A generalized algebraic approach to finding rough set approximations and generating logic rules

D. Sitnikov[1], O. Ryabov[2], O. Titova[1] & O. Romanenko[1]
[1]*Kharkov State Academy of Culture, Ukraine*
[2]*National Institute of Advanced Industrial Science and Technology, Japan*

Abstract

The rough set concept is a relatively new mathematical approach to vagueness and uncertainty in data. The rough set theory is a well-understood formal framework for building data mining models in the form of logic rules, on the basis of which it is possible to issue predictions that allow the classification of new cases. The indiscernibility relation and approximations based on this relation form the mathematical basis of the rough set theory. The classical topological definitions of rough approximations are based on this relation. Unlike the classical approaches it is possible to define rough approximations in an algebraic way. This paper represents a generalization of the algebraic approach suggested by the authors earlier. We use a set of discrete characteristic functions taking on values from finite sets (not necessarily Boolean values) and operations on them including comparison and Boolean operations, which we call the *approximation language*. We use the terms "*exact upper approximation*" and "*exact lower approximation*" to stress the fact that there can exist a variety of approximations but it is always possible to select the approximations that *cannot be improved* in the terms of the approximation language. We consider the process of generating logic rules based on the exact approximations in the case of arbitrary discrete characteristic functions taking on values from finite sets. Logic rules are naturally obtained from predicate formulae for the exact approximations. The introduced approach allows the generation of logic rules quickly and efficiently since only comparison operations with discrete values and Boolean operations with binary values are used to produce logic formulae.

1 Introduction

Data mining has to deal with making decisions under uncertain conditions. It often happens that many rows in a database can be interpreted as the indication of the fact that a new case (for example, whether or not a patient has a disease) should be classified as positive (the patient does have the disease). Nevertheless, many other rows state that patients with the same symptoms do not have the disease. Which decision should be taken when the database contains contradictory information? In order to enable the analyst to make decisions in situations where no conclusion can be drawn with a probability of 100%, various mathematical theories have been developed.

The rough set concept is a relatively new mathematical approach to vagueness and uncertainty in data [1, 2]. The indiscernibility relation and approximations based on this relation form the mathematical basis of the rough set theory. Classical definitions of lower and upper approximations were originally introduced to describe some topological properties of rough sets [3]. These definitions were proposed with reference to an indiscernibility relation, which was assumed to be an equivalence relation. Various generalized definitions of rough approximations have been developed, the majority of them dealing with more general types of relations [4]. In particular, some authors considered a tolerance relation (reflexive and symmetric) as a basis for approximations, which allowed them to express weaker forms of indiscernibility [5]. In some papers researchers proposed interesting definitions of rough approximations based on a similarity relation [6].

The authors of the above papers follow the classical topological way of defining ambiguity concepts. They start from introducing an indiscernibility relation, define some special properties for it and then lower and upper approximations appear in a natural way. Unlike these papers we suggested a different method for treating ambiguity concepts. In our previous paper [7] we considered an approach that did not require introducing any indiscernibility relation but used only predicates in terms of which an arbitrary set of objects could be described. We defined lower and upper approximations for sets of objects in an algebraic way using Boolean operations. Those new definitions were compared to the classical ones (which use an equivalence relation) and were shown to be more general in the sense that the classical definitions can be deduced from them if we put some restrictions on our model. We stressed that using binary codes allows us to quickly calculate the approximations of a set in accordance with the new definitions and generate logic rules based on the approximations.

In this paper we generalize the above approach. Now we do not suppose that object features must be Boolean. We consider arbitrary discrete values of information features. It turns out that in this general case the main results of the suggested algebraic approach remain true.

2 An algebraic definition of rough approximations

The classical rough sets theory deals with the "indiscernibility" concept. This concept considers some elements of a set, which cannot be "discerned" in terms of some relation (it normally can be an equivalence or tolerance relation). We would like to consider the ambiguity concept in a different way. We suppose that the available information on the elements of a set is represented with the help of their "properties". Such an approach allows us to consider *any* relations defined on the set as *properties* of elements, pairs of elements, ordered sets of elements etc. On can apply various operations to these relations and, as a result, obtain new relations. In fact, any information on the elements of the set can be represented in the form of a relation. Sometimes one can precisely express a given relation in terms of the available relations but often it is not possible to do it and one has to consider approximations to the relation being described.

A new algebraic approach to defining rough set approximations has been developed by the authors in [7]. Suppose we are given a finite nonempty set of objects $U=\{a_1, a_2, \ldots, a_n\}$, called *universe*. Consider also a set of unary predicates (functions that take on their values from the set $\{0,1\}$) defined on U:

$$P_1(t), P_2 (t), \ldots, P_k(t), \tag{1}$$

which we call *coordinates*.

The predicates $P_1, P_2, \ldots, P_k$ can be interpreted as characteristic functions for some properties of objects of the universe. In this case an object a_i has the property P_j if and only if $P_j (a_i) = 1$.

Following the basic concepts of the rough set theory we should describe an arbitrary set $X \subseteq U$ in terms of the coordinates. Since there exists a one-to-one correspondence between all the predicates defined on U and all the subsets of U, instead of a set $X \subseteq U$ we can consider a predicate X(t) that equals 1 if and only if $t \in X$. Thus we should give a description of an arbitrary predicate X(t) in terms of the predicates $P_1, P_2, \ldots, P_k$.

Also in [7] we have introduced the concept of the *approximation language* consisting of unary predicates $P_1, P_2, \ldots, P_k$ and Boolean operations: *conjunction* (&), *disjunction* (V) and *negation* ($\neg$). It was stressed that in the general case the approximation language can include other types of predicates and operations.

We have considered the set Φ of all possible formulae constructed with the help of the Boolean operations *conjunction* (&), *disjunction* (V) and *negation* ($\neg$) applied to the predicates $P_1, P_2, \ldots, P_k$. For example: $(P_1 \& P_2 \& \neg P_3) V P_4$, $(P_1 V (P_2 \& \neg P_3)) \& P_4$, $\neg (P_1 \& P_2 \& \ldots \& P_4)$ etc. On calculating all the formulae belonging to Φ , we will obviously obtain a set of predicates, which we will denote Λ. We noted that different formulae (not necessarily equivalent ones) can correspond to the same predicate.

In accordance with the above approach, if the predicate X(t) belongs to Λ, it means that X(t) *can be expressed in terms of the coordinates* and the set X corresponding to the predicate X(t) can be called *crisp* with respect to the coordinates. If the predicate X(t) does not belong to Λ, this predicate cannot be expressed in terms of the coordinates and we should describe it *approximately*.

Also from the viewpoint of the suggested approach in [7] we have introduced definitions for upper and lower approximations of X(t) and the concepts of the exact lower and upper approximations:

1. If $A(t) \in \Lambda$ and $\forall t \in U\ A(t) \rightarrow X(t)$ then we say that A(t) is a *lower approximation* for X(t).
2. If $B(t) \in \Lambda$ and $\forall t \in U\ X(t) \rightarrow B(t)$ then we say that B(t) is an *upper approximation* for X(t).
3. If a predicate $I_*(t)$ is a lower approximation for the predicate X(t) and for any lower approximation A(t) of this predicate $\forall t \in U\ A(t) \rightarrow I_*(t)$ then we say that $I_*(t)$ is an *exact lower approximation* for X(t).
4. If a predicate $I^*(t)$ is an upper approximation for the predicate X(t) and for any upper approximation B(t) of this predicate $\forall t \in U\ I^*(t) \rightarrow B(t)$ then we say that $I^*(t)$ is an *exact upper approximation* for X(t).

We should stress that the characteristic functions $P_1, P_2, \ldots, P_k$, which were used for describing object properties in [7] could take on only two values: 1 and 0. Any object either had a given property or did not have it.

Let us now consider a more general case where characteristic functions describing object properties can take on values from any finite set. For example, for the function P_1 denoting the property "colour" the following can be said: $P_1(a_i)=1$ means that the object a_i is red, $P_1(a_i)=2$ means that the object is orange, $P_1(a_i)=3$ means that it is yellow etc. In this case such functions can be described with the help of finite predicates: $P_k^w(a_i)=1$ means that the value of the property P_k for the object a_i is w, otherwise this finite predicate equals zero. Thus these finite predicates implement the comparison operation that compares values of discrete characteristic functions. The following conditions are satisfied for such predicates:

1. For any object a_i the property P_k necessarily takes on at least one value w from the set of possible values: $w \in W$, $W=\{1,2,\ldots,m\}$. Thus the following identity is true:

$$P_k^1(a_i) \vee P_k^2(a_i) \vee \ldots \vee P_k^m(a_i) \equiv 1\,. \qquad (2)$$

2. For any object a_i a property P_k cannot take on two different values from the set of possible values. Thus the following identity is true:

$$P_k^w(a_i)\, \&\, P_k^r(a_i) \equiv 0 \quad (w \in W,\ r \in W,\ w \neq r). \qquad (3)$$

Taking into account the above considerations we can define the new approximation language as one consisting of the set of discrete characteristic functions and the set of operations including Boolean ones and the comparison operator that gives 1 if two discrete values are equal and 0 if they are different.

3 Properties of the approximations

In [7] we have proven some properties of upper and lower approximations in case the functions describing object features are binary. The same properties remain true also in case the characteristic functions take on values from finite sets:

1. The set of lower approximations is not empty for any X(t). This follows from the fact that the predicate $0 = P_k^w \& P_k^r, (w \neq r)$ is always a lower approximation for X(t).
2. The set of upper approximations is not empty for any X(t). This statement is true since the predicate $1 = P_k^1 \vee P_k^2 \vee ... \vee P_k^m$ is always an upper approximation for X(t).
3. For any predicate X(t) there cannot exist more than one exact lower approximation.
4. For any predicate X(t) there exists at least one exact lower approximation.
5. For any predicate X(t) there exists the only exact lower approximation.
6. For any predicate X(t) there cannot exist more than one exact upper approximation.
7. For any predicate X(t) there exists at least one exact upper approximation.
8. For any predicate X(t) there exists the only exact upper approximation. It follows from properties 6 and 7.

The proof of the properties 3 to 7 is the same as in [7].

4 Calculating the exact approximations with the help of discrete characteristic functions

Let us consider now a method for finding the exact upper and lower approximations in case object properties are described with the help of non-binary values. Verifying all possible predicate formulae to calculate exact approximations is a time consuming procedure. Nevertheless, there exists a way of obtaining the approximations for a predicate that allows us to quickly write down necessary formulae. Consider Table 1.

Table 1.

	a_1	a_2	…	a_n
P_1	δ_{11}	δ_{12}	…	δ_{1n}
P_2	δ_{21}	δ_{22}	…	δ_{2n}
…	…	…	…	…
P_k	δ_{k1}	δ_{k2}	…	δ_{kn}
X	λ_1	λ_2	…	λ_n

Here $\delta_{1j} \in \{0,1..m_1\}$, $\delta_{2j} \in \{0,1..m_2\}$,...$\delta_{kj} \in \{0,1..m_k\}$, $\lambda_j \in \{0,1\}$, if $\delta_{ij}=w$ then $P_i(a_j)=w$, if $\lambda_j = 1$ then $X(a_j) = 1$, if $\lambda_j = 0$ then $X(a_j) = 0$.

Suppose that the predicate X should be described in terms of the coordinates P_1, P_2, …, P_k, which are now arbitrary discrete characteristic functions. Let us find the exact upper approximation for X. For this purpose consider the columns of the table that contain 1 for the predicate X and write down the corresponding predicate disjunctive normal form. A simple example is given below.

Suppose that the characteristic functions P_1, P_2, P_3 describing features of the objects a_1, a_2,..., a_5 can take on values from the set {0,1,2} as in Table 2.

Table 2.

	a_1	a_2	a_3	a_4	a_5
P_1	1	0	2	0	0
P_2	0	2	1	0	2
P_3	0	2	0	1	2
X	0	1	0	1	0

For this example we will get the following formula:

$$I^* = (P_1^0 \,\&\, P_2^2 \,\&\, P_3^2) \vee (P_1^0 \,\&\, P_2^0 \,\&\, P_3^1) \tag{4}$$

Consider now the columns that contain 0 for the predicate X and write down the corresponding conjunctive normal form. For this example:

$$I_* = (\neg P_1^1 \vee \neg P_2^0 \vee \neg P_3^0) \,\&\, (\neg P_1^2 \vee \neg P_2^1 \vee \neg P_3^0) \,\&\, (\neg P_1^0 \vee \neg P_2^2 \vee \neg P_3^2) \tag{5}$$

Formulae (4) and (5) produce results in Table 3.

Table 3.

	a_1	a_2	a_3	a_4	a_5
P_1	1	0	2	0	0
P_2	0	2	1	0	2
P_3	0	2	0	1	2
X	0	1	0	1	0
I^*	0	1	0	1	1
I_*	0	0	0	1	0

In the general case the predicates I^* and I_* can be represented as follows:

$$I_* = (\lambda_1 \& P_1^{\delta_{11}} \& P_2^{\delta_{21}} \& \ldots \& P_k^{\delta_{k1}}) \vee (\lambda_2 \& P_1^{\delta_{12}} \& P_2^{\delta_{22}} \& \ldots \& P_k^{\delta_{k2}}) \vee \ldots \vee (\lambda_n \& P_1^{\delta_{1n}} \& P_2^{\delta_{2n}} \& \ldots \& P_k^{\delta_{nn}}) \quad (6)$$

$$I_* = (\lambda_1 \vee \neg P_1^{\delta_{11}} \vee \neg P_2^{\delta_{21}} \vee \ldots \vee \neg P_k^{\delta_{k1}}) \& (\lambda_2 \vee \neg P_1^{\delta_{12}} \vee \neg P_2^{\delta_{22}} \vee \ldots \vee \neg P_k^{\delta_{k2}}) \& \ldots \& (\lambda_n \vee \neg P_1^{\delta_{1n}} \vee \neg P_2^{\delta_{2n}} \vee \ldots \vee \neg P_k^{\delta_{nn}}) \quad (7)$$

where $P_k^{\delta_{ij}} = 1$ if $P_k(a_i) = \delta_{ij}$, otherwise $P_k^{\delta_{ij}} = 0$, and $\neg P_k^{\delta_{ij}} = 0$ if $P_k(a_i) = \delta_{ij}$, otherwise $\neg P_k^{\alpha_n} = 1$ for any predicate P.

Let us show that the predicates I^* and I_* are the exact upper and lower approximations. It is obvious that the predicate I^* is an upper approximation for the predicate X in terms of definition 2. If one removes from formula (6) any conjunction where $\lambda_i=1$, the resulting formula will not be an upper approximation, as the predicate I^* will have 0 in a column where X has 1. (We suppose here that in case there are several conjunctions identical to the one removed all of them should be removed). For example if one removes the conjunction $(P_1^0 \& P_2^2 \& P_3^2)$ from formula (4), then $I^*(a_2)$ becomes 0 whereas $X(a_2) = 1$. It means that the approximation I^* cannot be improved and, therefore, I^* is the exact upper approximation. The predicate I_* is obviously a lower approximation for X in terms of definition 1. If one removes from formula (7) any disjunction where $\lambda_i=0$, the resulting formula will not be a lower approximation as the predicate I_* will have 1 in a column where X has 0. (We suppose that if there are several disjunctions identical to the one removed all of them should be removed). For example if one removes the disjunction $(\neg P_1^1 \vee \neg P_2^0 \vee \neg P_3^0)$ from formula (5), then $I^*(a_1) = 1$ whereas $X(a_1) = 0$. It means that the approximation I_* cannot be improved and, therefore, I_* is the exact lower approximation.

5 Approximation-based logic rules

Consider an example of generating logic rules with the help of formulae (4) and (5). Following traditional rough set concepts we can say that rules based on the exact upper approximation *may* exist in the data set, and rules based on the exact lower approximation *must* exist in the data. Transform the expression on the right side of formula (5) to get:

$$I_* = (\neg P_1^1 \vee \neg P_2^0 \vee \neg P_3^0) \& (\neg P_1^2 \vee \neg P_2^1 \vee \neg P_3^0) \& (\neg P_1^0 \vee \neg P_2^2 \vee \neg P_3^2)$$

We can now formulate the following *exact* rules:

A. An element belongs to the set X if

a) property P_1 is not equal to 1 or P_2 and P_3 are not equal to 0

AND

b) property P_1 is not equal to 2 or property P_2 is not equal to 1 or property P_3 is not equal to 0

AND

c) property P_1 is not equal to 0 or property P_2 is not equal to 2 or property P_3 is not equal to 2.

This rule says that if all of the conditions a), b) and c) hold, an element belongs to the set X.

Since each of the functions P_k (in this case P^1, P^2, P^3) takes on at least one value from the set of possible values, $\neg P_k^i = P_k^1 \vee P_k^2 \vee \ldots \vee P_k^{i-1} \vee P_k^{i+1} \vee \ldots \vee P_k^n$ and we can use this property to transform the above equation. After some simple transformations we can obtain the following formula:

$$I_* = P_3^1 \vee (P_1^1 \,\&\, P_3^2) \vee (P_1^1 \,\&\, \neg P_2^0) \vee (P_1^2 \,\&\, P_3^2) \vee (P_1^2 \,\&\, \neg P_2^1) \vee (P_1^0 \,\&\, \neg P_2^2) \vee (P_1^0 \,\&\, P_3^0)$$
$$\vee (P_3^2 \,\&\, \neg P_2^2) \vee (P_2^2 \,\&\, P_3^0).$$

We can avoid using the negation operation and deduce the following exact rules from the above formula:

B. An element belongs to the set X if

a) property P_3 is equal to 1

OR

b) property P_1 is equal to 1 and P_3 is equal to 2

OR

c) property P_1 is equal to 1 and P_2 is equal to 1 or 2

OR

d) property P_1 is equal to 2 and P_3 is equal to 2

OR

e) property P_1 is equal to 2 and P_2 is equal to 0 or 2

OR

f) property P_1 is equal to 0 and P_2 is equal to 0 or 1

OR

g) property P_1 is equal to 0 and P_3 is equal to 0

OR

h) property P_2 is equal to 0 or 1 and P_3 is equal to 2

OR

i) property P_2 is equal to 2 and P_3 is equal to 0

This rule says that if one of the conditions a), b), c), d), e), f), g), h) or i) holds, an element belongs to the set X.

Formula (4) for the exact upper approximation allows us to formulate the following *approximate* rules:

C. An element *may belong* to the set X *if*

P_1 is equal to 0 AND P_2 is equal to 2 *AND* property P_3 is equal to 3 for this element

D. An element *may belong* to the set X *if*

P_1 is equal to 3 AND P_2 is equal to 0 *AND* property P_3 is equal to 0 for this element

We have shown that logic rules that allow answering questions on whether or not an element belongs to a given set can be deduced from the available information on properties of elements by using comparison and Boolean operations. Since such calculations are very quick the resulting rules can be efficiently obtained even for great numbers of elements and characteristic functions.

6 Conclusions and discussion

In this paper a generalized algebraic approach to defining rough set approximations has been considered. The main idea of this approach is to describe a concept in terms of other concepts in an *algebraic way*, i.e. to find formulae that represent the most exact approximations of the concept under analysis. This paper represents a generalization of the approach suggested by the authors earlier where object features could take on only Boolean values. Now we have shown that the main idea of the algebraic approach holds also in the case where object properties can take on arbitrary values from finite sets. Logic rules generated with the help of the exact approximations can be used to classify new elements, i.e. to define whether or not an element belongs to a set, which is a classical problem of Data Mining. The proposed approach considers only comparison and Boolean operations, which makes the process of extracting logic rules very quick from the computational viewpoint. It should be noted that transformations of the finite predicate formulae for the exact approximations can be performed in a variety of ways and, therefore, it would be interesting to be able to find most simple and informative resulting rules.

References

[1] Pawlak Z.; (1982), *Rough sets*, International Journal of Computer and Information Sciences, 11, pp. 341-356.

[2] Pawlak, Z. (1995) *Vagueness and uncertainty: a rough set perspective*, Computational Intelligence, vol. 11 (issue 2), pp. 227-232.

[3] Pawlak, Z.; (1996), *Rough sets, rough relations and rough functions*, Fundamenta Informaticae, vol. 27, no. 2/3, pp. 103-108.

[4] Yao, Y.; Wong, S.; (1995), *Generalization of rough sets using relationships between attribute values*, in Proceedings of the 2nd Annual Joint Conference on Information Sciences, pp. 30-33.

[5] Polkowski, L.; Skowron, A.; Zytkow, J.; (1995), *Tolerance based rough sets*, in Lin, T; Wildberger, A.; (eds.), Soft computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Simulation councils, Inc., San Diego, pp. 55-58.

[6] Slowinski R.; Vanderpooten D.; (1997), *Similarity relation as a bases for rough approximations*, in Wong, P.P.; (ed.), Advances in Machine

Intelligence and Soft-Computing, vol. IV, Bookwrights, Raleigh, NC, pp. 17-33.

[7] D. Sitnikov, O. Ryabov. (2004), *An algebraic approach to defining rough set approximations and generating logic rules,* in Zanasi, A.; Ebecken, N.; Brebbia, C.; (eds), Data Mining V, Malaga, Spain, pp. 179-188.