Action Rules Discovery Based on Tree Classifiers and Meta-actions

Zbigniew W. Raś[1,2] and Agnieszka Dardzińska[3]

[1] Univ. of North Carolina, Dept. of Computer Science, Charlotte, NC, 28223, USA
[2] Polish-Japanese Institute of Information Technology, 02-008 Warsaw, Poland
[3] Bialystok Technical Univ., Dept. of Computer Science, 15-351 Bialystok, Poland
ras@uncc.edu, adardzin@uncc.edu

Abstract. Action rules describe possible transitions of objects from one state to another with respect to a distinguished attribute. Early research on action rule discovery usually required the extraction of classification rules before constructing any action rule. Newest algorithms discover action rules directly from a decision system. To our knowledge, all these algorithms assume that all attributes are symbolic or require prior discretization of all numerical attributes. This paper presents a new approach for generating action rules from datasets with numerical attributes by incorporating a tree classifier and a pruning step based on meta-actions. Meta-actions are seen as a higher-level knowledge (provided by experts) about correlations between different attributes.

1 Introduction

An action rule is a rule extracted from an information system that describes a possible transition of objects from one state to another with respect to a distinguished attribute called a decision attribute [13]. Attributes used to describe objects are partitioned into stable and flexible. Values of flexible attributes can be changed. This change can be influenced and controlled by users. Action rules mining initially was based on comparing profiles of two groups of targeted objects - those that are desirable and those that are undesirable [13]. An action rule was defined as a term $[(\omega) \wedge (\alpha \rightarrow \beta)] \Rightarrow (\phi \rightarrow \psi)$, where ω is the header and it is a conjunction of fixed classification features shared by both groups, $(\alpha \rightarrow \beta)$ represents proposed changes in values of flexible features, and $(\phi \rightarrow \psi)$ is a desired effect of the action. The discovered knowledge provides an insight of how values of some attributes need to be changed so the undesirable objects can be shifted to a desirable group. For example, one would like to find a way to improve his or her salary from a low-income to a high-income. Action rules tell us what changes within flexible attributes are needed to achieve that goal.

Meta-actions are defined as actions which trigger changes of flexible attributes either directly or indirectly because of correlations among certain attributes in the system. Links between meta-actions and changes they trigger within the values of flexible attributes can be defined by an ontology [3] or by a mapping linking meta-actions with changes of attribute values used to describe objects in the decision system. In medical domain, taking a drug is a classical example of a meta-action. For instance, Lamivudine is used for treatment of chronic hepatitis B. It improves the seroconversion of e-antigen

J. Rauch et al. (Eds.): ISMIS 2009, LNAI 5722, pp. 66–75, 2009.

positive hepatitis B and also improves histology staging of the liver but at the same time it can cause a number of other symptoms. This is why doctors have to order certain lab tests to check patient's response to that drug. Clearly, the cost of a drug is known.

The concept of an action rule was proposed in [13] and investigated further in [18] [19] [8] [14] [4] [10] [17]. Paper [6] was probably the first attempt towards formally introducing the problem of mining action rules without pre-existing classification rules. Authors explicitly formulated it as a search problem in a support-confidence-cost framework. The proposed algorithm has some similarity with Apriori [1]. Their definition of an action rule allows changes on stable attributes. Changing the value of an attribute, either stable or flexible, is linked with a cost [19]. In order to rule out action rules with undesired changes on attributes, authors assigned very high cost to such changes. However, that way, the cost of action rules discovery is getting unnecessarily increased. Also, they did not take into account the correlations between attribute values which are naturally linked with the cost of rules used either to accept or reject a rule.

Algorithm *ARED*, presented in [7], is based on Pawlak′s model of an information system S [9]. The goal was to identify certain relationships between granules defined by the indiscernibility relation on its objects. Some of these relationships uniquely define action rules for S. Paper [11] presents a strategy for discovering action rules directly from the decision system. Action rules are built from atomic expressions following a strategy similar to *LERS* [5]. In [19], authors introduced the cost of action rules and use it in the pruning step of the rule discovery process. Paper [20] introduced the notion of $action$ as a domain- independent way to model the domain knowledge. Given a data set about actionable features and an utility measure, a pattern is actionable if it summarizes a population that can be acted upon towards a more promising population observed with a higher utility. Algorithms for mining actionable patterns (changes within flexible attributes) take into account only numerical attributes. The distinguished (decision) attribute is called utility. Each action A_i triggers changes of attribute values described by terms [$a \downarrow$], [$b \uparrow$], and [c (don't know)]. They are represented as an influence matrix built by an expert. While previous approaches used only features - mined directly from the decision system, authors in [20] define actions as its foreign concepts. Influence matrix shows the link between actions and changes of attribute values and the same shows correlations between some attributes, i.e. if [$a \downarrow$], then [$b \uparrow$]. Clearly, expert does not know correlations between classification attributes and the decision attribute. Such correlations can be seen as action rules and they are discovered from the decision system. So, the definition of an action rule in [20] only refers to the increase/decrease of values of numerical attribute and the process of constructing action rules does not take into consideration neither their cost nor stable attributes.

This paper extends the definition of action rules [12] to numerical attributes and presents a new approach for discovering them from decision systems by incorporating a tree classifier, cost of action rules [19], and the pruning step based on meta-actions.

2 Background and Objectives

In this section we introduce the notion of an information system, meta-action and give examples.

By an information system [9] we mean a triple $S = (X, A, V)$, where:

1. X is a nonempty, finite set of objects
2. 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
3. $V = \bigcup\{V_a : a \in A\}$.

For example, Table 1 shows an information system S with a set of objects $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$, set of attributes $A = \{a, b, c, d\}$, and a set of their values $V = \{a_1, a_2, b_1, b_2, b_3, c_1, c_2, d_1, d_2, d_3\}$.

Table 1. Information System S

	a	b	c	d
x_1	a_1	b_1	c_1	d_1
x_2	a_2	b_1	c_2	d_1
x_3	a_2	b_2	c_2	d_1
x_4	a_2	b_1	c_1	d_1
x_5	a_2	b_3	c_2	d_1
x_6	a_1	b_1	c_2	d_2
x_7	a_1	b_2	c_2	d_1
x_8	a_1	b_2	c_1	d_3

An information system $S = (X, A, V)$ is called a decision system, if one of the attributes in A is distinguished and called the decision. The remaining attributes in A are classification attributes. Additionally, we assume that $A = A_{St} \cup A_{Fl} \cup \{d\}$, where attributes in A_{St} are called *stable* whereas in A_{Fl} are called *flexible*. Attribute d is the decision attribute. “Date of birth” is an example of a stable attribute. “Interest rate” for each customer account is an example of a flexible attribute.

By meta-actions associated with S we mean higher level concepts representing actions introduced in [20]. Meta-actions, when executed, are expected to trigger changes in values of some flexible attributes in S as described by influence matrix [20]. To give an example, let us assume that classification attributes in S describe teaching evaluations at some school and the decision attribute represents their overall score. Examples of classification attributes are: *Explain difficult concepts effectively*, *Stimulate student interest in the course*, *Provide sufficient feedback*. Then, examples of meta-actions associated with S will be: *Change the content of the course*, *Change the textbook of the course*, *Post all material on the Web*. The influence matrix [20] is used to describe the relationship between meta-actions and the expected changes within classification attributes. It should be mentioned here that expert knowledge concerning meta-actions involves only classification attributes. Now, if some of these attributes are correlated with the decision attribute, then any change in their values will cascade to the decision attribute through this correlation. The goal of an action rule discovery is to identify all correlations between classification attributes and the decision attribute.

In earlier works in [13] [18] [19] [7] [14], action rules are constructed from classification rules. This means that we use pre-existing classification rules to construct action rules either from certain pairs of these rules or from a single classification rule. For instance, algorithm $ARAS$ [14] generates sets of terms (built from values of attributes) around classification rules and constructs action rules directly from them. In [12] authors presented a strategy for extracting action rules directly from a decision system and without using pre-existing classification rules.

In the next section, we introduce the notion of action sets, action rules [12], the cost of an action rule, and the notion of an influence matrix (see [20]) associated with a set of meta-actions. The values stored in an influence matrix are action sets.

3 Action Rules and Meta-actions

Let $S = (X, A, V)$ is an information system, where $V = \bigcup\{V_a : a \in A\}$. First, we modify the notion of an atomic action set given in [11] so it may include numerical attributes.

By an *atomic action set* we mean any of the three expressions:

1. $(a, a_1 \rightarrow a_2)$, where a is a symbolic attribute and $a_1, a_2 \in V_a$,
2. $(a, [a_1, a_2] \uparrow [a_3, a_4])$, where a is a numerical attribute, $a_1 \leq a_2 < a_3 \leq a_4$, and $(\forall)[(1 \leq i \leq 4) \rightarrow (a_i \in V_a)]$,
3. $(a, [a_1, a_2] \downarrow [a_3, a_4])$ where a is a numerical attribute, $a_3 \leq a_4 < a_1 \leq a_2$, and $(\forall)[(1 \leq i \leq 4) \rightarrow (a_i \in V_a)]$.

If a is symbolic and $a_1 = a_2$, then a is called stable on a_1. Instead of $(a, a_1 \rightarrow a_1)$, we often write (a, a_1) for any $a_1 \in V_a$. The term $(a, a_1 \rightarrow a_2)$ should be read as *“the value of attribute a is changed from a_1 to a_2”*. The term $(a, [a_1, a_2] \uparrow [a_3, a_4])$ should be read as *“the value of attribute a from to the interval $[a_1, a_2]$ is increased and it belongs now to the interval $[a_3, a_4]$”*. Similarly, the term $(a, [a_1, a_2] \downarrow [a_3, a_4])$ should be read as *“the value of attribute a from to the interval $[a_1, a_2]$ is decreased and it belongs now to the interval $[a_3, a_4]$”*.

Also, a simplified version of the atomic action set will be used. It includes such expressions as: $(a, \uparrow [a_3, a_4])$, $(a, [a_1, a_2] \uparrow)$, $(a, \downarrow [a_3, a_4])$, and $(a, [a_1, a_2] \downarrow)$.

Any collection of atomic action sets is called a candidate action set. If a candidate action set does not contain two atomic action sets referring to the same attribute, then it is called an *action set*. Clearly, $\{(b, b_2), (b, [b_1, b_2] \uparrow [b_3, b_4])\}$ is an example of a candidate action set which is not an action set. By the domain of action set t, denoted by $Dom(t)$, we mean the set of all attribute names listed in t. For instance if $\{(a, a_2), (b, b_1 \rightarrow b_2)\}$ is the action set, then its domain is equal to $\{a, b\}$. By an *action term* we mean a conjunction of atomic action sets forming an action set. There is some similarity between atomic action sets and atomic expressions introduced in [15], [20].

Now, assume that M_1 is a meta-action triggering the action set $\{(a, a_2), (b, b_1 \rightarrow b_2)\}$ and M_2 is a meta-action triggering the atomic actions in $\{(a, a_2), (b, b_2 \rightarrow b_1)\}$. It means that M_1 and M_2 involve attributes a, b with attribute a remaining stable. The corresponding action terms are: $(a, a_2) \cdot (b, b_1 \rightarrow b_2)$ associated with M_1 and $(a, a_2) \cdot (b, b_2 \rightarrow b_1)$ associated with M_2.

Consider a set of meta-actions $\{M_1, M_2, ..., M_n\}$ associated with a decision system $S = (X, A \cup \{d\}, V)$. Each meta-action M_i may trigger changes of some attribute values for objects in S. We assume here that $A - \{d\} = \{A_1, A_2, ..., A_m\}$. The influence of a meta-action M_i on attribute A_j in S is represented by an atomic action set $E_{i,j}$. The influence of meta-actions $\{M_1, M_2, ..., M_n\}$ on the classification attributes in S is described by the influence matrix $\{E_{i,j} : 1 \le i \le n \wedge 1 \le j \le m\}$. There is no expert knowledge about what is their correlation with the decision attribute in S.

By *action rule* in S we mean any expression $r = [t_1 \Rightarrow (d, d_1 \rightarrow d_2)]$, where t_1 is an action set in S and d is the decision attribute. The domain of action rule r is defined as $Dom(t_1) \cup \{d\}$.

Now, we give an example of action rules assuming that an information system S is represented by Table 1, a, c, d are flexible attributes and b is stable. Expressions (a, a_2), (b, b_2), $(c, c_1 \rightarrow c_2)$, $(d, d_1 \rightarrow d_2)$ are examples of atomic action sets. Expression r = $[[(a, a_2) \cdot (c, c_1 \rightarrow c_2)] \Rightarrow (d, d_1 \rightarrow d_2)]$ is an example of an action rule. The rule says that if value a_2 remains unchanged and value c will change from c_1 to c_2, then it is expected that the value d will change from d_1 to d_2. The domain $Dom(r)$ of action rule r is equal to $\{a, c, d\}$. For simplicity reason, our example does not cover numerical attributes and the same we do not consider such terms as $(a, [a_1, a_2] \downarrow [a_3, a_4])$ or $(a, [a_1, a_2] \uparrow [a_3, a_4])$ which are also constructed by hierarchical classifiers.

Standard interpretation N_S of action terms in $S = (X, A, V)$ is defined as follow:

1. If $(a, a_1 \rightarrow a_2)$ is an atomic action set, then
   $N_S((a, a_1 \rightarrow a_2)) = [\{x \in X : a(x) = a_1\}, \{x \in X : a(x) = a_2\}]$.
2. If $(a, [a_1, a_2] \downarrow [a_3, a_4])$ is an atomic action set, then
   $N_S((a, [a_1, a_2] \downarrow [a_3, a_4])) = [\{x \in X : a_1 \le a(x) \le a_2\}, \{x \in X : a_3 \le a(x) \le a_4\}]$.
3. If $(a, [a_1, a_2] \uparrow [a_3, a_4])$ is an atomic action set, then
   $N_S((a, [a_1, a_2] \uparrow [a_3, a_4])) = [\{x \in X : a_1 \le a(x) \le a_2\}, \{x \in X : a_3 \le a(x) \le a_4\}]$.
4. If $(a, \uparrow [a_3, a_4])$ is an atomic action set, then
   $N_S((a, \uparrow [a_3, a_4])) = [\{x \in X : a(x) < a_3\}, \{x \in X : a_3 \le a(x) \le a_4\}]$.
5. If $(a, [a_1, a_2] \uparrow)$ is an atomic action set, then
   $N_S((a, [a_1, a_2] \uparrow)) = [\{x \in X : a_1 \le a(x) \le a_2\}, \{x \in X : a_2 < a(x)\}]$.
6. If $(a, \downarrow [a_3, a_4])$ is an atomic action set, then
   $N_S((a, \downarrow [a_3, a_4])) = [\{x \in X : a(x) < a_3\}, \{x \in X : a_3 \le a(x) \le a_4\}]$.
7. If $(a, [a_1, a_2] \downarrow)$ is an atomic action set, then
   $N_S((a, [a_1, a_2] \downarrow)) = [\{x \in X : a_1 \le a(x) \le a_2\}, \{x \in X : a_2 < a(x)\}]$.
8. If $t_1 = t_2 \cdot t$ is an action term, t_2 is an atomic action set, $N_S(t_2) = [Z_1, Z_2]$, and $N_S(t) = [Y_1, Y_2]$, then $N_S(t_1) = [Z_1 \cap Y_1, Z_2 \cap Y_2]$.

If t is an action rule and $N_S(t) = \{Y_1, Y_2\}$, then the support of t in S is defined as $sup(t) = min\{card(Y_1), card(Y_2)\}$.

Now, let $r = [t_1 \Rightarrow t_2]$ is an action rule, where $N_S(t_1) = [Y_1, Y_2]$, $N_S(t_2) = [Z_1, Z_2]$. Support and confidence of r are defined as:

1. $sup(r) = min\{card(Y_1 \cap Z_1), card(Y_2 \cap Z_2)\}$.
2. $conf(r) = [\frac{card(Y_1 \cap Z_1)}{card(Y_1)}] \cdot [\frac{card(Y_2 \cap Z_2)}{card(Y_2)}]$.

The definition of a confidence requires that $card(Y_1) \neq 0$ and $card(Y_2) \neq 0$. Otherwise, the confidence of an action rule is undefined.

Coming back to the example of S given in Table 1, we can find a number of action rules associated with S. Let us take r = $[[(b, b_1) \cdot (c, c_1 \rightarrow c_2)] \Rightarrow (d, d_1 \rightarrow d_2)]$ as an example of action rule. Then,

$$N_S((b, b_1)) = [\{x_1, x_2, x_4, x_6\}, \{x_1, x_2, x_4, x_6\}],$$
$$N_S((c, c_1 \rightarrow c_2)) = [\{x_1, x_4, x_8\}, \{x_2, x_3, x_5, x_6, x_7\}],$$
$$N_S((d, d_1 \rightarrow d_2)) = [\{x_1, x_2, x_3, x_4, x_5, x_7\}, \{x_6\}],$$
$$N_S((b, b_1) \cdot (c, c_1 \rightarrow c_2)) = [\{x_1, x_4\}, \{x_2, x_6\}].$$

Clearly, $sup(r) = 1$ and $conf(r) = 1 \cdot 1 = 1/2$.

The notion of a cost associated with an action rule was introduced in [19]. Some changes of values of attributes are not expensive and easy to achieve but some of them might be very costly. So, with every atomic action set t, we associate the cost $cost_S(t)$ needed to achieve the change of attribute value recommended in t for objects in S. If all atomic actions sets listed in $t = t_1 \cdot t_2 \cdot t_3 \cdot .... \cdot t_k$ are not correlated, then $cost_S(t) = \sum\{cost_S(t_i) : 1 \leq i \leq k\}$.

4 Action Rules Discovery

In this section we present the process of discovering action rules of acceptable cost from a decision system $S = (X, A \cup \{d\}, V)$ using a tree classifier and an influence matrix associated with meta-actions.

To reduce the number of values for numerical attributes in S we use a classical method based either on entropy or Gini index resulting in a hierarchical discretization. Classification attributes are partitioned into stable and flexible. Before we use any flexible attribute in the process of a decision tree construction, all stable attributes have to be used first. This way the decision table is split into a number of decision subtables leading to them from the root of the tree by uniquely defined pathes built from stable attributes [18]. Each path defines a header in all action rules extracted from the corresponding subtable. Initial testing shows that the action rules built that way are more compact (have larger intervals) than action rules built with prior discretization of the decision table done for instance by Rough Sets Exploration System [16].

For instance, let us assume that the table assigned to the root of the tree in Fig. 1 has to be converted into its decision tree representation and that we are looking for action rules of which purpose is to change the property of objects from d_3 into d_1. We also assume that both attributes in $\{a, b\}$ are stable and attributes in $\{c, e, f\}$ are flexible. Attribute $\{d\}$ is the decision attribute. Our goal is to split this table into sub-tables by taking a stable attribute with the largest entropy gain as the splitting one. In our example, we chose attribute a.

This process is recursively continued for all stable attributes (in our example only one stable attribute is left). The sub-tables corresponding to outgoing edges from the root node which are labelled by $a = 10$, $a = 3$ are removed because they do not contain decision value d_1. After all stable attributes are used, we begin splitting recursively each of the latest sub-tables by taking a flexible attribute with the largest entropy gain as the splitting one. In our example, the following pathes (from the root to a leaf) are finally

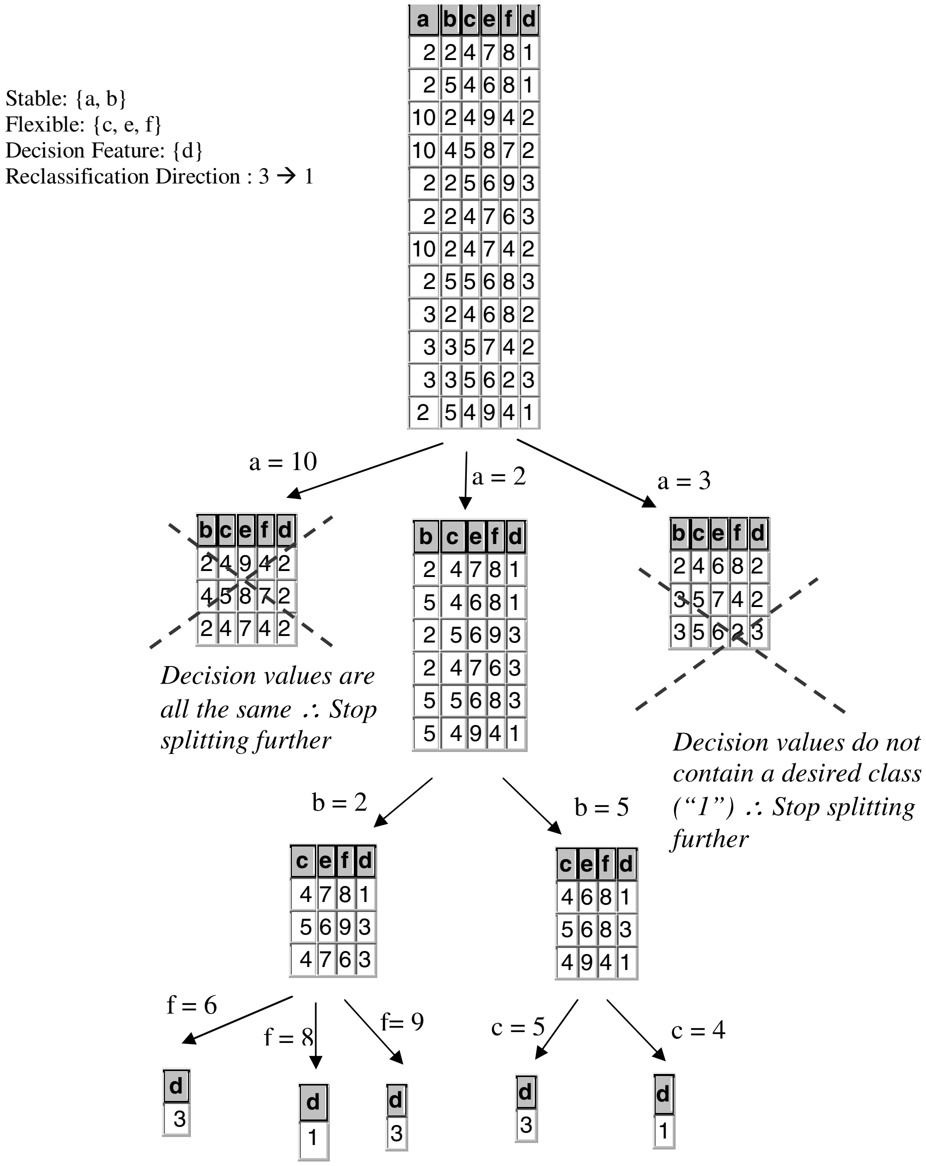

Fig. 1. Classification Tree Construction Process

built: $[[(a, 2) \wedge (b, 2)] \wedge (f, 6) \wedge (d, 3)]$, $[[(a, 2) \wedge (b, 2)] \wedge (f, 8) \wedge (d, 1)]$, $[[(a, 2) \wedge (b, 2)] \wedge (f, 9) \wedge (d, 3)]$, $[[(a, 2) \wedge (b, 5)] \wedge (c, 5) \wedge (d, 3)]$, $[[(a, 2) \wedge (b, 2)] \wedge (c, 4) \wedge (d, 1)]$.

By examining the nodes right above the leaves of the resulting decision tree, we get the following candidate action rules (based on strategy similar to *DEAR* [18]):

- $[[(a, 2) \wedge (b, 2)] \wedge (f, 6 \rightarrow 8)] \Rightarrow (d, 3 \rightarrow 1)]$,
- $[[(a, 2) \wedge (b, 2)] \wedge (f, 9 \rightarrow 8)] \Rightarrow (d, 3 \rightarrow 1)]$,
- $[[(a, 2) \wedge (b, 5)] \wedge (c, 5 \rightarrow 4)] \Rightarrow (d, 3 \rightarrow 1)]$.

Now, we can calculate the cost of each of these candidate action rules and delete all with cost below a user specified threshold.

Influence matrix associated with S and a set of meta-actions is used to identify which remaining candidate action rules are valid with respect to meta-actions and hidden correlations between classification attributes and the decision attribute.

Assume that $S = (X, A \cup \{d\}, V)$ is a decision system, $A - \{d\} = A_1 \cup A_2 \cup ... \cup A_m$, $\{M_1, M_2, ..., M_n\}$ are meta-actions associated with S, $\{E_{i,j} : 1 \leq i \leq n, 1 \leq j \leq m\}$ is the influence matrix, and $r = [(A_{[i,1]}, a_{[i,1]} \rightarrow a_{[j,1]}) \cdot (A_{[i,2]}, a_{[i,2]} \rightarrow a_{[j,2]}) \cdot ..... \cdot (A_{[i,k]}, a_{[i,k]} \rightarrow a_{[j,k]})] \Rightarrow (d, d_i \rightarrow d_j)$ is a candidate action rule extracted from S. Also, we assume here that $A_{[i,j]}(M_i) = E_{i,j}$. Value $E_{i,j}$ is either an atomic action set or *NULL*. By meta-actions based decision system, we mean a triple consisting with S, meta-actions associated with S, and the influence matrix linking them.

We say that r is valid in S with respect to meta-action M_i, if the following condition holds:

if $(\exists p \leq k)[A_{[i,p]}(M_i)$ is defined], then
$(\forall p \leq k)[$ if $A_{[i,p]}(M_i)$ is defined, then $(A_{[i,p]}, a_{[i,p]} \rightarrow a_{[j,p]})$ = $(A_{[i,p]}, E_{i,p})]$

We say that r is valid in S with respect to meta-actions $\{M_1, M_2, ..., M_n\}$, if there is i, $1 \leq i \leq n$, such that r is valid in S with respect to meta-action M_i.

To give an example, assume that S is a decision system represented by Table 1 and $\{M_1, M_2, M_3, M_4, M_5, M_6\}$ is the set of meta-actions assigned to S with an influence matrix shown in Table 2. Clearly, each empty slot in Table 2 corresponds to *NULL* value.

Table 2. Influence Matrix for S

	a	b	c
M_1		b_1	$c_2 \rightarrow c_1$
M_2	$a_2 \rightarrow a_1$	b_2	
M_3	$a_1 \rightarrow a_2$		$c_2 \rightarrow c_1$
M_4		b_1	$c_1 \rightarrow c_2$
M_5			$c_1 \rightarrow c_2$
M_6	$a_1 \rightarrow a_2$		$c_1 \rightarrow c_2$

In the example presented in previous section, two candidate action rules have been constructed:

$r1 = [[(b, b_1) \cdot (c, c_1 \rightarrow c_2)] \Rightarrow (d, d_1 \rightarrow d_2)]$ and
$r2 = [(a, a_2 \rightarrow a_1) \Rightarrow (d, d_1 \rightarrow d_2)]$.

Clearly $r1$ is valid in S with respect to M_4 and M_5. Also, r_2 is valid in S with respect to M_1, M_4, M_5 because there is no overlap between the domain of action rule r_2 and the set of attributes influenced by any of these meta-actions. However, we can not say that $r2$ is valid in S with respect to M_2 since b_2 is not listed in the classification part of r_2.

Assume that $S = (X, A \cup \{d\}, V)$ is a decision system with meta-actions $\{M_1, M_2, ..., M_n\}$ associated with S. Any candidate action rule extracted from S which is valid in a meta-actions based decision system is called action rule. So, the process of action rules discovery is simplified to checking the validity of candidate action rules.

5 Conclusion

New algorithm for action rules discovery from the data containing both symbolic and numerical attributes is presented. The method basically follows $C4.5$ or $CART$ with only one exception - before any flexible attribute is used as a splitting one, all stable attributes have to be processed first. Each path starting from the root and built from stable attributes defines a class of candidate action rules having the same heading. Meta-actions jointly with the influence matrix are used as a postprocessing tool in action rules discovery. Influence matrix shows the correlations among classification attributes triggered off by meta-actions. If the candidate actions rules are not on par with them, then they are not classified as action rules. However, the influence matrix may not show all the interactions between classification attributes, so still some of the resulting action rules may fail when tested on real data.

Acknowledgment

This work was supported by the Ministry of Science and Higher Education in Poland under Grant *N N519 404734*, by Bialystok Technical University under Grant *W/WI/2/09*, and by Polish-Japanese Institute of Information Technology under Grant *ST/SI/02/08*.

References

1. Agrawal, R., Srikant, R.: Fast algorithm for mining association rules. In: Proceeding of the Twentieth International Conference on VLDB, pp. 487–499 (1994)
2. Dardzińska, A., Raś, Z.: Extracting rules from incomplete decision systems. In: Foundations and Novel Approaches in Data Mining. Studies in Computational Intelligence, vol. 9, pp. 143–154. Springer, Heidelberg (2006)
3. Fensel, D.: Ontologies: a silver bullet for knowledge management and electronic commerce. Springer, Heidelberg (1998)
4. Greco, S., Matarazzo, B., Pappalardo, N., Slowiński, R.: Measuring expected effects of interventions based on decision rules. J. Exp. Theor. Artif. Intell. 17(1-2), 103–118 (2005)
5. Grzymala-Busse, J.: A new version of the rule induction system LERS. Fundamenta Informaticae 31(1), 27–39 (1997)
6. He, Z., Xu, X., Deng, S., Ma, R.: Mining action rules from scratch. Expert Systems with Applications 29(3), 691–699 (2005)
7. Im, S., Raś, Z.W.: Action rule extraction from a decision table: ARED, in Foundations of Intelligent Systems. In: An, A., Matwin, S., Raś, Z.W., Ślęzak, D. (eds.) Foundations of Intelligent Systems. LNCS (LNAI), vol. 4994, pp. 160–168. Springer, Heidelberg (2008)

8. Im, S., Raś, Z.W., Wasyluk, H.: Action Rule Discovery From Incomplete Data. Knowledge and Information Systems Journal (will appear, 2009)
9. Pawlak, Z.: Information systems - theoretical foundations. Information Systems Journal 6, 205–218 (1981)
10. Qiao, Y., Zhong, K., Wang, H.-A., Li, X.: Developing event-condition-action rules in real-time active database. In: Proceedings of the 2007 ACM symposium on Applied computing, pp. 511–516. ACM, New York (2007)
11. Raś, Z.W., Dardzińska, A.: Action rules discovery without pre-existing classification rules. In: Chan, C.-C., Grzymala-Busse, J.W., Ziarko, W.P. (eds.) RSCTC 2008. LNCS (LNAI), vol. 5306, pp. 181–190. Springer, Heidelberg (2008)
12. Raś, Z.W., Dardzińska, A., Tsay, L.-S., Wasyluk, H.: Association Action Rules. In: Proceedings of IEEE/ICDM Workshop on Mining Complex Data (MCD 2008), Pisa, Italy. IEEE Computer Society, Los Alamitos (2008)
13. Raś, Z.W., Wieczorkowska, A.: Action-Rules: How to increase profit of a company. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 587–592. Springer, Heidelberg (2000)
14. Raś, Z., Wyrzykowska, E., Wasyluk, H.: ARAS: Action rules discovery based on agglomerative strategy. In: Raś, Z.W., Tsumoto, S., Zighed, D.A. (eds.) MCD 2007. LNCS (LNAI), vol. 4944, pp. 196–208. Springer, Heidelberg (2008)
15. Rauch, J.: Considerations on logical calculi for dealing with knowledge in data mining. In: Advances in Data Management. Studies in Computational Intelligence. LNCS, vol. 223. Springer, Heidelberg (will appear, 2009)
16. `http://logic.mimuw.edu.pl/~rses/`
17. Suzuki, E.: Discovering Action Rules that are Highly Achievable from Massive Data. In: Proceedings of PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 713–722. Springer, Heidelberg (2009)
18. Tsay, L.-S., Raś, Z.W.: Action rules discovery system DEAR3. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS (LNAI), vol. 4203, pp. 483–492. Springer, Heidelberg (2006)
19. Tzacheva, A., Tsay, L.S.: Tree-based construction of low-cost action rules. Fundamenta Informaticae Journal 86(1-2), 213–225 (2008)
20. Wang, K., Jiang, Y., Tuzhilin, A.: Mining Actionable Patterns by Role Models. In: Proceedings of the 22nd International Conference on Data Engineering, pp. 16–25. IEEE Computer Society, Los Alamitos (2006)