Efficient Attribute Reduction Based on Discernibility Matrix

Zhangyan Xu[1,2], Chengqi Zhang[3,4], Shichao Zhang[1], Wei Song[2], and Bingru Yang[2]

[1] Department of Computer, Guangxi Normal University, 541004, Guilin China
xyzwlx72@yahoo.com.cn, zhangsc@mailbox.gxnu.edu.cn
[2] School of Information Engineering, Univ. of Science and Technology Beijing, China
{sgyzfr,bryang_kD}@yahoo.com.cn
[3] Faculty of Information Technology, University of Technology, Sydney, Australia
chengqi@it.uts.edu.au
[4] Department of Information Systems,City University of Hong Kong,Hong Kong, China
chengqi@it.uts.edu.au

Abstract. To reduce the time complexity of attribute reduction algorithm based on discernibility matrix, a simplified decision table is first introduced, and an algorithm with time complexity $O(|C||U|)$ is designed for calculating the simplified decision table. And then, a new measure of the significance of an attribute is defined for reducing the search space of simplified decision table. A recursive algorithm is proposed for computing the attribute significance that its time complexity is of $O(|U/C|)$. Finally, an efficient attribute reduction algorithm is developed based on the attribute significance. This algorithm is equal to existing algorithms in performance and its time complexity is $O(|C||U|)$ $+O(|C|^2|U/C|)$.

Keywords: Attribute reduction, simplified decision table, discernibility matrix.

1 Introduction

Recently, some efforts on attribute reduction have focused on dealing with inconsistency in decision information systems. Since Pawlak proposed the attribute reduction based on positive region [1], there has been some work developed for improving the efficiency of the attribute reduction based on positive region [2,3]. Latter, Skowron and Hu proposed the attribute reduction based on discernibility matrix [4,5]. There have been also other types of knowledge reduction [6,7].

In this paper, we study the attribute reduction based on discernibility matrix and to design correspondence attribute reduction algorithm. The attribute reduction algorithm based on discernibility matrix given in [8-14] starts with an empty set of attributes and heuristically adds new attributes one by one, in a greedy way, until a super reduction is constructed. In each loop, if the attribute a_k frequently occurs in the discernibility matrix, it will be added. This is equivalent to choosing the attribute that 'discerns' the largest number of pairs of objects with different decisions. Full details of the algorithm can be found in [8-14]. In these algorithms, the elements of discernibility matrix are used as the heuristic information. So it must be first to calculate the discernibility matrix. The time complexity and space complexity are both $O(|C||U|^2)$, where C and U are attributes set and objects set of a decision table, respectively. The time complexity of these algorithms proposed by authors in [8-10] is $O(|C|^2|U|^2)$. The time complexity of these algorithms proposed by authors in [11-14] is cut down to $O((|C|+\log|U|)|U|^2)$. Hence if the elements of discernibility matrix are used as the heuristic information to design attribute reduction algorithm, the best time complexity of this kind of algorithm is not lower than $O(|C||U|^2)$. On the other hand, it needs large space to store the discernibility matrix. When the data set is very large, algorithm is difficult to operate.

To lower the time complexity of attribute reduction algorithm based on discernibility matrix, we design an algorithm based on the significance of attribute and its time complexity is $O(|C||U|)+O(|C|^2|U/C|)$.

The rest of this paper is as follows. In Section 2, we introduce some basic concepts. We present our algorithm in Section 3 and illustrate the use of the algorithm with an example in Section 4. We summarize this paper in Section 5.

2 Concepts and Definitions

In this section, we introduce the basic concepts and correspondence definitions.

Definition 1. A decision table is defined as $S=(U,C,D,V,f)$, where $U=\{x_1,x_2,\cdots,x_n\}$ is the set of objects, $C=\{c_1,c_2,\cdots,c_r\}$ is the set of condition attributes, D is the set of decision attributes, and $C\cap D=\varnothing$; $V=\bigcup_{a\in C\cup D} V_a$, where V_a is the value range of attribute a. $f:U\times C\cup D\rightarrow V$ is an information function, in which an information value for each attribute of an object, i.e., $\forall a\in C\cup D, x\in U, f(x,a)\in V_a$. Every attribute subset $P\subseteq(C\cup D)$ determines a binary indiscernibility relation $IND(P)$:

$$IND(P)=\{(x,y)\in U\times U\,|\,\forall a\in P, f(x,a)=f(y,a)\}$$

$IND(P)$ determines a partition of U, which is denoted by $U/IND(P)$ (in short *U/P*). Any element $[x]_P=\{y\,|\,\forall a\in P, f(x,a)=f(y,a)\}$ in *U/P* is called equivalent class.

Definition 2. For a decision table $S=(U,C,D,V,f)$, let $U/D=\{D_1,D_2,\cdots,D_k\}$ be the partition of D to U, and $U/C=\{C_1,C_2,\cdots,C_m\}$ be the partition of C to U, where $C_i(i=1,2,\cdots,m)$ is basic block, then $POS_C(D)=\bigcup_{D_i\in U/D} C\_(D_i)$ is called positive region of C on D. If $POS_C(D)=U$, then the decision table is called consistent, else it is called inconsistent.

Theorem 1. For a decision table $S=(U,C,D,V,f)$, there is

$$POS_C(D)=\bigcup_{X\in U/C\wedge\forall x,y\in X\Rightarrow f(x,D)=f(y,D)} X$$

Proof: According to the definition of positive region of C for D, it is easy to know the proposition is right.

According to Theorem 1, we have the following definition of simplicity decision table.

Definition 3. For a decision table $S=(U,C,D,V,f)$, let $U/C=\{[x_1']_C,[x_2']_C,\cdots,[x_m']_C\}$ and $U'=\{x_1',x_2',\cdots,x_m'\}$. For the definition of positive region, there is $POS_C(D)=[x_{i_1}']_C\cup[x_{i_2}']_C\cup\cdots\cup[x_{i_t}']_C$, where $\{x_{i_1}',x_{i_2}',\cdots,x_{i_t}'\}\subseteq U'$ and $\forall x,y\in[x_{i_s}']_C$ $(s=1,2,\cdots,t)$, there are $f(x,D)=f(y,D)$; let $U_{pos}'=\{x_{i_1}',x_{i_2}',\cdots,x_{i_t}'\}$ and $U_{neg}'=U'-U_{pos}'$. It is said that the 5-tuple $S'=(U',C,D,V,f)$ is a simplicity decision table.

Definition 4. For a decision table $S=(U,C,D,V,f)$, we define discernibility matrix $M=(m_{ij})$, whose elements are defined as follow:

$$m_{ij}=\begin{cases}\{c_k \mid c_k\in C, f(x_i,c_k)\neq f(x_j,c_k), f(x_i,D)\neq f(x_j,D)\}\\ \varnothing \quad \text{el se}\end{cases}$$

where $k=1,2,...,r$.

Definition 5. For a decision table $S=(U,C,D,V,f)$, $M=(m_{ij})$ is discernibility matrix, $\forall B\subseteq C$, if B satisfies: (1) $\forall\varnothing\neq m_{ij}\in M$, such that $B\cap m_{ij}\neq\varnothing$; (2) $\forall b\in B$, $B-\{b\}$ is not satisfied (1), then B is called the attribute reduction of C for D based on discernibility matrix.

In next section, we would define a new reasonable formula for measuring the significance of attribute to reduce the search space of simplified decision table.

3 The Significance of Attribute

In these old algorithm based on discernibility matrix, it was first to calculate the discernibility matrix, so the time complexity of the algorithm is not lower than $O(|C||U|^2)$. To cut down the time complexity of the attribute reduction, we designed a new and reasonable formula for measuring the significance of attribute to reduce the search space of the simplicity decision table. In order to propose the significance of attribute, we first introduce the follow proposition.

Definition 6. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table. $\forall B\subseteq C$, we define knowledge of the attribute set B for D as follow:

$$Sig_D(B)=\{\bigcup_{X\in U'/B\wedge(X\subseteq U_{pos}')\wedge|X/D|=1}X\}\cup\{\bigcup_{X\in U'/B\wedge(X\subseteq U_{neg}')\wedge|X/C|=1}X\}.$$

where we consider $Sig_D(\varnothing)=\varnothing$. We can easily know $Sig_D(C)=U'$.

Theorem 2. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table, $M=(m_{ij})$ is the deiscernibility matrix of the old decision table. $\forall B\subseteq C$, if $Sig_D(B)=Sig_D(C)$, there is $\forall\varnothing\neq m_{ij}\in M$ such that $B\cap m_{ij}\neq\varnothing$.

Theorem 3. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table, and $M=(m_{ij})$ is the discernibility matrix of the old decision table. $\forall B \subseteq C$, if $\forall \varnothing \neq m_{ij} \in M$ such that $B \cap m_{ij} \neq \varnothing$, there has $Sig_D(B)=Sig_D(C)$.

Theorem 4. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table, and $M=(m_{ij})$ is the discernibility matrix of the old decision table. $\forall B \subseteq C$, if $Sig_D(B) \neq Sig_D(C)$, there must exist $\varnothing \neq m_{i_0 j_0} \in M$ such that $B \cap m_{i_0 j_0} = \varnothing$.

Theorem 5. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table. $\forall B \subseteq C$, if $Sig_D(B)=Sig_D(C)$ and $\forall b \in B$ there is $Sig_D(B-\{b\}) \neq Sig_D(C)$, then B is an attribute reduction C for D based on the discernibility matrix.

Definition 7. (The significance of attribute) For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table. For $P \subseteq C$, the significance of arbitrary attribute a ($a \in (C-P)$) to attribute set P is defined as follow:

$$I_p(a)=\left|Sig_D(P \cup \{a\})-Sig_D(P)\right| .$$

Theorem 6[3]**.** For a decision table $S=(U,C,D,V,f)$, to $\forall P \subseteq C, \forall a \in (C-P)$, there is $U/(P \cup \{a\}) = \underset{X \in U/P}{\cup}(X/\{a\})$.

Theorem 7. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table. For $P \subseteq C, \forall a \in (C-P)$, there is

$$U'/(P \cup \{a\}) = \underset{X \in U'/P}{\cup}(X/\{a\}) .$$

Theorem 8. For a decision table $S=(U,C,D,V,f)$, $S'=(U',C,D,V,f)$ is its simplicity decision table. For $P \subseteq C, \forall a \in (C-P)$, there is

$$I_p(a)=\left|Sig_D(P \cup \{a\})-Sig_D(P)\right|$$

$$\bigcup\{\underset{X \in U'/P \wedge (X \not\subset U'_{pos} \wedge X \not\subset U'_{neg}) \wedge Y \in X/\{a\} \wedge Y \subseteq U'_{pos} \wedge |Y/D|=1}{\cup} Y\}$$

$$=|\{\underset{X \in U'/P \wedge X \subseteq U'_{pos} \wedge |X/D| \neq 1 \wedge Y \in X/\{a\} \wedge |Y/D|=1}{\cup} Y\}$$

$$\bigcup\{\underset{X \in U'/P \wedge (X \not\subset U'_{pos} \wedge X \not\subset U'_{neg}) \wedge Y \in X/\{a\} \wedge Y \subseteq U'_{neg} \wedge |Y/C|=1}{\cup} Y\}$$

$$\bigcup\{\underset{X \in U'/P \wedge X \subseteq U'_{neg} \wedge |X/C| \neq 1 \wedge Y \in X/\{a\} \wedge |Y/C|=1}{\cup} Y\}| .$$

4 Algorithm for Calculating the Significance of Attribute

According to Definition 6, it is first to calculate the simplified decision table before calculating the significance of attribute. So we first propose an efficient algorithm for calculating the simplified decision table. Calculating the simplified decision table is in fact to calculate the *IND*(*C*). To our best knowledge, the best algorithm for computing *IND*(*C*) is the algorithm of [3] with the time complexity $O(|C||U|\log|U|)$ at present. So we used radix sorting to design a good algorithm for computing *IND*(*C*). And its time complexity is cut down to $O(|C||U|)$.

Algorithm 1. Computing the simplicity decision table

```
Input: Decision table $S=(U,C,D,V,f)$ , $U=\{x_1,x_2,\cdots,x_n\}$ , $C=\{c_1,c_2,\cdots,c_r\}$
Output: $U'_{pos},U'_{neg},U'$ , $M_i,m_i(1\le i\le s)$ .
1.  To each $c_i(i=1,2,\cdots,r)$ , calculate the maximum and minimum of $f(x_j,c_i)$
    $(j=1,2,\cdots,n)$ and denote $M_i$ and $m_i$ respectively;
2.  use state list to store the objects $x_1,x_2,\cdots,x_n$ in turn ; let the head pointer of the
    list point to $x_1$ ;
3.  for (i=1;i<r+1;i++)
      3.1 the ith "distribution": construct $M_i$-$m_i$+1 empty queues, let $front_k$ and
          $end_k$ ($k=0,1,\ldots, M_i-m_i$ ) be the head pointer and tail pointer of the kth
          queue respectively. Distribute the object $x$ of the list $U$ to the
          $f(x,c_i)-m_i$ th queue according to the elements order of list U.
      3.2 the ith "collection": the head pointer of the list points to the head pointer
          of the first nonempty queue, modify the tail pointer of each nonempty
          and let it point to the head object of the next nonempty queue. In this
          way, recombine $M_i-m_i+1$ queues to a new list;
4.  Let the objects sequence of list from Step 3 be $x'_1,x'_2,\cdots,x'_n$ ;
        $t=1$; $B_t=\{x'_1\}$ ;
        for (j=2;j<n+1;j++)
          if any $c_i\in C(i=1,2,\cdots,r)$ there is $f(x'_j,c_i)=f(x'_{j-1},c_i)$ ,
          then $B_t=B_t\cup\{x'_j\}$ ;
          else {  $t=t+1$;  $B_t=\{x'_j\}$ ;  }
5.  $U'_{pos}=\varnothing;U'_{neg}=\varnothing;$
        for ( i=1;i<t+1;i++)
          if any $x,y\in B_i$ there is $f(x,D)=f(y,D)$ , then we take out the first
        object of $B_i$ to $U'_{pos}$ ; Else we take out the first object of $B_i$ to $U'_{neg}$ ;
    $U'=U'_{pos}\cup U'_{neg}$ ;
```

Complexity analyze of Algorithm 1. the time complexity for the first step of algorithm is $O(|C||U|)$; the time complexity for the second step is $O(|U|)$; the time complexity for Step 3.1 is $O(|U|+M_i-m_i+1)$, the time complexity for Step 3.2 is $O(M_i-m_i+1)$, so the time complexity for Step 3 is $O(|C||U|+\sum_{i=1}^{r}(M_i-m_i+1))$; the time complexity for Step 4 is $O(|C||U|)$; the time complexity for Step 5 is $O(|D||U|)$ (the decision attribution usually is only one). Hence the time complexity of algorithm is $O(|C||U|+\sum_{i=1}^{r}(M_i-m_i+1))$. In most condition, especially to the large-scale decision table, there is often $\max_{1\le i\le s}(M_i-m_i+1)\le|U|$ (for example, the mushroom in UCI mushroom, it has more 8000 objects and 22 attributions, but these attribution values are single letter, so there is at most 26 kinds difference values in each attribution.) ,hence $(|C||U|+\sum_{i=1}^{r}(M_i-m_i+1))\le|C||U|+|C||U|$, therefore the time complexity of Algorithm 1 is $O(|C||U|)$. It is easily to know that the space complexity of Algorithm 1 is $O(|U|)$.

According to Theorems 7 and 8, we can design the following efficient algorithm for calculating the significance of attribute.

Algorithm 2. compute $I_p(a)$

```
Input： $S_P=\{X\mid X\in U'/P\wedge((X\not\subset U'_{pos}\wedge X\not\subset U'_{neg})$
        $\vee(X\subseteq U'_{pos}\wedge|X/D|\ne1)\vee(X\subseteq U'_{neg}\wedge|X/C|\ne1))\}$, $M_a$, $m_a$;
Output： $S_{P\cup\{a\}}=\{X\mid X\in U'/(P\cup\{a\})\wedge((X\not\subset U'_{pos}\wedge X\not\subset U'_{neg})$
        $\vee(X\subseteq U'_{pos}\wedge|X/D|\ne1)\vee(X\subseteq U'_{neg}\wedge|X/C|\ne1))\}$ ; $I_p(a)$ ;
1. for (j=1; j<$|S_P|$+1; j++)  $\forall x\in X_j$, let $x.flag=j$; // $X_j\in S_P$
2. Let $T=X_1\cup X_2\cup\cdots\cup X_{|S_P|}=\{x_1,x_2,\cdots,x_z\}$; where $X_j\in S_P(j=1,2,\cdots,|S_P|)$;
3. use state list to store the objects $x_1,x_2,\cdots,x_z$ in turn ;
4. construct $M_a$-$m_a$+1 empty queues, let $front_k$ and $end_k$ ($k=0,1,\dots,M_a-m_a$) be
   the head pointer and tail pointer of the kth queue respectively. Distribute the
   object $x\in T$ of the list to the $f(x,a)-m_a$ th queue according to the elements
   order of list.  //where $M_a,m_a$ are the maximal value and the minimal value of
   the attribute a in the old decision table.
5. $I_p(a)=0$; $S_{P\cup\{a\}}=\varnothing$;
6. It deals with the each not empty queue $Col=\{y_1,y_2,\cdots,y_h\}$ as follow acquired
   from the four step;
     t=1; $B_t=\{y_1\}$;
     for (j=2;j<h+1;j++)
        { if ( $y_j.flag==y_{j-1}.flag$)
               $B_t=B_t\cup\{y_j\}$;
          else { t=t+1; $B_t=\{y_j\}$; };
        }
     for (i=1;j<t+1;i++)
           if ( $B_i\subseteq U'_{pos}\wedge|B_i/D|=1$ )      $I_p(a)=I_p(a)\cup|B_i|$;
           else  if ( $B_i\subseteq U'_{neg}\wedge|B_i/C|=1$ )
                         $I_p(a)=I_p(a)\cup|B_i|$;
                else  $S_{P\cup\{a\}}=S_{P\cup\{a\}}\cup\{B_i\}$;
```

Complexity analyses of Algorithm 2. The time complexity of the first step is $O(|T|)$; The time complexity of the fourth step is $O(|T|)$; The time complexity of the first *for* cycle of the sixth step is $O(|Col|)$. The time complexity of the second *for* cycle is also $O(|Col|)$. So the time complexity of the sixth step is $O(|T|)$. Therefore, the time complexity of Algorithm 2 is $O(|T|)$. Because of $O(|T|) \le O(|U'|)$, the worst time complexity of Algorithm 2 is $O(|U'|)$. The space complexity is $O(|T|)$. For the same reason, the worst space complexity of the algorithm is $O(|U'|)$.

5 Attribute Reduction Algorithm Based on Discernibility Matrix

According to Algorithms 1 and 2, we now can design an efficient attribute reduction algorithm based on discernibility matrix.

Algorithm 3. Attribute reduction algorithm based on discernibility matrix

```
Input: decision table $S=(U,C,D,V,f)$, $U=\{x_1,x_2,\cdots,x_n\}$, $C=\{c_1,c_2,\cdots,c_s\}$;
Output: attribute reduction $R$;
1. It uses the algorithm 1 to calculate $U'=\{u_1,u_2,\cdots,u_m\}$, $m_i, M_i (i=1,2,\cdots,s)$,
   $U'_{pos}$, $U'_{neg}$;
2. let $R=\varnothing$; $S_R=\{\{u_1,u_2,\cdots,u_m\}\}$; // When algorithm begins, input set $S$ is the
   only one equivalence, i.e. $U'$;
3. To each attribute $c_i \in C-R$, calculating $I_R(c_i)$;Denote $I_R(c_k)=\max_{c_i\in C-R} I_R(c_i)$; If
   the attribute like that is not only one, we arbitrary select one ;
4. If $S_{R\cup\{c_k\}}$ is an empty set, then stop the algorithm, output the attribute
   reduction $R\cup\{c_k\}$ ; // $S_{R\cup\{c_k\}}$ is the corresponding output set to calculate
   $I_R(C_k)$.
5. If $S_{R\cup\{c_k\}} \ne \varnothing$, then $R=R\cup\{c_k\}$;The algorithm will turn to step 3;
```

The complexity of Algorithm 3. It can be known from Algorithm 1 that the time and space complexities of the first step are $O(|C||U|)$ and $O(|U|)$ respectively. The worst time and space complexity third step are $O(|U'|)=O(|U/C|)$ from analyzing of Algorithm 2. So the worst time and space complexities of the third step are $O(|U/C||C-R|)$ and $O(|U/C||C-R|)$ respectively. The worst time complexity of the third step to the fifth is $O(|U/C||C|)+O(|U/C||C-1|)+\cdots+O(|U/C|)$ $=O(|U/C||C|^2)$. The worst space complexity of the third step to the fifth is $O(|U/C||C|)$. Therefore the worst time and space complexities of Algorithm 3 are $(O(|C||U|)+O(|C|^2|U/C|)$ and $O(|U|)+O(|C||U/C|)$ respectively.

6 Conclusion

At present, the elements of discernibility matrix are used as the heuristic information by all the existing attribute reduction algorithms based on discernibility matrix. In these algorithms, it is first to calculate the discernibility matrix. Because it must be first to calculate the Skowron's discenibility matrix in this kind of attribute reduction algorithm, the best time complexity of this kind algorithm is not lower than $O(|C||U|^2)$. On the other hand, it needs the large space to store the discernibility matrix. Once the data set is very large, algorithm is difficult to operate. To lower the time complexity of attribute reduction algorithm based on discernibility matrix, firstly, the simplified decision table and the significance of attribute are introduced. Then an efficient attribute reduction based on the significance of attribute is proposed. And And it is proved that our algorithm is equivalent to existing algorithms in performance and the time complexity is $O(|C||U|) + O(|C|^2|U/C|)$.

Acknowledgement

This work is partially supported by Australian Large ARC Grants (DP0449535, DP0559536 and DP0667060), a China NSF Major Research Program (60496327), a China NSF Grant (60463003), an Overseas Outstanding Talent Research Program of the Chinese Academy of Sciences (06S3011S01), and an Overseas-Returning High-level Talent Research Program of China Hunan-Resource Ministry.

References

1. Pawlak, Z.: Rough set theory and its application to data analysis. Cybernetics and Systems, 9(1998) 661-668
2. Guan, J.W., Bell ,D.A.: Rough computational methods for information systems, Artificial intelligence, 105(1998) 77-103
3. Nguyen, S.H., Nguyen, H.S.: Some efficient algorithms for rough set methods. In Proceedings of the Conference of Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU'96, Granada, Spain, (1996) 1451-1456
4. Skowron, A., Rauszer, C.: The discernibilinity matrices and functions in information systems. In: R. Slowincki (ed), Intelligent decision support-handbook of applications and advances of the rough sets theory, Dordrecht:Kluwer,(1992) 331-362
5. Hu, X.H., Cercone, N.: Learning in relational databases: A rough set approach. International journal of computational intelligence, 11(1995) 323-338
6. Kkryszkiewicz, M.: Comparative study of alternative types of knowledge reduction in inconsistent systems. International Journal of Intelligent Systems. 16(2001) 105-120
7. Li, D.R., Zhang, B.: On knowledge reduction inconsistent decision information systems. International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, 50(2004) 651-672
8. Wang, J., Cui, J., Zhao, C.: Investigation on AQ11, ID3 and the Principle of Discernibility Matrix. Journal of Computer Science and Technology, 16(2001)1-12
9. Wang, J., Wang, J.: Reduction algorithms based on discernibility matrix: the ordered attributes method, Journal of Computer Science and Technology, 16(2001)489-504
10. Zhang, J., Wang, J., Li, D.(ed.): A new heuristic reduction algorithm base on rough sets theory. Lecture Notes in Computer Sciences, 2762(2003): 247- 253.

11. Wang, B., Chen, S.B.: A complete algorithm for attribute reduction. T.-J. Tarn et al. (Eds.): Robotic Welding, Intelligence and Automation, Lecture Notes in Computer Sciences, Vol. 299. Springer -Verlag Berlin Heidelberg New York(2004)
12. Bazan, J., Nguyen, H.S., Nguyen, S.H.(ed.): Rough set algorithms in classification problems. In Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems. Vol.56. Physica-VerlagNew York(2000)
13. Hu, K.Y., Diao, L.L., Lu, Y.C.(ed.): A Heuristic Optimal Reduction Algorithm. Leung, K.S., Chan, L.W., Meng, H.(Eds.): IDEAL 2000, LNCS.Vol.1983. Springer-Verlag Berlin Heidelberg New York(2000)
14. Korzen, M., Jaroszewicz, S.: Finding reducts without building the discernibility matrix. In Proc. of the 5th Int. Conf. on Intelligent Systems Design and Applications (ISDA'05)