

A Parallel Reduct Algorithm Based On Extension Matrix

Pei Li Zhou
Faculty of Information Technology
Monash University
Victoria 3145
Australia

Abstract

In this paper, on the basis of studying the limitations of the basic rough set model, we present a general rough set model, which is based on a family set of tolerance relations which is based on a family set of tolerance relations instead of equivalence relations between objects. The model inherits most of the characteristics of the basic model of rough set; and they also have a better effect of approximation classification. Based on this model, we propose two algorithms, one for a single processor and the other for parallel processors that will give us a near-optimal attributes reduct using concepts from extension matrix.

KEYWORDS: Machine Learning, Rough Set, Reduct, Information System, Extension Matrix, Tolerance Relation, Knowledge Acquisition, Knowledge Representation, Parallel algorithms

1. Introduction

In the early 1980s Professor Pawlak proposed rough set theory on the basis of research on uncertainty, vagueness, and uncompleted knowledge representation [1, 2, 3, 4]. It is a new mathematical tool to deal with vagueness and uncertainty. Rough set theory has some advantages over other similar formal tools, and has been used widely in the areas of information

system analysis, artificial intelligence and its application, knowledge and data discovery, pattern recognition and classification and faulty detection [5].

In an approximation space, it is required that a set of equivalence relations is given first, and it also contains one kind of definitely formal method, which will map any subset of equivalence relations into an equivalent relation on the universe. But, in real world practices, there is not always possible. In this paper, we are going to lower this requirements, and we shall use a set of tolerance relations which be resulted by known attributes on a subset of objects to define a new kind of approximation space. In such a given approximation space, we have to define a mapping relation like above as well. In general speaking, we couldn't give one unified and well-defined method to construct the mapping from a set of tolerant relations to one tolerant relation. Thus, a different mapping relation will determine a different approximation space and an information system as well.

Firstly, we are going to give the definition of Tolerance Approximation Space and, and its main concepts in a tolerance approximation space model. Then, A f_{β} -tolerance-membership function will be given, and based on its definition, we'll discuss the set approximation. At last, we will fully describe one single processor algorithm to

solve a near optimal reduct and its parallel processor version which is far more efficient.

2. Tolerance Approximation Space

A concept is a set of objects, the concept family set (classes) is the knowledge on universe U , and the family set of classes on U could be known as one knowledge base of universe U . In other words, approximation space is a knowledge base which is the set of classification methods.

When we could only do partial classification, the classification done by the rough set model is completely correct and certain, it could not give a classification with a kind of controllable misclassification. However, in the real situation, this kind of classification could give better understand and process of the analyzed data. Secondly, another limitation of approximation space is from the viewing which the universe U of the considering data objects are all known, and the derived result from that model is only usable for that objects set. However, in real life situations, we obviously need to extend the result derived from the limited object set to a larger data set. It is because of the model proposed by Ziarko Z - Variable precision rough sets model with asymmetric bounds [6, 7, 8], through the thinning the boundaries of the concepts, we could get a approximation classification with a certain misclassification error.

Except this, we could also tell that the classification of objects in the approximation space is based on the approximation classification of equivalence relations. As long as there is one value difference of one attribute between two objects, then they could be classified to different classes. This kind of classification is very restrictive, it pays too much attention to the differences between objects, it doesn't see the similarities between objects. So these kinds of classification methods don't benefit the knowledge discovery of very large data sets.

Specially when the attribute values are continuous data, we often need to do discretization. Real value discretization is also a difficult problem of knowledge discovery [9]. So, Skowron A and Stepaniuk J has proposed a general approximation space concept which uses a tolerance relation among objects to process approximation classification [10].

Based on the above concerns, we would like to take a step further into defining the concept of approximation space of rough set. First of all, we take a set of tolerance relations to represent the given knowledge. Based on the given knowledge, when we classify the object sets, not only we have to consider the differences of the objects in between every tolerance relation, but also the similarities of the objects in between every tolerance relation. Secondly, every tolerance relation is defined on a subset of the all objects set. Because in practice, not every object under every given tolerance relation could be classified. For example, missing values of attributes in the data table is one of the important problems of KDD. Using the proposed model of extended rough set model from this paper, we could partially solve this missing values problem. Thirdly, the definition of approximation space by us, is related to the mapping a set of tolerance relations into one tolerance relation on the whole objects set (universe). The basic components of the indiscernibility relation are formed by the intersection of a set of equivalence relations. In tolerance relations, it is very difficult to define such kind of basic components; based on this kind of basic components, we can apply the operations of set theory to form a tolerance relation based on a set of tolerance relations. This tolerance relation is used to described the tolerance on all objects set (universe).

A binary relation R on objects set X is called a *tolerance relation* if and only if relation R is self-reflective and systemic. Obviously, tolerance relation is kind of tolerance relations. Here, the

“tolerance” means the differences among objects in the same tolerance class are acceptable.

Definition 1 A tolerance approximation space is a triple $\mathbf{K} = (U, \mathbf{R}, \tau)$, where U is a non-empty objects universe; $\mathbf{R} = \{R_1, R_2, \dots, R_n\}$, in which $R_i \in \mathbf{R}$ is a tolerance relation on $X_i \subseteq U$; τ is a mapping defined as follow:

$$\tau : \text{Powerset}(\mathbf{R}) - \{\emptyset, \{R_1\}, \{R_2\}, \dots, \{R_n\}\} \rightarrow \mathbf{TR}$$

where \mathbf{TR} is the family of tolerance relations on universe U .

Below we will further discuss on our definitions. In the tolerance approximation space, we treat the known knowledge as a set of tolerance relations on a subset of Universe U . It has two meanings: firstly, because of the incompleteness of the knowledge, we sometimes won't be able to find out whether there exists one relation of the universe U or not, only have the relation of its some subsets; secondly, this relation is a kind of tolerance relation.

In basic rough set model, because equivalence relations have good mathematical properties, equivalence classes can be used as the basic components of knowledge. The indiscernibility generated from a set of equivalence relations is also an equivalence relation, where the basic components are formed by the intersection of the set of equivalence classes of equivalence relations. Similarly, in the tolerance approximation space, we still need to find the mapping of the tolerance relation on universe from a set of tolerance relations. But, this kind of tolerance relation can not be uniform defined by using the operators of general set theory. In other words, in approximation space, if the method of dedicated mapping τ definition changes, then the meaning of corresponding tolerance approximation space changes too. In section 4, we will research on a few methods for how to define a mapping τ . Note that, given a set of tolerance relations are defined on different subset of universe, and the mapping τ is defined on the universe U . This means that one

specific tolerance approximation must clearly define the tolerance relation in between all objects in universe U .

Let $B \in \mathcal{P}(\mathbf{R}) - \{\emptyset, \{R_1\}, \{R_2\}, \dots, \{R_n\}\}$, if $\tau(B) = I_B$, where $I_B \in \mathbf{TR}$, then we call I_B is the confirmed tolerance relation of the tolerance relation set B . Specially, when $B = \mathbf{R}$, we call $I_{\mathbf{R}}$ is the whole tolerance relation of the tolerance approximation space. Tolerance approximation space with a decision classification can be defined as a four-tuple $\mathbf{K} = (U, \mathbf{R}, d, \tau)$, where d is an equivalence relation on universe U , the rest symbols has the same meaning as defined earlier.

Definition 2 Given a tolerance relation I on objects set X , and the tolerance set of object $x \in X$ can be denoted as $TS_I(x)$, its definition is as follows: $TS_I(x) = \{y \mid (y \in X) \wedge (xIy)\}$.

When the tolerance relation I is clear, $TS_I(x)$ can be simply noted as $TS(x)$. The set $TS(X) = \{TS(x) \mid \forall x \in X\}$ is a set of the tolerance sets of all objects in the objects set X . In general, $TS(X)$ may not generate one classification of objects set X , only if tolerance relation I is an equivalence relation, $TS(X)$ is one classification of set X .

It is easy to proof that Given a tolerance relation I on objects set X , and objects $x, y \in X$, then $y \in TS(x)$ if and only if $x \in TS(y)$. This means that any object y of every tolerance set $TS(x)$ about objects set X , its tolerance set $TS(y)$ also contains object x .

The tolerance approximation space can be degenerated into the approximation space of basic rough set model. Given tolerance approximation space $\mathbf{K} = (U, \mathbf{R}, \tau)$, where $\forall R \in \mathbf{R}$, the set $\{TS(x) \mid \forall x \in U\}$ can generate one classification of Universe U , and mapping τ is defined as IND , at this time the tolerance approximation space \mathbf{K} is the approximation space of the basic rough set model.

2.1. Tolerance Member Function and Set Approximation

In the basic rough set model, in order to tell whether one object x is belong to object set X or not, is dependant on knowing whether the equivalence classes which contain object x is contained by set X or not. In tolerance rough set model, we can utilize the tolerance set of object x to achieve the above goal. In order to generate the approximation classification with uncertainty or acceptable mis-classification error, we firstly defined a new thinner version of membership function which is based on the concepts proposed by [6] and [10].

Definition 3 f_β -Membership Function

In approximation space $\mathbf{K} = (U, \mathbf{R})$, let $x \in X$ and $X \subseteq U$. The f_β -membership function between object x and objects set X is defined as: $\mu(f_\beta)(x, X) = f_\beta(t)$, where $t = \text{card}(X \cap \text{TS}(x)) / \text{card}(\text{TS}(x))$. Here \mathbf{R} means $IND(\mathbf{R})$; $0 \leq \beta < 0.5$; f_β is regarded as a *boundary thinning function*, it is defined on the closed interval $[0, 1]$ as one non-decreasing function, i.e., $f_\beta: [0, 1] \rightarrow [0, 1]$, and the mapping satisfies $f_\beta(t) = 0$ if and only if $0 \leq t < \beta$; $f_\beta(t) = 1$ if and only if $1 - \beta \leq t < 1$.

In this definition, we just give the rough membership function value of approximation space another f_β mapping. The reason that we introduce f_β is to redefine the rough membership function to get thinner boundary, so we can reach the objective of having approximation classification with controllable misclassification. The dedicated definition of β and f_β can be clearly defined by using the domain knowledge in real world implementations. This is different to basic rough set model. In basic rough set model, it is definitely not allowed to use any knowledge other than given data tables. Using certain domain knowledge is one of the basic ideas of this paper. Note that, if $\beta = 0$ and let f_β become the identity mapping of interval $[0, 1]$, then the f_β -membership

function $\mu(f_\beta)$ is the same as definition of rough membership function.

Definition 4 f_β -tolerance-membership function

Given a tolerance approximation space $\mathbf{K} = (U, \mathbf{R}, \tau)$, let $x \in X$, $X \subseteq U$, $0 \leq \beta < 0.5$; f_β , and $\tau(\mathbf{R})$. The f_β -tolerance-membership function $\mu(I, f_\beta)$ of object x and set X can be defined as: $\mu(I, f_\beta)(x, X) = f_\beta(t)$, where $t = \text{card}(X \cap \text{TS}(x)) / \text{card}(\text{TS}(x))$. Specially, if $\beta = 0$ and f_β is an identity mapping, then we call $\mu(I, f_\beta)$ is a tolerance membership function.

The purpose of tolerance membership function is to convert the problem from identifying whether a object x belongs objects set X or not, to identifying whether the tolerance set $\text{TS}(x)$ is contained by set X or not. Generally, this kind of membership function is related to the specific β . Furthermore, the above definition is subjective to the tolerance relation $\tau(\mathbf{R}) = I$, in fact it can be used in any tolerance relation on any objects set.

Then we can give definition of Lower Approximation and Upper Approximation. In approximation space $\mathbf{K} = (U, \mathbf{R}, \tau)$, the lower approximation and the upper approximation of objects set X are defined as: $L(\mathbf{K}, X) = \{x \mid (x \in X) \wedge (\mu(I, f_\beta)(x, X) = 1)\}$, and $U(\mathbf{K}, X) = \{x \mid (x \in X) \wedge (\mu(I, f_\beta)(x, X) > 0)\}$.

In a data table, $a(x)$ - the value of object x given by the primitive attribute a , could be viewed as the preliminary concept of object x , that is the name of the equivalence class $IND(a)$, it also is said that $a(x)$ is the name of $[x]_{IND(a)}$. The name of the basic concept which contains the object x and defined by the attribute subset $B \subseteq A$, is similar to the arbitrary set of (**attribute, value**), that is $\{(a, a(x)) \mid a \in B\}$. Let $P \subset A$ and $x_i, x_j \in U$, if $\forall p \in P$ all have $p(x_i) = p(x_j)$, then x_i and x_j are not discernible according to P . In fact, an attribute is the equivalence class (a kind of classification method of objects) of the objects, so attribute set A is the set of equivalence classes. The concept of knowledge dependency, (which is the question of

whether every equivalence relation is necessary or not in the approximation space), is interpreted as the equivalent question of whether it also means all the attributes are necessary to keep or not. If $B \subset C$, define the positive region of B in $IND(D)$ is $POS_B(D) = \cup \{B_+(X) : X \in U/IND(D)\}$, if $POS_B(D) = POS_{B-(p)}(D)$, then regard the attribute $p \in B$ to D is not necessary; otherwise p is known as indiscernible. Obviously, we could delete the unnecessary attributes without affecting the classification result. The core C - the indiscernible attributes to D in the conditional attribute C , is denoted as $CORE(C)$; if $B \subset C$ and is the smallest subset which satisfies $POS_C(D) = POS_B(D)$, then we call B one reduct of C .

3. Description of the Algorithm Based on Extension Matrices

In this section, we will give out one algorithm which solves a near optimal reduct. it uses the concepts of positive and negative examples (negative extension matrix) from extension matrix [11, 12].

3.1 The proposed single processor algorithm to solve a near optimal reduct

Given a tolerance decision information system $S = (U, A, \{d\}, \tau)$, we propose the algorithm below to solve the near optimal reduct R of attribute set A .

Algorithm 3.1.1 the algorithm to solve the near optimal reduct based on tolerance extension matrix.

Input: A given tolerance decision information system $S = (U, A, \{d\}, \tau)$;

Output: one near optimal reduct R of attribute set A ;

Begin

Divide all objects (U) into positive and negative examples, according decision attribute (d) from the given tolerance decision information system $S = (U, A, \{d\}, \tau)$;

$R = \emptyset$;

Construct Negative Extension Matrices from all positive examples;

Reduct Matrix $M = \cap$ all Negative Extension Matrix ;

(Negative Extension Matrix 1 \cap Negative Extension Matrix 2 \cap Negative Extension Matrix 3 \cap Negative Extension Matrix n)

$R =$ Remaining attributes that still contain values in M ;

Output Reduct R ;

End

The above are the steps of the algorithm. Now we give out an example using the above algorithm.

3.2 An example of the above algorithm

Example below uses the decision table provided by paper [10] (table 3.1) to explain the process/steps of algorithm 3.1.1.

	Height	Weight	haiR	Eyes	D
1	Short	Light	Dark	Blue	1
2	Tall	Heavy	Dark	Blue	1
3	Tall	Heavy	Dark	Brown	1
4	Tall	Heavy	Blond	Brown	1
5	Short	Light	Blond	Brown	1
6	Tall	Heavy	Red	Blue	2
7	Short	Light	Blond	Blue	2
8	Tall	Heavy	Blond	Blue	2

Table 3.1 The decision table of a tolerance information system S

We could now construct the tolerance decision information system based on the decision table above. It is $S = (U, A, \{d\}, \tau)$, where $U = \{1,2,3,4,5,6,7,8\}$; condition attribute set $A = \{H,W,R,E\}$, where attributes are Height, Weight, haiR, Eyes, they all generate the equivalence relation on universe U ; decision attribute is d .

Mapping τ Is defined as: $\forall B \subseteq A$, where $card(B) \geq 1$, if $\forall x, y \in U$ on attribute subset B have values different on one or more than one attribute, then these two objects are non-tolerant, otherwise they are tolerant.

We then divide U into positive and negative

examples according to the decision attribute d . We have positive examples {1,2,3,4,5} and negative examples {6,7,8}. So we now ready to construct negative extension matrices for the positives examples and derive reduct matrix \mathbf{M} from all negative extension matrices.

Step 1:

Negative Extension Matrix 1:

	Height	Weight	haiR	Eyes
1	Short	Light	Dark	Void
2	Void	Void	Dark	Void
3	Void	Void	Dark	Brown
4	Void	Void	Blond	Brown
5	Short	Light	Blond	Brown

Constructed for Positive Example 1:

[Tall Heavy Red Blue]

Step 2:

Note, please here onwards, we will denote 'Void' as '*'.
Negative Extension Matrix 2:

	Height	Weight	haiR	Eyes
1	*	*	Dark	*
2	Tall	Heavy	Dark	*
3	Tall	Heavy	Dark	Brown
4	Tall	Heavy	*	Brown
5	*	*	*	Brown

Constructed for Positive Example 2:

[Short Light Blond Blue]

Step 3:

Negative Extension Matrix 3:

	Height	Weight	haiR	Eyes
1	Short	Light	Dark	*
2	*	*	Dark	*
3	*	*	Dark	Brown
4	*	*	*	Brown
5	*	Light	*	Brown

Constructed for Positive Example 3:

[Tall Heavy Blond Blue]

Step 4:

Negative Extension Matrix 1 \cap Negative Extension Matrix 2:

	Height	Weight	haiR	Eyes
1	*	*	Dark	*
2	*	*	Dark	*

3	*	*	Dark	Brown
4	*	*	*	Brown
5	*	*	*	Brown

Step 5:

Negative Extension Matrix 1 \cap Negative

Extension Matrix 2 \cap Negative Extension Matrix

3:

	Height	Weight	haiR	Eyes
1	*	*	Dark	*
2	*	*	Dark	*
3	*	*	Dark	Brown
4	*	*	*	Brown
5	*	*	*	Brown

This above matrix is the Reduct Matrix \mathbf{M} we

illustrated in algorithm 3.1.1. Take out the

remaining attributes whose values are not void,

and then we have two attributes: **Hair** and **Eyes**.

We could tell that our near optimal reduct (Hair,

Eyes) is in this case the optimal reduct that could

be solved [13]. Please note that our algorithm may

not always give the optimal reduct, but it will

certainly contain one optimal reduct, so we always

will have a near optimal reduct. Please note for

space reason we have used a very small table and

did not define tolerance relations on attributes, so

in this example, our tolerance relations are the

same as equivalence relations. For uses of

tolerance relations in tolerance information

systems to solve reducts, refer to reference [14].

4 The Proposed Parallel Algorithm

In algorithm 3.1.1 discussed in Section 3.1, we

can see that each a_i value in a positive example is

matched with all the a_i values of the negative

examples. Our proposed algorithm exploits this

property to do the same computation in parallel in

a mesh of $m \times n$ processors. In the mesh, a

processor in row i and column j is denoted as $p_{i,j}$,

for $0 \leq i < m$ and for $0 \leq j < n$.

The algorithm starts by partitioning U

row wise into u and v , where u consists of all the

positive examples and v consists of all the

negative examples of U . Let $|u|$ and $|v|$ be the

number of examples in u and v respectively. Also, let $u_{i,j}$ represent the value of a_i in the positive example i , and $v_{i,j}$ represent the value of a_j in the negative example i .

The algorithm starts by evenly distributing the values in u into $m \times n$ partitions. Each partition is labelled as $g_{i,j}$ where $0 \leq i < m$ and $0 \leq j < n$. Each $g_{i,j}$ consists of $u_{s,t}$, where $i \times \frac{|u|}{m} \leq s < (i+1) \times \frac{|u|}{m}$ and $j \times \frac{k}{n} \leq t < (j+1) \times \frac{k}{n}$.

Similarly v is partitioned in to n partitions. Each partition is labelled as h_i , where $0 \leq i < n$. Each h_i , consists of $v_{j,i}$, for $i \times \frac{k}{n} \leq j < (i+1) \times \frac{k}{n}$ and for $0 \leq j < |v|$.

The algorithm then transfers the values in $g_{i,j}$ and h_j to $p_{i,j}$. Then each $p_{r,c}$ will compute:

$$w_{i,j}^s = u_{i,j} \otimes v_{s,j} \quad (1)$$

Where $r \times \frac{|u|}{m} \leq i < (r+1) \times \frac{|u|}{m}$, $c \times \frac{k}{n} \leq j < (c+1) \times \frac{k}{n}$, and $0 \leq s < |v|$. $u_{i,j} \otimes v_{s,j} = u_{i,j}$ if $u_{i,j} = v_{s,j}$, and $u_{i,j} \otimes v_{s,j} = null$, if $u_{i,j} \neq v_{s,j}$.

At last, each $p_{r,c}$ will compute:

$$Z_{i,j} = w_{i,j}^0 \cap w_{i,j}^1 \cap w_{i,j}^2 \cap \dots \cap w_{i,j}^{|v|-1} \quad (2)$$

Where $r \times \frac{|u|}{m} \leq i < (r+1) \times \frac{|u|}{m}$ and $c \times \frac{k}{n} \leq j < (c+1) \times \frac{k}{n}$.

5 Cost Analysis

The cost of the proposed parallel algorithm consists of communication cost and computation cost, which we will denote as C_{comm} and C_{comp} respectively. The communication cost is the time taken to transfer the g and the h values from processor $p_{0,0}$ to processor $p_{m-1,n-1}$, while the computation cost is the time taken to compute the w and z values by any one of the processors, say

$p_{m-1,n-1}$.

The communication cost is further divided into $C_{startup}$ cost and C_{data} cost. The $C_{startup}$ is the cost of packing the data to be transferred at the source and then unpacking it at the destination. The C_{data} is the cost of transferring one attribute value from one processor to another.

The communication cost of the proposed parallel algorithm is the cost of transferring the g and the h values from $p_{0,0}$ to $p_{m-1,n-1}$. The cost of transferring the g values is $(t_{startup} + \frac{|u|}{m} \times \frac{k}{n} \times t_{data}) \times (m+n-2)$ and the cost of transferring the h values is $(t_{startup} + |v| \times \frac{k}{n} \times t_{data}) \times (m+n-2)$. Therefore,

the total communication cost is

$$C_{comm} = (2 \times t_{startup} + \frac{k}{n} \times t_{data} \times (\frac{|u|}{m} + |v|)) \times (m+n-2) \quad (3)$$

Computation cost of the proposed parallel algorithm is the cost of computing the w and the z values. The cost of computing w values is $\frac{|u| \times k}{m \times n} \times |v|$. The cost of computing the z values is $\frac{|u| \times k}{m \times n} \times (|v|-1)$. Therefore, the total computation cost is

$$C_{comp} = \frac{|u| \times k \times (2 \times |v| - 1)}{m \times n} \quad (4)$$

Hence, the total cost of the proposed parallel algorithm is:

$$C_p = C_{comp} + C_{comm} \quad (5)$$

Computation/Communication Ratio. The communication complexity is $O(\frac{k}{m} \times |u|)$ and the computation complexity is $O(\frac{k}{m} \times \frac{|v|}{n} \times |u|)$.

The ratio of $\frac{t_{comp}}{t_{comm}}$ shows that as $|U|$ increases,

the effect of the communication cost decreases,

thus the performance of the parallel algorithms increases.

Speed up is the measure of how much faster the parallel algorithm is compared to the fastest single processor algorithm known to solve the same problem. The cost of the single processor algorithm discussed in Section 3.1 is

$$C_s = |u| \times |k| \times (2 \times |v| - 1).$$

Hence, for a given $m \times n$

mesh, the speedup, which is $\frac{C_s}{C_p}$ will be closer to

$m \times n$ as $|U|$ increases.

6 Conclusion

In short, through this paper, on the basis of studying the limitations of the basic rough set model, we present an extended rough set model that is based on a family set of tolerance relations between objects when given a set of tolerance relations. This model inherits most of the characteristics of the basic model of rough set; and it also has a better effect of approximation classification. And the concepts of general and specific tolerance information systems are presented, so they can be further used to illustrate the relationships between objects in various real applications.

In the extended rough set model based on the tolerance relations proposed here, according to normal primitive data tables, we could set up a corresponding tolerance information system. On the tolerance information system, we studied the uses of the negative and positive examples concepts from Extension Matrix; from there we proposed two algorithms, one for a single processor and the other for multi processors, that gave us near-optimal attribute reduct based on Extension Matrix Approach. Due to the nature of the extension matrices used in algorithm 3.1.1, our parallel version of that algorithm is very efficient.

References

- [1] Pawlak Z. Rough sets. *International Journal of Computer and Information Science*, 1982, 11(5): 341-356.
- [2] Pawlak Z. *Rough Sets: Theoretical Aspects of Reasoning about Data*. Dordrecht: Kluwer Academic Publishers, 1991
- [3] Pawlak Z, Grzymala-Busse J, Slowinski R, Ziarko W. Rough sets. *Communications of the ACM*, 1995, 38(11): 89-95.
- [4] Pawlak Z. Rough sets present state and further prospects. ICS Research Report 32/95, Warsaw University of Technology, Poland, 1995.
- [5] Alagar V S, Bergler S, Dong F Q eds. *Incompleteness and Uncertainty in Information Systems*. London: Springer-Verlag, 1994.
- [6] Ziarko W. Variable precision rough set model. *Journal of Computer and System Sciences*, 1993, 46: 39-59.
- [7] Katzberg J D, Ziarko W. Variable precision rough sets with asymmetric bounds. In: Ziarko W P ed. *Rough Sets, and Fuzzy Sets and Knowledge Discovery (RSKD'93)*. London: Springer-Verlag, 1994, 167-176.
- [8] Shan, N, Hu X H, Ziarko W. A generalised rough sets model. In: Shi Z Z eds. *Proceedings of the 3rd Pacific Rim International Conf. On Artificial Intelligence*, (Vol, 1). Beijing: International Academic Publishers, 1994, 437-443.
- [9] Cai Y, Cercone N, Han J. Attribute-oriented induction in relational databases. In: Piatetsky-Shapiro L, Frawley W J eds. *Knowledge Discovery in Databases*. Menlo Park, California: AAAI Press / The MIT Press, 1991, 213-228.
- [10] Skowron A, Stepaniuk J. Generalized approximation spaces. In: Lin T Y ed. *Conference Proceedings of the Third International Workshop on Rough Sets and Soft Computing (RSSC'94)*. San Jose, California, USA, 1994, 156-163.
- [11] Hong, J. R. 1985. AE1: An Extension Matrix Approximate Method for the General Covering Problem. *International Journal of Computer and Information Science* 14: 421-437.
- [12] J.R. Hong and C. Uhrig, *The Extension Matrix Approach to Attribute-Based Learning*, *Progress in Machine Learning*, I. Bratko and N. Lavrac (Eds.), Wilmslow: Sigma Press, England, 1987.
- [13] Zhou P L. *Data Mining Using an Extending Rough Set Model*. Monash University, Australia, 2001.
- [14] Zhou P L. *Dynamic Reducts of Tolerance Information Systems*. *The International Conference on Artificial Intelligence (ICAI'03)*. Las Vegas, Nevada, USA, 2003.