Fundamenta Informaticae 59 (2004) 299–313
IOS Press

RRIA: A Rough Set and Rule Tree Based Incremental Knowledge Acquisition Algorithm

Zheng Zheng* **and Guoyin Wang**

Institute of Computer Science and Technology,

Chongqing University of Posts and Telecommunications

Chongqing, R.P. China

Bao_zhengzheng@263.net;Wanggy@cqupt.edu.cn

Abstract. As a special way in which the human brain is learning new knowledge, incremental learning is an important topic in AI. It is an object of many AI researchers to find an algorithm that can learn new knowledge quickly, based on original knowledge learned before, and in such way that the knowledge it acquires is efficient in real use. In this paper, we develop a rough set and rule tree based incremental knowledge acquisition algorithm. It can learn from a domain data set incrementally. Our simulation results show that our algorithm can learn more quickly than classical rough set based knowledge acquisition algorithms, and the performance of knowledge learned by our algorithm can be the same as or even better than classical rough set based knowledge acquisition algorithms. Besides, the simulation results also show that our algorithm outperforms ID4 in many aspects.

Keywords: rough set, rule tree, incremental learning, knowledge acquisition, data mining

1. Introduction

As a kind of a special intelligent system, human brain has superior ability in learning and discovering knowledge. It can learn new knowledge incrementally, repeatedly, and increasedly. And this learning and discovering way of humans is essential on many occasions. For example, when we are learning new knowledge at a university, we need not to again learn the knowledge we have already learned in elementary school and high school. We can update our knowledge structure according to new knowledge.

*Address for correspondence: Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, R.P. China

Based on this understanding, AI researchers are working hard to simulate this special learning way. Schlimmer and Fisher developed a decision tree based incremental learning algorithm ID4 [1, 2]. Utgoff designed ID5R algorithm [2] that is an improvement of ID4. G.Y. Wang developed several parallel neural network architectures (PNN's) [3, 4, 5], Z.T. Liu presented an incremental arithmetic for the smallest reduction of attributes [6], Z.H. Wang developed an incremental algorithm for rule extraction based on concept lattice [7], and A. Wojna presented an incremental constraint based algorithm [20], etc.

Incremental knowledge acquisition algorithms are special rule extraction methods. They can be used not only in the fields in which classical knowledge acquisition algorithms can be used but also in some other special fields, as follows. First, acquiring knowledge from huge data–sets. How to process huge data–sets is always an important topic in data mining. Unfortunately, classical knowledge acquisition algorithms are always inefficient while processing huge data-sets. With incremental knowledge acquisition algorithms, we can divide the original huge data–set into several small parts, acquire knowledge from one part of it at first and then incrementally learn the other parts, one by one. Second, we can acquire new knowledge based on the original rule set learned when learning some unseen objects rather than knowledge based on the original data set. For example, in medical domain, some knowledge is already saved in the form of rules. Original data corresponding to this knowledge might not be available. Algorithms can only be based on the rule set. In this case, traditional knowledge acquisition algorithms cannot be used to learn new knowledge, while incremental knowledge acquisition algorithms can. For example, we can adapt the reliability of some matched rules according to those unseen objects. Third, we can acquire real–time knowledge. In this case, we cannot learn the data–set as a whole because it takes too much time. We can only use some high–speed incremental knowledge acquisition algorithms.

In recent years, many rough set based algorithms for reduction of attributes, and knowledge acquisition have been developed. They are almost all based on static data. However, real databases are always dynamic. So, many researchers suggest that knowledge acquisition in databases should be incremental. Incremental arithmetic for the minimal reduction of attributes [6] and incremental algorithm of rule extraction based on concept lattice [7] have been developed, but there are few incremental rough set based algorithms for knowledge acquisition. On the basis of former results, we develop a rough set and rule tree based incremental knowledge acquisition algorithm (RRIA) in this paper. Simulation results show that our algorithm can learn more quickly than classical rough set based knowledge acquisition algorithms, and the performance of knowledge learned by our algorithm can be the same or even better than classical rough set based knowledge acquisition algorithms. Besides, we compare our algorithm with ID4 algorithm. The results show that the rule quality and the recognition rate of our algorithm are both better than those of ID4.

In Section 2, we introduce some basic notions of incremental learning in decision table. In Section 3, we discuss some related incremental learning algorithms. In Section 4, we present the concept of rule tree, we develop a rough set and rule tree based incremental knowledge acquisition algorithm (RRIA), and we analyze its complexity and properties. In Section 5, we test RRIA's efficiency and effectiveness, and we compare it with the classical rough set based algorithm and with ID4 algorithm in some simulation experiments. The last section is the conclusion of this paper.

2. Basic Concepts of Decision Table

For the convenience of description, we recall some basic definitions.

Definition 2.1. A decision table is any tuple $S =< U, R, V, f >$, where U is a finite set of objects, $R = C \cup D$ is a finite set of attributes, C is the condition attribute set, and D is the decision attribute set. With every attribute $a \in R$, a set of its values V_a is associated. A function $f : U \times R \to \bigcup\{V_a : a \in R\}$ defines for each object x and each attribute $a \in R$, the value $f(x, a)$ of attribute a on the object x. We assume $f(x, a) \in V_a$ for all x, a.

Definition 2.2. Let $S =< U, R, V, f$ be a decision table, and let $B \subseteq C$. Then the rule set F generated from S and B consists of all rules of the form

$$\bigwedge\{(a, v) : a \in B \text{ and } v \in V_a \cup \{*\}\} \to d = v_d \tag{1}$$

where $v_d \in V_d$. The symbol $*$ means that the value of the corresponding attribute is irrelevant for the rule, i.e., in conjunction we are not considering the descriptor for this attribute a. The number of rules is equal to the cardinality of F.

For example, in a decision system with 5 condition attributes (a_1,..., a_5), (a_1=1)$\wedge$ (a_3=2)$\wedge$(a_4=3) $\to d$=4 is a rule. In this paper, we write it as (a_1=1)$\wedge$(a_2=*)$\wedge$(a_3=2)$\wedge$(a_4=3)$\wedge$ (a_5=*) $\to d$=4 according to Definition 2.2.

3. Related Incremental Algorithms

In order to compare our algorithm with other related algorithms, we introduce and discuss some decision table based incremental algorithms at first, such as ID4 algorithm [1, 2], ID5R algorithm [2], incremental arithmetic for the smallest reduction of attributes [6], and incremental algorithms for rule extraction based on concept lattice [7], etc. For the convenience of calculating complexities of these algorithms, we suppose that m is the number of attributes in the original decision table, n is the number of objects in the universe U, and b is the maximal number of attribute values. The complexity that we calculate for ID4, ID5R, and the incremental algorithm for rule extraction based on concept lattice is the maximal time complexity to learn one unseen object incrementally.

3.1. ID4 algorithm [1, 2]

Building on the concept of a decision tree, Quinlan developed ID3 algorithm. Decision trees are tree structures of rule sets. They have higher speed of searching and matching than ordinary rule set. Comparing with ordinary rule set, a decision tree has unique matching path. Thus, using decision tree can avoid conflicting of rules and accelerate the speed of searching and matching. However, there are also some drawbacks to decision trees. After recursive partitions, some data is too scarce to express some knowledge or concept. Besides, there are also some problems with the constructed tree, such as overlap, fragment and replication [21]. As shown in Table 1, the results of our experiments also show that the rule set expressed by decision tree has lower recognition rate than most of the rough set based knowledge acquisition algorithms. We randomly choose half parts of some data sets to be training sets and the generated rule sets are used to recognize each total data set respectively. In our experiments, ID3 algorithm is programmed using VC6.0 and other algorithms are from our RIDAS system [15].

So, we should find a method that has the merits of decision tree and avoids its drawbacks.

Table 1. Comparison of ID3 and some rough set based knowledge acquisition algorithms with respect to recognition rate

Data set	ID3	General algorithm[14]	Heuristic algorithm[14]	Discernibility matrix based algorithm[14]	Inductive algorithm[14]
AUTO-MPG	62.6%	64.8%	51.2%	62.8%	47.0%
Humidity	78.9%	83.6%	58.6%	83.8%	84.4%
BEASTCANCER	52.8%	89.4%	65.7%	88.1%	66.1%
GLASS	50%	59.3%	50%	55.6%	51.9%
HAYES-ROTH	50%	87.1%	56.1%	60.6%	60.6%

Schlimmer and Fisher designed an incremental learning algorithm ID4 that solved ID3's problem on incremental learning. By adding, deleting and updating the original rule set, ID4 algorithm effectively uses the original knowledge and reduces the processing time. As shown in Tables 2–4, we test and compare the processing time of ID3 and ID4. We use three data sets of UCI: LIVER (1260 objects), ARRHYTHMIA (452 objects) and ABALONE (4177 objects). We divide one of the data sets into two parts, and then:

Step 1 One of the two parts is processed by ID3 algorithm to create decision tree;

Step 2 The other part is incrementally learned by ID4 algorithm based on the decision tree created in Step 1.

In Tables 2–4, the first row represents the number of the objects to be incrementally learned, the second row represents the time of the first step processing and the data of the third row are the times of the second step processing. From Tables 2–3, the results show that by using ID4 algorithm the processing time of learning is greatly reduced. However, if there is too small difference among the objects in a data set, such as in the data set ABALONE, the decision tree needs to be rebuilt too many times and the time of incremental learning may be greater than that of general learning, as shown in Table 4.

The complexity of ID4 is $O(bm)$ when it needs not to update the tree, and $O(b(m-x)^2n^2)$ when it should rebuild one child tree in its xth layer.

3.2. ID5R [2]

In 1988, to improve ID4 algorithm's learning ability, Utgoff developed ID5 decision algorithm and afterwards the algorithm is updated to ID5R. The decision trees generated by ID5R algorithm are the same as those generated by ID3. Its complexity is $O(wb^{x-1})$ when some node needs to be elevated by w layers, otherwise, $O(bm)$. Here, x is the layer number of the elevated node before being elevated. Each node of the decision tree generated by ID5R must save a great deal of information, so the space complexity of ID5R is higher than ID4. Besides, when the decision tree nodes are elevated too many times, the algorithm time complexity is too high.

3.3. Incremental arithmetic for the smallest reduction of attributes [6]

Reducing for knowledge is an important step in KDD. In theory of rough sets, the principle and methods of the reducing for attributes are specially discussed. Many scholars work out algorithms for reduction,

Table 2. LIVER

	0	100	400	860	1060
ID3	335ms	320ms	280ms	110ms	60ms
ID4	0ms	<1ms	<1ms	50ms	220ms

Table 3. ARRHYTHMIA

	0	52	152	300	400
ID3	13729ms	10045ms	6430ms	2053ms	270ms
ID4	0ms	4456ms	7280ms	6509ms	1793ms

Table 4. ABALONE

	0	177	1177	2088	3000
ID3	23253ms	19629ms	13980ms	6800ms	2464ms
ID4	0ms	9023ms	45315ms	50363ms	31475ms

but the research is almost exclusively done in case of static data. However, databases are dynamic. The incremental arithmetic for the smallest reduction of attributes gets the sets of the new smallest attribute reductions from the old sets after a record is added to the database. That is, based on the known smallest reduction of attributes, this algorithm searches the new smallest reduction of attributes when some attributes are added to the data set. The time complexity of the algorithm is $O(n(Km)^2)$ when an attribute is added to the original data set, where K is the number of attributes in the known smallest reduction of attributes.

3.4. Incremental algorithm of rule extraction based on concept lattice [7]

Many analyzes have shown that concept lattice is an efficient tool for data analysis and rule extraction. It is convenient to model dependence and casuality, and it provides a vivid and concise account of the relations among the variables in the universe of discourse. The algorithm for rule extraction based on concept lattice can extract rules from the built lattice efficiently. The basic idea is to generate all non–redundant rules at each single node by examining the number and content of their parent nodes. In particular, when the number of objects is large and the size of conceptual clustering is big, that is, most of nodes in the lattice have relatively few parent nodes, the algorithm shows much better performance. In the process of updating a concept lattice incrementally, the new method to find the parent node of a new node is given. The complexity of this algorithm is $O(2^{2m}n)$.

This algorithm is based on the incremental learning idea, but it encounters some problems in knowledge acquisition: first, its complexity is too high; second, it takes too much time and space to construct the concept lattice and the Hasse map; third, the rule set cannot be extracted from concept lattice directly.

3.5. Incremental constraint based algorithm [20]

A. Wojna proposed a method based on a special type of monotonic constraints that allows to reduce searching in the space of rules without substantial changes in the classification quality. The author's point is that the space of rules is usually too large for searching for all reduct rules. So, the algorithm limits the set of maintained reduct rules to those satisfying some constraints. The constraints used in this algorithm are confidence and coverage.

However, there are some disadvantages of this method. First, the domain discussed in this method is defined by $U = \{0, 1\}$, while sometimes we need more general one; second, there are so many definition of constraints and user interestingness now, and how to choose among them is mainly based on the object of mining. So, we think it is not suitable to decide the constraints before the application of algorithms.

3.6. Comparison of incremental algorithms

The advantages and disadvantages of these algorithms are listed and compared with each other in Table 5.

Table 5. Comparison of Incremental Algorithms

Algorithm	Advantages	Disadvantages
ID4	It uses original rule set and decision table selectively. It uses tree structure to represent rule set.It's searching and matching speed is high.	There are overlap, fragment and replication in its resulting tree. The rule set extracted by the algorithm has low recognition rate.
ID5R	Its learning ability is high and the decision trees constructed by this algorithm are the same as that constructed by ID3.	The time complexities of generating and elevating decision tree are high. The space complexity of constructing a decision tree is also high. The rule set extracted by the algorithm has low recognition rate.
Incremental arithmetic for the smallest reduction of attributes	It can incrementally acquire the smallest reduction of attributes.	It cannot incrementally acquire knowledge.
Incremental algorithm of rule extraction based on concept lattice	It is easy to establish the dependent and causal relation model. And concept lattice can clearly describe the extensive and special relation.	Its complexity is high. It is time and space consuming to construct concept lattice and Hasse map. Rule set cannot be extracted from the concept lattice directly.
Incremental constraint based algorithm	It uses original rule set and decision table selectively. And it can incrementally acquire knowledge without substantial changes in the classification quality	Its domain discussed is binary and it is unsuitable to decide the constraints before the application of the algorithm.

4. Rough Set and Rule Tree Based Incremental Knowledge Acquisition Algorithm (RRIA)

Motivated by the above discussion of incremental learning algorithms and our understanding of incremental learning algorithms, we develop a rough set and rule tree based incremental knowledge acquisition algorithm in this section. In this algorithm, we use the original rule set and decision table selectively in the incremental learning process. We use tree structure to express rules to speed up the searching and matching process. Besides, we use rough set based algorithm to generate the original rule set to avoid some disadvantages of decision tree, such as overlap, fragment and replication. In this part, we introduce the concepts of rule tree first and then present our incremental learning algorithm. Then, we analyze the algorithm complexity and performance. For the convenience of description, we suppose that *OTD* represents the training original data set in our algorithm, *ITD* represents the incremental training data set, *ORS* represents the original rule set, and *ORT* represents the original rule tree.

4.1. Rule tree

4.1.1. Basic concept of rule tree

Definition 4.1. A rule tree is defined as follows:
(1) A rule tree is composed of one root node, some leaf nodes and some middle nodes.
(2) The root node represents the whole rule set.
(3) Each path from the root node to a leaf node represents a rule.
(4) Each middle node represents an attribute testing. Each possible value of an attribute in a rule set is represented by a branch. Each branch generates a new child node. If an attribute is reduced in some rules, then a special branch is needed to represent it and the value of the attribute in this rule is supposed as "*", which is different from any possible value of the attribute (Ref. Definition 2.2).

4.1.2. Algorithms for building rule tree

The first step of RRIA is to create the rule tree according to the rule set *ORS* generated by the classical rough set knowledge acquisition algorithm. The algorithm is illustrated as follows. The algorithm first arranges the condition attributes in ascending order of the number of their values to speed up the searching and matching process. Then it is repeated until all rules in ORS are processed by a child algorithm, AddRule, which is described in Algorithm 2. At last, a rule tree *ORT* that generalizes all the rules in *ORS*is created.

Algorithm 1: CreateRuleTree(*ORS*)
Input: *ORS*;
Output: *ORT*.
Step 1 Arrange the condition attributes in ascending order of the number of their values in *ORS*. Then, each attribute is the discernibility attribute of each layer of the tree from top to bottom;
Step 2 For each rule R of *ORS* {AddRule(*R*)}.

We suppose that the attribute with fewer number of attribute values is the discernibility attribute in a higher layer of a rule tree. By this supposition, the number of nodes in each layer should be the least

possible, and the number of searching nodes should also be the least possible, which should speed up the searching and matching process.

The algorithm AddRule starts at a rule tree and the rule R to be incrementally learned and added to the tree. This process is repeated on each layer of the rule tree. In each layer, we search whether there is a branch of the current node CN representing the current discernibility attribute value of rule R. If so, the node generated by the branch is the new current node. Otherwise, create a branch to represent the current discernibility attribute value of rule R to be the new current node.

Algorithm 2: AddRule(R)
Input: a rule tree and a rule R;
Output: a new rule tree updated by rule R.
Step 1 CN=root node of the rule tree;
Step 2 For i=1 to m /*m is the number of layers in the rule tree*/
{If there is a branch of CN representing the i–th discernibility attribute value of rule R, then
CN=node I;/*node I is the node generated by the branch*/
else {create a branch of CN to represent the i–th discernibility attribute value;
CN=node J; /*node J is the node generated by the branch*/}}

4.1.3. Rule tree's properties

Property 1. A rule tree is the tree form of a rule set. It is equivalent to the rule set.
Property 2. The time complexity for searching a rule tree is less than that of a rule set.
Property 3. The searching path of a rule tree is not unique. There may be one or two branches in each layer that can be matched to an object.
Property 4. There may be more than one rule matched to an unseen object in a rule tree. Some method is needed to deal with this case.

According to properties of a rule tree, we know a rule tree is a efficient and effective form to represent a rule set. So, in our algorithm, we use rule tree to speed up our searching and some other processes.

4.2. Rough set and rule tree based incremental knowledge acquisition algorithm (RRIA)

The feature of a useful incremental knowledge acquisition algorithm is that it can selectively use the original rule set and the original data set. Besides, it should process the new objects with different methods according to its relationship with the original rule set and decision table. In our RRIA algorithm, these features are realized. We use the tree structure to express rules to speed up the searching and matching process and we divide the new objects into three kinds. The objects of the first kind are consistent with the original rule tree, so there is no need to modify anything. The objects of the second kind are inconsistent objects, which have no consistent rule in the rule set. The objects of the third kind are new objects, which have neither consistent nor inconsistent rule in the rule set. The algorithm is illustrated as follows.

Algorithm 3: Rough set and rule tree based incremental knowledge acquisition algorithm
Input: OTD, ITD=$object_1$, $object_2$, ... , $object_t$;
Output: a rule tree.
Step 1 Use rough set based algorithm to generate ORS from OTD;

Step 2 *ORT*=CreateRuleTree(*ORS*);
Step 3 For i=1 to t

(1) *CN*=root node of *ORT*;

(2) Create array *Path*[]; /*Array *Path* records the current searching path in MatchRule() and the initial value of *Path* is NULL*/

(3) *MatchRules*=MatchRule(*CN*, $object_i$, *Path*);

(4) *R*=SelectRule (*MatchRules*);

(5) If *R* exists and the decision value of *R* is different from $object_i$'s, then UpdateTree($object_i$,*R*); /**R* is conflicting with $object_i$*/

(6) If *R* does not exist then AddRule($object_i$).

Algorithm AddRule is described in Section 4.1 and algorithms MatchRule, SelectRule and UpdateTree will be further illustrated in the following section.

4.3. Child algorithms of RRIA

In the incremental learning process, we need to match the object to be learned incrementally with rules of *ORT* at first. The resulting algorithm is like Algorithm 4. The algorithm uses as input the current matched node and the object to be incrementally learned. MatchRule is a recursive algorithm and the primary idea of the algorithm is the same as in Algorithm 2 (AddRule), that is, search each layer of the rule tree whether there is a branch of the current node representing the current discernibility attribute value of the object to be learned. Besides, an array is needed to record the current searching path and in the algorithm the array is defined as *Path*. After the algorithm is processed, all the matched rules are saved in the dual array *MatchRules* if they exist. There may be more than one matched rule in the result.

Algorithm 4: MatchRule(*CN*, *Obj*, *Path*)
Input: the current matched node *CN*, the current object *Obj* to be learned incrementally, the current searching path *Path* and *ORT*;
Output: MatchRules. /*which records the matched rules of *Obj* in *ORT*.*/
Step 1 *matchedrulenum*=0;
Step 2 For i=1 to the number of child nodes of *CN*

{If the branch generating child node i represents the current discernibility attribute value of *Obj*, or represents a possible reduced attribute, then

{If node i is a leaf node, then

{*MatchRules*[*matchedrulenum*++]=*Path*; *Path*=NULL;}

else { *CN*=node i;

Path[layer_of(node$i$)]=value(node i);

/*The result of value(node i) is the current discernibility attribute value that the branch generating node i represents and the result of layer_of(node i) is the layer of *ORT* that node i is in. "*" represents a possible reduced attribute value (Ref. Definition 2.2*/

MatchRule(node i, *Obj*, *Path*);}}

else *Path*=NULL;} /* If neither of the two kinds of child nodes exist */

This algorithm is recursive. The maximal number of nodes in *ORT* to be checked is *mb*, and the complexity of this algorithm is O(*mb*).

There may be more than one rule in MatchRules. There are several methods to select the best rule from MatchRules, such as High confidence first principle [14], Majority principle [14], and Minority principle [14, 18]. Here we design a new principle, that is Majority Principle of Incremental Learning Data, which is more suitable for our incremental learning algorithm and described as follows.

Algorithm 5: SelectRule(*MatchRules*)
Input: the output of algorithm MatchRule, that is, *MatchRules*;
Output: the final matched rule.

During the incremental learning process, we consider that the incremental training data is more important than the original training data. The importance degree of a rule will be increased by 1 each time it is matched to an unseen object to be learned in the incremental learning process. We choose the rule with the highest importance degree to be the final matched rule.

If we cannot find the rule absolutely matched to the object to be learned, but we can find a rule R matching its condition attribute values, that is, the object to be learned is conflicting with the rule R, then we should update the rule tree. The algorithm is as follows. Suppose $\{C_1, C_2, ..., C_p\}$ is the result of arranging the condition attributes reduced in R in ascending number of their values. The primary idea of the algorithm is to find the condition attributes that have been reduced in R, which can eliminate the conflict between rule R and the learned objects. First, we choose the objects that are consistent with R from *OTD*, and the objects that have been incrementally learned in *ITD*. Then, we look for the least number of attributes that can eliminate the conflict. At last, we delete rule R from *ORS*, create some new rules that include the condition attributes in R and the attributes found before and add the rules into *ORS*.

Algorithm 6: UpdateTree(*Obj*, *R*)
Input: the current object *Obj* to be learned incrementally, rule *R*, *ORT*, *OTD* and *ITD*;
Output: a new rule tree.
Step 1 Check all objects of *OTD* and the objects that have been incrementally learned in *ITD* and suppose $object_1$, $object_2$, ..., $object_q$ are all objects matched to rule R;
Step 2 For i=1 to q
{ $dis\_num[i]$=0;
For j=1 to p {$mark[i][j]$=0;}}
Step 3 For i=1 to q
For j=1 to p
If *Obj* and $object_i$ have different values on attribute C_j, then
{ $mark[i][j]$=1;
$dis\_num[i]$++;}
Step 4 For i=1 to p
{$delattr[i]=C_i$;}
Step 5 For i=1 to q
{If $dis\_num[i]$==0, then
{$delattr$=NULL; go to Step 7;}
else If $dis\_num[i]$==1, then

{*j*=1;
While(*mark*[*i*][*j*]≠1) *j*=*j*+1;
delattr=*delattr*-{C_j};}}

Step 6 For *i*=1 to *q*
{ *discerned*=FALSE;
If *dis_num*[*i*]>1then
{*j*=1;
While (*discerned*==FALSE) and (*j*<*p*+1)
{If (*mark*[*i*][*j*]==1 and $C_j \neq delattr[k]$ (*k*=1,..., *p*)), then
{*discerned*=TRUE;}
j=*j*+1;}
If *discerned*==FALSE, then
{ *j*=1;
While(*mark*[*i*][*j*]≠1) *j*=*j*+1;
delattr=*delattr*-{C_j};}}}

Step 7 Delete rule *R* from the rule tree and generate rules for $object_1$, $object_2$,..., $object_q$, *Obj* separately, their condition attributes include all condition attributes but those in *delattr*. There will be *q*+1 rules, and we suppose they are $rule_1$, $rule_2$,...,$rule_{q+1}$;

Step 8 For *i*=1 to *q*+1 AddRule($rule_i$).

The algorithm complexity is O(max(qp^2,*mn*)), where *q* is the number of matched objects and *p* is the number of attributes reduced from *R*.

If we can neither find the absolutely matched rule nor a conflict rule, it means that the rule tree does not include any information on the object to be learned, and we should create a new one, so we add the object as a new rule *R* to the rule tree, that is AddRule($object_i$). The algorithm complexity is O(*mb*). In addition to the method we use, there may be some other method to create a new rule that represents the object to be learned, such as choosing one of the condition attributes of the object to be the only condition attribute in the new rule. But after some experiments, it shows that our method is better.

4.4. Analysis of RRIA

While learning an unseen object incrementally, the total complexity of algorithm MatchRule and algorithm SelectRule is O(*mb*), the complexity of algorithm UpdateTree is O(max(qp^2,*mn*)), where *q* is the number of matched objects and *p* is the number of attributes reduced from the final matched rule. The complexity of algorithm AddRule is O(*mb*). Obviously, $b < n$, $q << n$ and $p << m$, so RRIA algorithm's complexity is:

$O(max(qp)^2, mn))$, if the rule tree need to be rebuilt;

$O(mb)$, otherwise.

RRIA is based on the classical rough set rule acquisition idea. *ORS* is generated with classical rough set based algorithm. The algorithm UpdateTree and algorithm AddRule are also based on the rough set classification idea. The main idea of algorithm UpdateTree is generating the smallest rule for the unseen object to be learned and the rule can be discerned with other rules. This idea is the same as most of rough set based knowledge acquisition algorithms.

RRIA can be used repeatedly when unseen objects are generated continuously. For example, there are 100 objects to be learned incrementally. We can either learn them all at a time by RRIA, or divide

them into 10 parts, learn the first part first and then learn the other 9 parts by our algorithm sequentially. There will be no difference between the results of these two methods

5. Results of Experiments

In order to test the validity and ability of RRIA algorithm, we implement all algorithms using VC6.0. We compare RRIA with classical rough set based knowledge acquisition algorithm and ID4 algorithm. Some classical data sets from UCI and other data sets used by many researchers are used in our experiment.

Experiment 1: Comparison between RRIA and classical rough set knowledge acquisition algorithm.
Step 1 Choose $\beta\%$ objects from the total data set to be *OTD*. Use classical rough set based knowledge acquisition algorithm to generate *ORS*;
Step 2 Build *ORT* from *ORS* with Algorithm 1;
Step 3 Choose $\alpha\%$ objects from the total data set to be *ITD*. And incrementally learn *ITD* with RRIA algorithm;
Step 4 Test the recognition rate of the rule set generated in step 3 on the total data set;
Step 5 Generate a rule set from *OTD+ITD* with the classical rough set based knowledge acquisition algorithm used in Step 1. Test the recognition rate of the generated rule set on the total data set.(*OTD+ITD* represents the union of *OTD* and *ITD*).

The experimental results are shown in Table 6. We suppose A_1 is the discernibility matrix based knowledge acquisition algorithm [8] and A_2 is the general knowledge acquisition algorithm [8] used in Step 1. RSet means the classical rough set knowledge acquisition algorithm. For the convenience of comparison, we suppose that n is the total number of objects in a data set, $\alpha\%$ of each data set is selected as its *ITD*, $\beta\%$ of each data set is selected as its *OTD*, t is the running time of RRIA in the incremental learning process and A_1(or A_2) separately, c is the correct recognition rate of the generated rules, l is the average length of generated rules, and a is the number of the generated rules. From Experiment 1, we can find that RRIA algorithm has higher speed than the classical rough set based knowledge acquisition algorithm, and there is no much difference between the recognition rate of RRIA and the classical rough set knowledge acquisition algorithm. In some cases, the recognition rate of RRIA is better than that of the classical rough set knowledge acquisition algorithm.

Experiment 2: Comparison between RRIA and ID4 algorithm.
Step 1 Choose $\beta\%$ objects from the total data set to be *OTD* and use classical rough set knowledge acquisition algorithm to generate *ORS*;
Step 2 Build *ORT* from *ORS* with Algorithm 1;
Step 3 Choose $\alpha\%$ objects from the total data set to be *ITD*;
Step 4 Incrementally learn *ITD* with RRIA algorithm and suppose the generated rule set is IT_1;
Step 5 Use ID3 algorithm to build a decision tree from *OTD*;
Step 6 Incrementally learn *ITD* with ID4 algorithm and suppose the generated rule sets is IT_2;
Step 7 Test the recognition rates of IT_1 and IT_2 on the total data set.

The experiment results are shown in Table 7. The meaning of the symbols in Table 7 is the same as that in Table 6. The results show that the recognition rate of our algorithm is better than in case of ID4.

Table 6. Comparison between RRIA and classical rough set knowledge acquisition algorithm

Data set	Alg.	β	α	RRIA			RSet		
				t	c	$l(a)$	t	c	$l(a)$
AUTO MPG	A_1	40	10	< 1	64.8	2.07(165)	170	62.8	1.01(156)
AUTOMPG	A_1	33	17	<1	65.1	1.94(159)	170	62.8	1.03(156)
AUTOMPG	A_1	25	25	10	61.6	3.78(166)	170	62.8	1.03(156)
BEASTCANCER	A_1	40	10	10	87.7	1.33(192)	2013	88.1	1.04(233)
BEASTCANCER	A_1	33	17	10	88.7	2.21(179)	2013	88.1	1.04(233)
BEASTCANCER	A_1	25	25	30	67.0	6.02(276)	2013	88.1	1.04(233)
HUMIDITY	A_2	40	10	50	84.2	3.11(170)	471	83.6	1.79(161)
HUMIDITY	A_1	33	17	60	84.1	3.88(174)	471	83.6	1.79(161)
HUMIDITY	A_2	25	25	111	83.7	4.60(195)	471	83.6	1.79(161)
ABALONE	A_2	40	10	781	54.9	3.76(2121)	2063	52.7	3.35(1937)
ABALONE	A_2	33	17	1222	55.2	3.95(2253)	2063	52.7	3.35(1937)
ABALONE	A_2	25	25	1812	55.0	4.24(2397)	2063	52.7	3.35(1937)
BALANCE	A_1	40	10	20	22.6	3.11(345)	41	18.4	3.92(312)
BALANCE	A_1	33	17	20	18.4	3.33(312)	41	18.4	3.92(312)
BALANCE	A_1	25	25	30	19.4	3.94(316)	41	18.4	3.92(312)

Table 7. Comparison of RRIA and ID4 algorithm

Data set	Alg.	β	α	RRIA			ID4		
				t	c	$l(a)$	t	c	$l(a)$
AUTOMPG	A_1	40	10	< 1	64.8	2.07(165)	< 1	56.0	2.71(164)
AUTO MPG	A_1	33	17	< 1	65.1	1.94(159)	10	60.6	2.51(162)
AUTO MPG	A_1	25	25	10	61.6	3.78(166)	10	51.0	4.57(166)
BEAST CANCER	A_1	40	10	10	87.7	1.33(192)	10	52.5	2.77(322)
BEAST CANCER	A_1	33	17	10	88.7	2.21(179)	20	52.1	3.88(335)
BEAST CANCER	A_1	25	25	30	67.0	6.02(276)	30	50.1	6.74(342)
HUMIDITY	A_2	40	10	50	84.2	3.11(170)	10	69.5	4.27(172)
HUMIDITY	A_1	33	17	60	84.1	3.88(174)	10	65.6	4.93(177)
HUMIDITY	A_2	25	25	111	83.7	4.60(195)	< 1	65.1	7.21(188)
ABALONE	A_2	40	10	781	54.9	3.76(2121)	151	41.7	3.90(1836)
ABALONE	A_2	33	17	1222	55.2	3.95(2253)	221	37.4	4.50(1824)
ABALONE	A_2	25	25	1772	55.0	4.24(2397)	250	31.3	5.27(1832)
BALANCE	A_1	40	10	20	22.6	3.11(345)	< 1	18.4	3.85(156)
BALANCE	A_1	33	17	20	18.4	3.33(312)	< 1	18.4	3.90(184)
BALANCE	A_1	25	25	30	19.4	3.94(316)	10	18.1	3.93(212)

Besides, the average length of the rules generated by RRIA is smaller than that of ID4, and the number of rules generated by RRIA is also less than that of ID4.

6. Conclusion

Incremental learning is an important topic in AI. Many rough set based algorithms about the smallest or smaller attribute reduction and rule set induction from static data have been developed in the past years. Unfortunately, databases in real life are always dynamic. So, many researchers suggest that knowledge acquisition in databases should be incremental. Incremental learning algorithms can be used not only in the fields that classical algorithms can be used in with higher speed but also in some other special fields. Incremental arithmetic for the smallest reduction of attributes and incremental algorithms of rule extraction based on concept lattice have already been studied, but there are few incremental rough set based algorithms about knowledge acquisition. In this paper, ID4 algorithm, ID5R algorithm, incremental arithmetic for the smallest reduction of attributes, and incremental algorithms for rule extraction based on concept lattice are discussed.

After studying these incremental learning algorithms, we find that not only the advantages of decision trees can be preserved but also the disadvantages of ID4, ID5R and some other incremental learning algorithms can be avoided if tree structure and rough set knowledge acquisition idea are combined together. Based on this idea, we develop rough set and rule tree based incremental knowledge acquisition algorithm. Through analysis and simulation tests, we have found that the RRIA algorithm has higher speed than classical rough set based knowledge acquisition algorithm, and the performance of knowledge learned by our algorithm can be the same as or even better than classical rough set based knowledge acquisition algorithms. We also compare RRIA with ID4 algorithm. The results show that some capabilities of RRIA algorithm, such as rule length, number of rules, recognition rate, etc, are also better than ID4.

Acknowledgements

This paper is partially supported by National Natural Science Foundation of P.R.China (No.60373111), National Climb Program of P.R.China, Foundation for University Key Teacher by the State Education Ministry of P.R.China (No.GG-520-10617-1001), Scientific Research Foundation for the Returned Overseas Chinese Scholars by the State Education Ministry of P.R.China, and Application Science Foundation of Chongqing, and Science and Technology Research Program of the Municipal Education Committee of Chongqing of P.R.China (No.020505).

References

[1] Schlimmer, J. C., Fisher, D. A.: Case Study of Incremental Concept Induction. In Proceedings of the Fifth National Conf. on Artificial Intelligence, Los Altos, 1986

[2] Utgoff, P. E.: Incremental induction of decision trees. Machine Learning **4** (1989) 161–186

[3] Wang, G. Y., Shi, H. B., Deng, W.: A parallel neural network architecture based on nara model and sieving method. Chinese Journal of Computers **19**(9) (1996)

[4] Wang, G. Y., Nie, N.: PMSN: A Parallel multi–sieving neural network architecture. Journal of Computer Research and Development **36** (Suppl.) (1999) 21–25

[5] Wang, G. Y., Shi, H. B.: Parallel neural network architectures and their applications. In Proceedings of Int. Conf. on Neural Networks, Perth, Australia, 1995, 1234–1239

[6] Liu, Z. T.: An Incremental arithmetic for the smallest reduction of attributes. Chinese Journal of Electronics **27**(11) (1999) 96–98

[7] Wang, Z. H., Liu, Z. T: General and incremental algorithms of rule extraction based on concept lattice. Chinese Journal of Computers **22**(1) (1999) 66–70

[8] Hu, X. H., Cercone, N.: Learning in relational databases: A rough set approach. Computational Intelligence **11**(2) (1995) 323–338

[9] Jelonek, Krawiec, K., Słowiński, R.: Rough set reduction of attributes and their domains for neural networks. Computational Intelligence **11**(2) (1995) 339–347

[10] Miao, D. Q.: The Researching of Rough Set Theory and Its Application in Machine Learning [PH.D thesis]. Beijing Institute of Automation, Chinese Academy of Sciences, 1997

[11] Fayyad, U. M., Piatetsky-Shapiro, L., Smyth, P., Uthurusamy, R.: Advances in Knowledge Discovery and Data Mining. Menlo Park, CA: AAAI Press/The MIT Press, 1996

[12] Cercone, V., Tsuchiya, M.: Lousy Editor's Introduction. IEEE Transations on Knowledge and Data Engineering **5**(6) (1993) 901-902

[13] Piatetsky-Shapiro, L., Frawley, W. J.: Knowledge Discovery in Databases. Menlo Park, CA: AAAI Press/The MIT Press, 1991

[14] Wang, G. Y.: Rough Set Theory and Knowledge Acquisition. Xi'an Jiaotong University Press, Xi'an, 2001

[15] Wang, G. Y., Zheng, Z., Zhang, Y.: RIDAS-A Rough Set Based Intelligent Data Analysis System. In Proceedings of the First Int. Conf. on Machine Learning and Cybernetics, Beijing, 1991, 646–649

[16] Su, J., Gao, J.: Metainformation based rough set incrementally rule extraction algorithm. Pattern Recognition and Artificial Intelligence **14**(4) (2001) 428–433

[17] Kou, Y. J., Wang, C. H., Huang, H. K.: An incremental algorithm for maintaining constrained association rules. Journal of Computer Research and Development **38**(8) (2001) 947–951

[18] Wang, G. Y., Liu, F.: The inconsistency in rough set based rule generation. In Ziarko, W., Yao, Y.Y. eds. Second International Conference on Rough Sets and Current Trends in Computing,LNAI 2005, Springer-Verlag, Berlin, 2001, 370–377

[19] Han, J. W., Kamber, M.: Data Mining Concepts and Techniques. Morgan Kaufmann, San Francisco, 2001.

[20] Wojna, A.: Constraint based incremental learning of classification rules. In Ziarko, W., Yao, Y.Y. eds. Second International Conference on Rough Sets and Current Trends in Computing, LNAI 2005, Springer-Verlag, Berlin, 2001, 428–435

[21] Lu, R. Q.: Artificial Intelligence. Scientific Press, Beijing, 2002.