A NOVEL DYNAMIC INCREMENTAL RULES EXTRACTION ALGORITHM BASED ON ROUGH SET THEORY

SEN GUO, ZHI-YAN WANG, ZHI-CHENG WU, HE-PING YAN
School of Computer Science and Engineering, South China University of Technology, GuangZhou, GuangDong, China, 510640
E-MAIL: ybbsss1210@126.com

Abstracts:

The incremental rules extraction is a focus problem of KDD. In this paper, a novel incremental rules extraction algorithm which is called "RDBRST" (Rule Derivation Based On Rough set and Search Tree) is proposed. It is one kind of width first heuristic search algorithms. The incremental rules are extracted and the existing rule set is updated based on this algorithm. We present an example to illustrate characteristics of this new incremental algorithm.

Keywords:

Rough set; search tree; width first heuristic search algorithm; rule derivation

1. Introduction

The rough set theory ([1]) has been successful applied in the area of data reduction, decision support, classification, pattern recognition and so on. Knowledge discovery is one of the most important application of rough set theory because the growth of size and number of databases far exceeds the ability of human to analyze. The theory of rough set is a mathematical tool to extracting knowledge from databases. The existing algorithms of rough set have the ability to generate a set of classification rules efficiently, but they cannot generate rules incrementally when new objects are given. In practical application, the recorders of database are often increased dynamically. So, if new object arrival, we have to compute the whole database again. This procession is due to consume huge compute time and memory space. It is necessary to study the efficient incremental rules extracting algorithms based on RS. Some researchers have proposed their algorithms ([2], [3], [4], [5], [6]). For instance, the ∂ -decision matrix algorithm proposed by AN Liping, WU Yuhua[4], the algorithm based on extended_ discernibility matrix proposed by Yu Dongjun, Wang Shitong etc [3]. The common problem of existing algorithms is that they should store and process one $n\times n$ matrix in memory (the number n is the size of database). Their space complexity is so huge that those algorithms are limitation when the size of database is large. In this paper, we bring forwards a novel algorithm to extract rules without processing such a matrix, so this algorithm has better space complexity compared to original ones. In addition, in the procession of rules generation, this algorithm can reduce search space and further improve efficiency based on taken the known rule set as heuristic information.

This paper is organized as follows: Theoretical aspects of rough set data analysis and the concepts of search tree which are relevant to the work are introduced in section 2. We present a novel algorithm in section 3, which is called "RDBRST" (Rule Derivation Based On Rough set and Search Tree). In section 4, we discuss the known rule set updated algorithm based on "RDBRST" algorithm. We draw a conclusion in section 5.

2. Basic Concepts

2.1. Basic Concepts of Rough set

Knowledge representation in rough sets is done via information systems which is denoted as $S=(U,A,V,f)$, where U is finite set of objects, A is finite set of attributes, $A=C\bigcup D$, C is the condition attributes and D is the decision attributes. V is the domain of A and f is an information function, $f:U\times A\rightarrow V$.

Assume a subset of attributes, $R\subseteq A$, the indiscernibility relation is written as ind(R), $ind(R)=\{(x,y)\in U\times U \mid \forall a\in R, f(x,a)=f(y,a)\}$, and $U/ind(R)$ is used to denote the partition of U.

A rough set approximates traditional sets using a pair of sets named the lower and upper approximation of set.

0-7803-9091-1/05/$20.00 ©2005 IEEE

The lower and upper approximation are defined by the equations (2.1) and (2.2) respectively below:

$$\underline{RY}=\bigcup\left\{X \mid X\in {U}/{ind(R)}, X\subseteq Y\right\} \quad (2.1)$$

$$\overline{RY}=\bigcup\left\{X \mid X\in {U}/{ind(R)}, X\bigcap Y\neq\phi\right\} \quad (2.2)$$

The equivalence class including x with respect to the equivalence relation is denoted by $[x]_R$ $R\subseteq C$. The characteristic of $[x]_R$ is described by $Des([x]_R)$, which implies $\{R(x)=v, v\in V\}$.

If the relation: $[x]_R\subseteq[x]_D$ is satisfied, then a rule is obtained, which denote as $rx: Des([x]_R)\Rightarrow Des([x]_D)$.

2.2. The concept of search tree

Search tree which is defined as a sub_ tree of state space tree is constructed by nodes and points generated in the procession of searching. It is a directed tree, and its root node hasn't father node, other nodes have and only have one father node. The nodes which have no child node are called "left node", the rest nodes are called "internal" nodes. In this paper, our search strategy is designed based on heuristic width first search algorithm. The basic principle of this algorithm is that: The nodes is expanded layer by layer according to their distance with root node, The search tree is rearranged and pruned by heuristic information till solution is found or space of solution is exhaustive.

3. The "RDBRST" algorithm

The algorithm which is called "RDBRST" (Rule Derivation Based On Rough set and Search Tree) is on grounds of rough set theory and the concept of search tree. The search strategy of algorithm is width first search algorithm which takes known rules set as heuristic information. There are two important concept of this algorithm: relativity and relevant object set (ROS).

3.1. The concept of relativity and related object set (ROS)

Definition 1 (relativity): Given an information system $S=(U,A,V,f)$, U is finite set of objects, $U=\{x_1,x_2,...,x_n\}$, $A=C\bigcup D$, C is the condition attributes and D is the decision attributes, x is one object of U, and it satisfies equation below:

$$[x]_{\hat{C}}\subseteq[x]_D, \hat{C}\subseteq C \quad (3.1)$$

If a new object y is added to U, in the new set of objects U\`, $U'=U\bigcup\{y\}$, the equation (3.1) is also satisfied, then object x is called "irrelevant" to y, otherwise, they are called "relevant" to y.

Definition 2(relevant object set): the relevant object set is defined as the set which consists of the object y and all its relevant object.

Theorem 3.1: Given an information system $S=(U,A,V,f)$, the new object is y, In the new set U\`, $U'=U\bigcup\{y\}$, if object x, satisfies one of the two equations (3.2) and (3.3) below, then x is called "irrelevant" to y, otherwise, it called "relevant" to y.

(1) $\forall c\in C, c(x)\neq c(y)$ (3.2)

(2) $\forall d\in D, d(x)=d(y)$ (3.3)

Proof: Let S be an information system, $S=(U,A,V,f)$ and r_x is the rules extracted from U with respect to object x, $rx: Des([x]_{\hat{C}})\Rightarrow Des([x]_D)$, $\hat{C}\subseteq C$. According to the basic concept of RS, it can conclude :

$$[x]_{\hat{C}}\subseteq[x]_D \quad (3.4)$$

We add a new object y to the set and form a new information system S\`, $S'=(U',A,V,f)$, $U'=U\bigcup\{y\}$. According to the $[x]_{\hat{C}}$ and $[x]_D$ in the equation (3.4), we define two notations in the new set U', $[x]'_{\hat{C}}$ and $[x]'_D$.

(1) Since $\forall c\in C, c(x)\neq c(y)$, according to the definition of equivalence class, it can derivate $y\notin[x]'_{\hat{C}}$. This is to say, after adding new object y to object set U, the equivalence class which includes objects x isn't affected by the modification, $[x]'_{\hat{C}}=[x]_{\hat{C}}$.

There are two cases in total:

Case 1: $d(x)=d(y)$.

In this case, $[x]'_D=[x]_D\bigcup\{y\}$, and $[x]_D\subset[x]'_D$.

Case 2: $d(x)\neq d(y)$.

In this case, $[x]_D=[x]'_D$.

$\because [x]_{\hat{C}}\subseteq[x]_D \therefore [x]'_{\hat{C}}\subseteq[x]_D$

$\therefore [x]'_{\hat{C}} \subseteq [x]'_{D}$.

According to definition 1, the object *x* is "irrelevant" to object y.

(2) Since $\forall d \in D, d(x) = d(y)$, then $[x]'_{D} = [x]_{D} \bigcup \{y\}$.

There are also two cases: $\hat{C}(x) = \hat{C}(y)$ and $\hat{C}(x) \neq \hat{C}(y)$. According to the definition of equivalence class, if $\hat{C}(x) = \hat{C}(y)$, then $[x]'_{\hat{C}} = [x]_{\hat{C}} \bigcup \{y\}$, otherwise, if $\hat{C}(x) \neq \hat{C}(y)$, then $[x]'_{\hat{C}} = [x]_{\hat{C}}$, so it can concluded that $[x]'_{\hat{C}} \subseteq ([x]_{\hat{C}} \bigcup \{y\})$.

$\because [x]_{\hat{C}} \subseteq [x]_{D}$ $\therefore [x]'_{\hat{C}} \subseteq ([x]_{\hat{C}} \bigcup \{y\}) \subseteq ([x]_{D} \bigcup \{y\})$

$\because [x]'_{D} = [x]_{D} \bigcup \{y\}$ $\therefore [x]'_{\hat{C}} \subseteq [x]'_{D}$

According to definition 1, the object *x* is "irrelevant" to object *y*.

3.2. The "RDBRST" algorithm

The algorithm of rule derivation based on rough set and search tree is in the following way:

Algorithm 1: RDBRST algorithm

Input:

(1) $S = (U, A, V, f)$, $A = C \bigcup D$, $C = \{c_1, c_2, ...c_m\}$, $D = \{d\}$

(2) The known rule set *R*.

(3) The new object: $u = \{u_1, u_2, ..., u_m, d_u\}$

(4) $S' = S \bigcup \{u\}$.

(5) The object set *P*, which is the relevant object set according to the new object *u*.

(6) $open = \{Des([u]_{C_1}), Des([u]_{C_2}), ...Des([u]_{C_m})\}$

Output: *Q*, which is the new added rule set generated from *S'* according to new object *u* added.

Begin:

(1) Take rules whose consequence is equal to $Des([u]_D)$ from known rule set R into a new object set *Q*.

(2) If the set open is empty, go to end.

(3) Take out the first node of open, and put it into the extend node set closed.

Assumed the node is $Des([u]_{\hat{C}})$, its successor nodes are $\bigcup \{Des([u])_{C'}\}$ where $C' = \{C'' \in C, C'' \notin \hat{C} : C'' \bigcup \hat{C}\}$, put all its successor nodes into closed.

(4) Do data screening in closed

Assume $Des([u])_{C'}$ is one node of closed, if it satisfies one of the three conditions below, filter it out.

A: In *Q*, there exits one rule $Des([u']_{\bar{C}}) \Rightarrow Des([u']_D)$ which satisfies $Des([u']_{\bar{C}}) \subseteq Des([u]_{C'})$.

B: In open and closed, there are identical nodes already.

C: In the relevant object set *P*, the relation $Des([u]_{C'}) \subseteq Des([u]_D)$ is satisfied.

According to the nodes which satisfy condition C,

$$R = R \bigcup \{Des([u]_{C'})\} \Rightarrow Des([u]_D)$$

$$Q = Q \bigcup \{Des([u]_{C'}) \Rightarrow Des([u]_D)\}$$

(5) Put all the rest nodes of closed into rear part of open.

(6) Go to (2).

End.

We illustrate the above algorithm by the example below. Table 1 is an information system, the condition attributes set is $\{C_1, C_2, C_3, C_4\}$, and decision attribute set is $\{D\}$. Table 2 is the known rules set derived from table 1.

Let new object $u = (1,2,3,1,3)$, then the relevant object set according to the new object u is shown in table 3. According to the algorithm above, the initial set is $open = \{C_1 = 1, C_2 = 2, C_3 = 3, C_4 = 1\}$, and the output result is $(C_1 = 1 \wedge C_2 = 2) \Rightarrow (D = 3)$. The search graph of solution space is shown in Figure 1.

Table 1. Information system

C1	C2	C3	C4	D
2	1	2	3	1
2	2	1	2	1
1	3	1	2	2
3	3	1	3	2
1	1	3	1	3

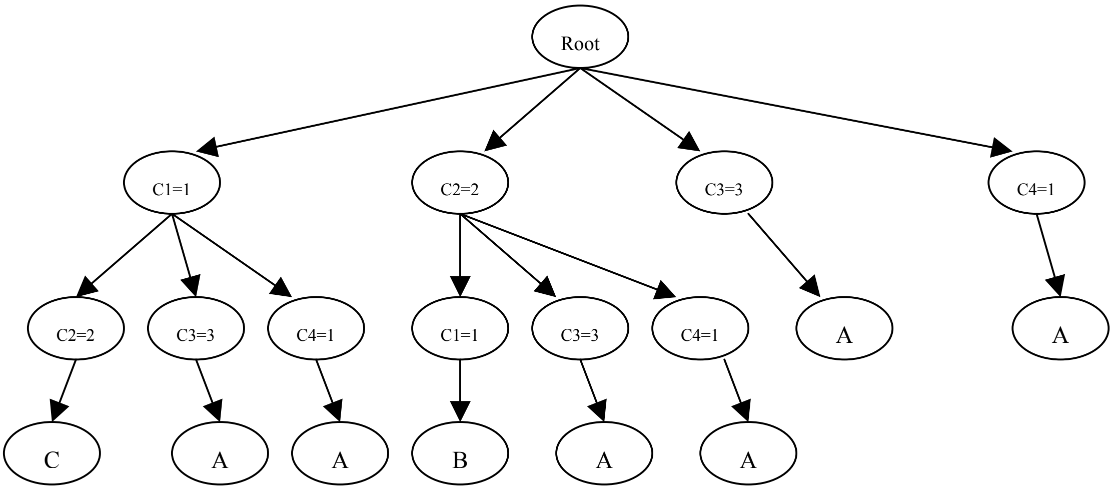

Figure 1. The search graph of solution space
Remark: The notation A,B and C are according to the three conditions of "RDBRST" algorithm

Table 2. The known rules set

r_x	$r_x \mid C \Rightarrow r_x \mid D$	r_x	$r_x \mid C \Rightarrow r_x \mid D$
1	$(C_1 = 2) \Rightarrow (D = 1)$	7	$(C_1 = 1 \wedge C_4 = 2) \Rightarrow (D = 2)$
2	$(C_3 = 2) \Rightarrow (D = 1)$	8	$(C_1 = 3) \Rightarrow (D = 2)$
3	$(C_2 = 1 \wedge C_4 = 3) \Rightarrow (D = 1)$	9	$(C_3 = 1 \wedge C_4 = 3) \Rightarrow (D = 2)$
4	$(C_2 = 2) \Rightarrow (D = 1)$	10	$(C_3 = 3) \Rightarrow (D = 3)$
5	$(C_2 = 3) \Rightarrow (D = 2)$	11	$(C_1 = 1 \wedge C_2 = 1) \Rightarrow (D = 3)$
6	$(C_1 = 1 \wedge C_3 = 1) \Rightarrow (D = 2)$	12	$(C_4 = 1) \Rightarrow (D = 3)$

Table 3. The relevant object set

C1	C2	C3	C4	D
1	3	1	2	2
2	2	1	2	1
1	2	3	1	3

3.3. Analysis of complexity of RDBRST algorithm

Given an information system $S = (U, A, V, f)$, $A = C \bigcup D$, $C = \{C_1, C_2, \ldots, C_m\}$, $D = \{d\}$. The new added object is u, $u = \{u_1, u_2, \ldots, u_m, d_u\}$.

When we get the result only after searching the first level, the space complexity of RDBRST algorithm is m, which is the best result of this algorithm. The worst case of this algorithm is that it should traversal whole search nodes expanded in the procession of search. The number of nodes of k^{th} level of the search tree constructed in RDBRST algorithm is $closed(k) = \prod_{i=1}^{k} (m + 1 - i)$, so the sum of total nodes expanded, in the other words, the worst case of space complexity is $O\left(\sum_{k=1}^{m} closed(k)\right)$. In reality, the number of condition attributes is far less than the number of object in information system, so the space complexity of RDBRST algorithm is superior to original algorithm.

The time complexity of this algorithm is decided by three factors below:

(1) The size of relevant object set.

(2) The number of rules which consequent is equal to $Des([u]_D)$ in the known rule set.

(3) The number of nodes of search tree.

Suppose the size of relevant object set is q, the number of rules match the condition of factor (2) above is p, and the number of nodes of closed set and open set in kth level of the search tree is *closed(k)* and *open(k)*, then the best result of time complexity is $m\times(p+q)$, the worst result is $O\left(\sum_{k=1}^{m}(open(k-1)+p+q)\times closed(k)\right)$. The time complexity of this algorithm is still huge, but in RDBRST algorithm, we can reduce the search space greatly by taking the known rule set as heuristic which can improve the time complexity evidently.

4. The known rule set updated

There are two processes in the incremental rules extraction: new rules derivation and the known rule set updated. The core of our algorithm is RDBRST algorithms discussed above.

With the changes of information system, some of the known rules set should be updated.

Theorem 4.1: Let S be an information system, $S=(U,A,V,f)$, $A=C\bigcup D$. The new added object is u, For $x\in U$, if the relation $\widehat{C}(x)=\widehat{C}(u)\wedge d(x)\neq d(u)$, $\widehat{C}\subseteq C$ is satisfied, then the rules generated according to object x should be updated.

Proof: It follows immediately from the theorem 3.1.

Algorithm 2: Known rule set updated.

(1) $S=(U,A,V,f)$, $A=C\bigcup D$, $C=\{C_1,C_2,...,C_m\}$, $D=\{d\}$

(2)The known rule set *R*.

(3)The new object: $u=\{u_1,u_2,...,u_m,d_u\}$.

(4) $S'=S\bigcup\{u\}$.

(5)The object set *P*, which is the relevant object set according to the new object *u*, and it is one subset of *S'*.

Output: The updated rule set *R*.

Begin:

(1) Take one object *x* from *P*, except *u*.

(2) $\widehat{C}=\{c\mid c\in C\wedge c(x)=c(u)\}$

(3) If rule $Des([x]_{\bar{C}})\Rightarrow Des([x]_D)$ exists in R, and $\bar{C}\subseteq\widehat{C}$, delete this rule from R.

(4) Generate the relevant object set according to *x*, take the object *x* as the new object and the initial open set is $\{Des[x]_{\widehat{C}}\}$. Call the "RDBRST' algorithm, add the output of "RDBRST" algorithm to *R*.

(5) If all object in *P* have been computed, go to End, else go to (1).

End.

We illustrate the above algorithm by the same example as section 3, and the graph of search procession is shown in Figure 2.

According to the known rule set updated algorithm, the rule $(C_2=2)\Rightarrow(D=1)$ need to be updated. Call the "RDBRST' algorithm, the initial open set is $\{C_2=2\}$, and the rule is updated with two rules: $(C_2=2\wedge C_3=1)\Rightarrow(D=1)$ and $(C_2=2\wedge C_4=2)\Rightarrow(D=1)$.

The result of the incremental rules extraction algorithm is shown in table 4.

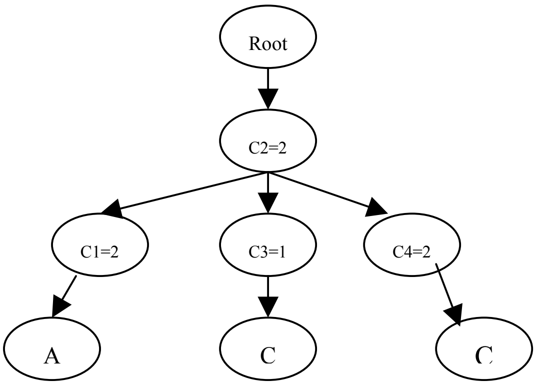

Figure 2. The search graph of known rule set updated.
Remark**:** The notation A and C are according to the condition of RDBRST algorithm in section 3.2

Table 4. The rule set after new object added

r_x	$r_x\mid C\Rightarrow r_x\mid D$	r_x	$r_x\mid C\Rightarrow r_x\mid D$
1	$(C_1=2)\Rightarrow(D=1)$	8	$(C_1=1\wedge C_4=2)\Rightarrow(D=2)$
2	$(C_3=2)\Rightarrow(D=1)$	9	$(C_1=3)\Rightarrow(D=2)$
3	$(C_2=1\wedge C_4=3)\Rightarrow(D=1)$	10	$(C_3=1\wedge C_4=3)\Rightarrow(D=2)$
4	$(C_2=2\wedge C_3=1)\Rightarrow(D=1)$	11	$(C_3=3)\Rightarrow(D=3)$
5	$(C_2=2\wedge C_4=2)\Rightarrow(D=1)$	12	$(C_1=1\wedge C_2=1)\Rightarrow(D=3)$
6	$(C_2=3)\Rightarrow(D=2)$	13	$(C_4=1)\Rightarrow(D=3)$
7	$(C_1=1\wedge C_3=1)\Rightarrow(D=2)$	12	$(C_1=1\wedge C_2=2)\Rightarrow(D=3)$

5. Conclusion

Compared to the existing algorithms, our incremental rules extraction algorithm has better space complexity, especially, when the size of database is large. The time complexity of this algorithm is affected by structure of information system, the known rule set and the relevant object set. How to reduce the time complexity while the accuracy of rules derivation isn't affected is the issue further study.

References

[1] Pawlak Z. Rough set. International Journal of Computer and information science, 1982,11(5):341-356.

[2] Grazymala Busse, J.W.Lers. A system of knowledge discovery based on Rough sets . Tokyo：Proceeding of 5th international workshop RSFD'96, 1996, 443-444.

[3] Yong Liu, Congfu XU, Xuelan Li, Yunhe pan. A parallel approximate rule extracting algorithm based on the improved discernibility matrix. Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), v 3066, Rough Sets and Current Trends in Computing, 2004, p 498-503.

[4] Tong LingYun, An LiPing. Incremental learning of decision rules based on rough set theory. Proceedings of the World Congress on Intelligent Control and Automation (WCICA), v 1, 2002, p 420-425.

[5] YU Dongjun, WANG Shitong, YANG jingyu. An Incremental Rule Extraction Algorithm. Mini_Micro system, vol 25 No1, Jan 2004.

[6] AN Liping, WU Yuhua, TONG Lingyun. Rough Set Approach to Incremental Acquisition of Rules. A cta Scientiarum Naturalium Universitatis Nankaiensis. Vol 36 No 2, Jan 2003.