, 2) \rightarrow (D, H)]$; Sup($r_6$)=2

Five action rules are constructed from these six classification rules:

- $[(B, P \rightarrow R) \rightarrow (D, L \rightarrow H)]$; Sup=2
- $[(B, P \rightarrow S) \rightarrow (D, L \rightarrow H)]$; Sup=2
- $[(C, 0 \rightarrow 2) \rightarrow (D, L \rightarrow H)]$; Sup=2
- $[(B, S \rightarrow R) \rightarrow (D, L \rightarrow H)]$; Sup=2
- $[(C, 1 \rightarrow 2) \rightarrow (D, L \rightarrow H)]$; Sup=1

Now, we give *E-LERS* algorithm which differs from *S-LERS* by marking only terms in *S-LERS* which have support below the threshold value. We do not mark terms from which classification rules are constructed. This algorithm is given below:

Algorithm for Extracting All Classification Rules from Incomplete Decision System S
E-LERS$(S, \lambda_1, \lambda_2, L(D))$;
$S = (X, A, V)$ - incomplete decision system,
λ_1 - threshold for minimum support,
λ_2 - threshold for minimum confidence,
$L(D)$ set of rules discovered from S by *E-LERS*.
begin
$i := 1$;
while $i \leq I$ **do**
begin
$j := 1$; $m := 1$
while $j \leq J[i]$ **do**
begin
if $sup(a[i, j] \rightarrow d(m)) < \lambda_1$ **then** $mark[a[i, j]^\star \preceq d(m)^\star]$;
if $sup(a[i, j] \rightarrow d(m)) \geq \lambda_1$ and $conf(a[i, j] \rightarrow d(m)) \geq \lambda_2$ **then**
$output[a[i, j] \rightarrow d(m)]$
$j := j + 1$
end

end
$I_k := \{i_k\}$; /i_k - index randomly chosen from $\{1, 2, ..., I\}$/
for all $j_k \leq J(i_k)$ **do** $a[(i_k, j_k)]_{i_k \in I_k} := a(i_k, j_k)$;
for all i, j **such that** both rules
$a[(i_k, j_k)]_{i_k \in I_k} \rightarrow d(m)$,
$a[i, j] \rightarrow d(m)$ are not marked and $i \notin I_k$ **do**
begin
if $sup(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i, j] \rightarrow d(m)) < \lambda_1$
then $mark[(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i, j])^\star \preceq d(m)^\star]$;
if $sup(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i, j] \rightarrow d(m)) \geq \lambda_1$ and
$conf(a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i, j] \rightarrow d(m)) \geq \lambda_2$ **then**
$output[a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i, j] \rightarrow d(m)]$
else
begin
$I_k := I_k \cup \{i\}$;
$a[(i_k, j_k)]_{i_k \in I_k} := a[(i_k, j_k)]_{i_k \in I_k} \cdot a[i, j]$
end
end

It can be easily checked that nineteen certain classification rules are generated by *E-LERS*. Clearly, all six classification rules extracted by *S-LERS* are included in this new set. Five new classification rules extracted by *E-LERS* are given below:

- $r_7 = [(A, 0) \wedge (C, 0) \rightarrow (D, L)]$; Sup($r_7$)=1
- $r_8 = [(A, 2) \wedge (B, P) \rightarrow (D, L)]$; Sup($r_8$)=2
- $r_9 = [(A, 2) \wedge (C, 0) \rightarrow (D, L)]$; Sup($r_9$)=1
- $r_{10} = [(A, 0) \wedge (B, S) \wedge (C, 1) \rightarrow (D, L)]$; Sup($r_{10}$)=1
- $r_{11} = [(A, 0) \wedge (B, S) \wedge (C, 0) \rightarrow (D, L)]$; Sup($r_{11}$)=1

Eight action rules can be constructed by using all nineteen classification rules generated by *E-LERS* strategy. Clearly five action rules constructed from rules generated by *S-LERS* are included in that new set. The new action rules are listed below:

- $[(C, 0 \rightarrow 1) \rightarrow (D, L \rightarrow H)]$; Sup=2
- $[[(B, S \rightarrow R) \wedge (C, 0 \rightarrow 1)] \rightarrow (D, L \rightarrow H)]$; Sup=2
- $[[(B, P \rightarrow R) \wedge (C, 2 \rightarrow 1)] \rightarrow (D, L \rightarrow H)]$; Sup=2

This example shows that some action rules constructed from classification rules generated by *E-LERS* are specifications of action rules constructed from classification rules generated by *S-LERS*. As an example, we can use rules: $[[(B, S \rightarrow R) \wedge (C, 0 \rightarrow 1)] \rightarrow (D, L \rightarrow H)]$ which is *E-LERS*-based and $[(B, S \rightarrow R) \rightarrow (D, L \rightarrow H)]$ which is *LERS*-based. However some action rules constructed from classification rules generated by *E-LERS* are independent from action rules constructed from classification rules generated by *S-LERS*. As an example, we can take: $[(C, 0 \rightarrow 1) \rightarrow (D, L \rightarrow H)]$. It shows that either all

classification rules have to be used to generate all action rules or an alternative strategy similar to the concept of representative rules in a class of association rules is needed.

The definition of *support* of (r_1, r_2)-action rule r in S is not very pleasant because it requires rules r_1, r_2 to be extracted first from S. In order to construct all action rules based on S, we need to consider all pairs of rules extracted from S and pick up only those which satisfy the condition required for action rules construction. Then, from each of these pairs we construct an action rule and calculate its support and confidence. To speed up the process of action rules construction, we can concentrate on action rules which are built from certain pairs of rules extracted from S. For instance, we can consider pairs of rules not only satisfying the required condition for action rules construction but also which have a small number of overlapping flexible attributes in their conditional part. This way we minimize the number of attribute values required by an action rule to be changed for any object supporting that rule. But the question is if action rules constructed that way can be seen as a set of generators for the remaining rules?

4 Discovering Interesting Action Rules

Assume that $S = (U, A_1 \cup A_2 \cup \{d\})$. Let $b \in A_1$ is a flexible attribute and b_1, b_2 are its two values. By $\rho_S(b_1, b_2)$ we mean a number from $(0, +\infty]$ which describes the average cost to change the attribute value from b_1 to b_2 for any of the qualifying objects in S. Object $x \in U$ qualifies for the change from b_1 to b_2, if $b(x) = b_1$. If the above change is not feasible for one of the qualifying objects in S, then we write $\rho_S(b_1, b_2) = +\infty$. The value of $\rho_S(b_1, b_2)$ close to *zero* is interpreted that the change of values from b_1 to b_2 is trivial to accomplish for qualifying objects in S whereas any large value of $\rho_S(b_1, b_2)$ means that this change of values is practically very difficult to get for some of the qualifying objects in S.

If $\rho_S(b_1, b_2) < \rho_S(b_3, b_4)$, then we say that the change of values from b_1 to b_2 is *more feasible* than the change from b_3 to b_4.

We assume that the values $\rho_S(b_{j1}, b_{j2})$ are provided by experts. They are treated as atomic expressions and used later to introduce the notion of the feasibility and the cost of action rules in S.

Let us assume that $r = [(b_1, v_1 \rightarrow w_1) \wedge (b_2, v_2 \rightarrow w_2) \wedge \wedge (b_p, v_p \rightarrow w_p)](x) \Rightarrow (d, k_1 \rightarrow k_2)(x)$ is a (r_1, r_2)-action rule. By the *cost* of r denoted by $cost(r)$ we mean the value $\sum\{\rho_S(v_k, w_k) : 1 \leq k \leq p\}$. We say that r is *feasible* if $cost(r) < \rho_S(k_1, k_2)$.

Assume now that user would like to re-classify some customers in S from the group k_1 to the group k_2, where $k_1, k_2 \in V_d$. We may look for an appropriate action rule, possibly of the lowest cost value, to get a hint which attribute values need to be changed. Another words, let us assume that $R_S[(d, k_1 \rightarrow k_2)]$ denotes

the set of all action rules in S having the term $(d, k_1 \rightarrow k_2)$ on their decision site. Now, among all action rules in $R_S[(d, k_1 \rightarrow k_2)]$ we may identify a rule which has the lowest cost value. But the rule we get may still have the cost value much to high to be of any help to us. Let us notice that the cost of the action rule

$r = [(b_1, v_1 \rightarrow w_1) \wedge (b_2, v_2 \rightarrow w_2) \wedge ... \wedge (b_p, v_p \rightarrow w_p)](x) \Rightarrow (d, k_1 \rightarrow k_2)(x)$

might be high only because of the high cost value of one of its sub-terms in the conditional part of the rule.

Let us assume that $(b_j, v_j \rightarrow w_j)$ is that term. In such a case, we may look for an action rule in $R_S[(b_j, v_j \rightarrow w_j)]$ which has the smallest cost value.

Assume that $r_1 = [(b_{j1}, v_{j1} \rightarrow w_{j1}) \wedge (b_{j2}, v_{j2} \rightarrow w_{j2}) \wedge ... \wedge (b_{jq}, v_{jq} \rightarrow w_{jq})](y) \Rightarrow (b_j, v_j \rightarrow w_j)(y)$ is such a rule which is also feasible in S. Since $x, y \in U$, we can compose r with r_1 getting a new feasible rule which is given below:

$[(b_1, v_1 \rightarrow w_1) \wedge ... \wedge [(b_{j1}, v_{j1} \rightarrow w_{j1}) \wedge (b_{j2}, v_{j2} \rightarrow w_{j2}) \wedge ... \wedge (b_{jq}, v_{jq} \rightarrow w_{jq})] \wedge ... \wedge (b_p, v_p \rightarrow w_p)](x) \Rightarrow (d, k_1 \rightarrow k_2)(x)$.

Clearly, the cost of this new rule is lower than the cost of r. However, if its support in S gets too low, then such a rule is useless for us. Otherwise, we may recursively follow this strategy trying to lower the cost of re-classifying objects from the group k_1 into the group k_2. Each successful step will produce a new action rule which cost is lower than the cost of the current rule. This heuristic strategy was presented in [18] as an $A^\star$-algorithm generating action rules of the cost below some user specified threshold value. Such rules are called interesting rules.

5 Conclusion

System *E-LERS* generates all classification rules from S (satisfying two thresholds, the first one for a minimum support and second for a minimum confidence) defining values of a decision attribute in S, in terms of the remaining attributes. These classification rules are used by *DEAR* to generate all action rules. So, the current strategy requires the generation of classification rules from S to form a base, before the process of action rules construction starts.

Alternate strategy is to partition the set of action rules using the notion of a cost of rule proposed in [18]. Next we generate the set of action rules following *DEAR* strategy based on classical classification rules. This set is seen as the set of generators to which A^*-search strategy proposed in [18] is applied to generate new action rules, classified as the most interesting rules with respect to the cost. This approach although does not generate all action rules, it generates rules which seems to be the mostly desirable by user.

References

1. Adomavicius, G., Tuzhilin, A., 1997, Discovery of actionable patterns in databases: the action hierarchy approach. In: Proceedings: KDD'97 Conference, Newport Beach, CA, AAAI Press.
2. Chmielewski, M.R., Grzymala-Busse J. W., Peterson N. W., Than S., 1993, The Rule Induction System LERS - a version for personal computers, *International Journal of Approximate Reasoning*, **18**(3-4), Institute of Computing Science, Technical University of Poznań, Poland, pp. 181-212.
3. Dardzińska, A., Raś, Z.W., 2003, On Rule Discovery from Incomplete Information Systems. In: Proceedings: ICDM'03 Workshop on Foundations and New Directions of Data Mining, T.Y. Lin, X. Hu, S. Ohsuga, C. Liau (eds), Melbourne, Florida, IEEE Computer Society, pp. 31-35.
4. Geffner, H., Wainer, J., 1998, Modeling action, knowledge and control. In: Proceedings: ECAI'98, the 13th European Conference on AI, H. Prade (ed), John Wiley & Sons, pp. 532-536.
5. Grzymala-Busse, J., 1997, A new version of the rule induction system LERS, *Fundamenta Informaticae*, **31**(1), pp. 27-39.
6. Kryszkiewicz. M., 1998, Fast discovery of representative association rules, In: Lecture Notes in Artificial Intelligence, Vol. 1424, Proceedings of RSCTC 98, Springer-Verlag, 214-221.
7. Liu, B., Hsu, W., Chen, S., 1997, Using general impressions to analyze discovered classification rules, In: Proceedings: KDD'97 Conference, Newport Beach, CA, AAAI Press.
8. Pawlak, Z., 1991, *Rough sets-theoretical aspects of reasoning about data*, Kluwer, Dordrecht.
9. Pawlak, Z., 1981, Information systems - theoretical foundations, *Information Systems Journal*, **6**, pp. 205-218.
10. Polkowski, L., Skowron A., 1998, Rough sets in knowledge discovery, In: *Studies in Fuzziness and Soft Computing*, Physica-Verlag, Springer.
11. Raś, Z.W., Tzacheva, A., Tsay, L.-S., 2005, Action Rules, In: *Encyclopedia of Data Warehousing and Mining*, J. Wang (ed.), Idea Group Inc., will appear.
12. Raś, Z., Wieczorkowska, A., 2000, Action Rules: how to increase profit of a company, In: *Principles of Data Mining and Knowledge Discovery*, D.A. Zighed, J. Komorowski, J. Zytkow (eds.), Proceedings: PKDD'00, Lyon, France, LNCS/LNAI, No. 1910, Springer-Verlag, pp. 587-592.
13. Raś, Z.W., Tsay, L.-S., 2003, Discovering Extended Action-Rules (System DEAR), In: *Intelligent Information Systems 2003*, Proceedings: IIS'03 Symposium, Zakopane, Poland, Advances in Soft Computing, Springer-Verlag, pp. 293-300.
14. Raś, Z., Gupta, S., 2002, Global action rules in distributed knowledge systems, in *Fundamenta Informaticae Journal*, IOS Press, **51**(1-2), pp. 175-184.
15. Silberschatz, A., Tuzhilin, A., 1995, On subjective measures of interestingness in knowledge discovery, In: Proceedings: KDD'95 Conference, AAAI Press.
16. Silberschatz, A., Tuzhilin, A., 1996, What makes patterns interesting in knowledge discovery systems, *IEEE Transactions on Knowledge and Data Engineering*, **5**(6).
17. , Suzuki, E., Kodratoff, Y., 1998, Discovery of surprising exception rules based on intensity of implication, in *Proc. of the Second Pacific-Asia Conference on Knowledge Discovery and Data mining (PAKDD)*

18. Tzacheva, A., Raś, Z.W., 2004, Action rules: feasibility and lowest cost reclassification, In: Special Issue on Knowledge Discovery, Z.W. Ras (ed.), *International Journal of Intelligent Systems*, Wiley, will appear.
19. Tzacheva, A., Raś, Z.W., 2003, Discovering non-standard semantics of semi-stable attributes, In: Proceedings: FLAIRS'03 Conference, St. Augustine, Florida, I. Russell, S. Haller (eds.), AAAI Press, pp. 330-334.