Dynamic Knowledge Update Using Three-Way Decisions in Dominance-Based Rough Sets Approach While the Object Set Varies

Lei Wang1,2,*, Min Li1,2, Jun Ye1,2, Xiang Yu1,2, Ziqi Wang3, Shaobo Deng1,2
1 College of Information Engineering, Nanchang Institute of Technology, Nanchang, Jiangxi, 330099, China
2 Jiangxi Provincial Key Laboratory of Water Information Cooperative Sensing and Intelligent Processing, Nanchang, Jiangxi, 330099, China
3 Shenzhen HeXunHuaGu Information Technology Co., Ltd., Guangzhou Branch, Guangzhou, Guangdong, 510620, China

Abstract
Dominance-based rough set approach is the extension of classical Pawlak rough set theories and methodologies, in which the information with preference-ordered relation on the domain of attribute value is fully considered. In the dominance-based information system, upper and lower approximations will be changed while the object set varies over time and the approximations need to be updated correspondingly for their variations result in changes of knowledge and rules. Considering that three-way decisions is a special class of general and effective human ways of problem solving and information processing, a new incremental maintenance mechanism using three-way decisions is proposed in this paper, namely, the universe is divided into three pair-wise disjoint subsets firstly, then appropriate strategies are developed and acted on each subsets. Furthermore, the corresponding methods for updating the approximations of upward unions and downward unions of decision classes are analyzed systematically under the variations of object set in the dominance-based information system from the perspective of three-way decisions. Moreover, considering vector representation and calculation is intuitive and concise, two incremental update algorithms of approximations are suggested and implemented in the MATLAB platform. Finally, some tests on data sets from UCI (UC Irvine Machine Learning Repository) are undertaken to verify the effectiveness of the proposed methods. Compared with the non-incremental updating methods, the proposed incremental updating method with three-way decisions generally exhibits a better performance.

1. INTRODUCTION
Rough set theory (RST) provides a powerful mathematical tool for modeling and processing problems with incomplete, imprecise, uncertain and vague information [1] and it is one of the three primary models of Granular Computing [2] (GrC). The recent two decades have witnessed the booming interest and growing development in research of RST and its applications. As a kind of information processing tools, RST has been widely applied to the fields of artificial intelligence and cognitive sciences, such as pattern recognition [3], knowledge discovery [4,5], decision making [6,7], inductive reasoning [8] and machine learning [9]. In multiple-criteria decision making problems, attributes with preference-order domain (Criteria) constitute an important kind of attribute and should be brought to our great attention. However, only attribute values themselves, by which to tell one object from another, are considered and the preference-order relations among criteria values domain are not considered in RST. On this account, the information with preference-order attribute values domain cannot be processed by RST. Greco et al. [10–12] proposed the Dominance-based Rough Set Approach (DRSA) to address this issue, in which the innovation mainly lies in the substitution of the indiscernibility relation by dominance relation. Comparing with the decision rules derived from RST, those derived from DRSA are easy to understand as well as consistent with the practical situation due to the fact that the ordering properties of attributes are taken into account in DRSA. In real-life application, information system (IS) always varies with time for the generation and the collection of data are dynamic, called dynamic IS, it can cause the corresponding changes of the approximations of a concept from which the knowledge are acquired [13]. So updates of the approximations are of great importance for knowledge maintenance and other related tasks. Incremental update scheme is an effective and efficient technique for knowledge update, which enables to acquire new knowledge on the basis of prior knowledge without recalculation from scratch under variation of IS. A great deal of researches have been done in the area of incremental knowledge maintenance under rough sets methodologies. In general, the researches on incremental knowledge update can be fallen into four categories owing to the fact that ISs evolve over time with four levels of variational situations, i.e., (1) incremental knowledge update while the object set varies [14–17]; (2) incremental knowledge update while the attribute set varies [18–23]; (3) incremental knowledge update while the attribute values vary [24–27]; (4) incremental knowledge update under the

*Corresponding author. Email: ezhoulei@163.com

© 2019 The Authors. Published by Atlantis Press SARL.
This is an open access article distributed under the CC BY-NC 4.0 license (http://creativecommons.org/licenses/by-nc/4.0/).
simultaneous variations of object set and attribute set (or object set and attribute value, attribute set and attribute value) [28–32]. All these researches will help decision makers update knowledge from different perspectives in different kinds of ISs. Some achievements based on rough set methodologies under the changed IS are as follows:

Li et al. presented a new approach for incrementally updating approximations of a concept under the characteristic relation-based rough sets when the attribute set varies over time for dynamic attribute generalization, which can deal with the case of adding and removing some attributes simultaneously in the IS [14]. Huang et al. proposed mechanisms for updating approximations in probabilistic set-valued IS with the variation of attributes [31]. Zhang et al. developed the change of approximations in neighborhood decision systems when the object set evolves over time, and proposed two fast incremental algorithms for updating approximations when multiple objects enter into or get out of the neighborhood decision table [15]. Considering dynamic updating of attribute reduction for data vary with time, Jing et al. examined an incremental attribute reduction algorithm with a multi-granulation view to maintenance reduct of large-scale data sets dynamically [33,34]. Liu et al. proposed some relevant strategies and algorithms for incremental learning knowledge in probabilistic rough sets when attributes evolve with time [22]. Zhang et al. suggested an incremental method for updating rough approximations in interval-valued ISs while attributes set evolves [18]. Luo et al. developed an incremental learning technique for hierarchical multi-criteria classification while attribute values vary across different levels of granulations [25]. He further analyzed the updating mechanisms of approximations under the variation of the object set and criteria values respectively in set-valued-ordered IS and set-valued decision system [35]. Zhang et al. examined the updating approach of approximations of the concept in set-valued IS by using matrix under the variation of attribute set [36]. Wang et al. proposed an incremental matrix method for updating approximations under the variable precision rough set model at insertion or deletion of a single object [37]. Huang et al. introduced an incremental mechanism for updating rough fuzzy approximations with simultaneous variation of objects and attributes set by using matrix operator [31].

The theory of three-way decisions was introduced by Yao in order to model a particular class of human heuristic ways of problem solving and information processing [38,39] and its essential ideas are widely applied in many fields and disciplines. As for the application of three-way decisions in dynamic knowledge update field, Yu et al. proposed a new tree-based incremental overlapping clustering method using the three-way decision theory in order to efficiently update the clustering when the data changes [40]. Yang et al. proposed a unified model of sequential three-way decisions and multi-level incremental processing for complex problem solving [28,41], then they suggested an unified dynamic framework of decision-theoretic rough sets for incrementally updating three-way probabilistic regions [42].

In this paper, inspired by the trisecting-and-acting model of three-way decisions proposed by Yao, the essential ideas of the trisecting-and-acting model, namely, tri-partition and action are applied to analyze the incremental update mechanisms of approximations with the variation of objects set in dynamic dominance-based IS. Subsequently, the updates of the approximations is realized by using column vector as representation tool as well as computational tool, considering that the column vector is concise and intuitive in representation and operation. Our research motivation is to investigate incremental update of approximations from a new view and to exploit its feasibility and efficiency.

The remainder of the paper is organized as follows: In Section 2, some basic concepts of DRST as well as relevant operations of Boolean column vector are introduced. Under the variation of the object set, the re-judgement of existing relationship between sub-sets is examined in Section 3, the analyses on incremental update of approximations with three-way decisions and its corresponding illustrated example are presented in Sections 4 and 5, respectively. In Section 6, we propose two incremental algorithms using vector operations for computing approximations based on the preceding analyses. Performance evaluations are illustrated in Section 7. The paper ends with conclusions and further research topics in Section 8.

There are two contributions from our research. One is to provide a three-way decisions perspective on incremental update of approximations in DRSA. The other is an application of Boolean column vector inner product in judging the relationship (inclusion or intersection) between two sets, by which some objects need to be merged to the prior approximations and some other objects need to be removed from the prior approximations during the incremental update process of approximations.

2. PRELIMINARIES

2.1. Symbol Notation

For the sake of convenience, the symbols used in this paper and their corresponding meanings are listed in Table 1.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(X)</td>
<td>The column vector of subset (X)</td>
</tr>
<tr>
<td>(D_P, D_{P}^T)</td>
<td>Dominance relation matrix and its transposed matrix</td>
</tr>
<tr>
<td>(P\left(C_{i_j}^{p}\right), P\left(C_{i_j}^{p, g}\right))</td>
<td>The column vector of (P\left(C_{i_j}^{p}\right)) before and after insertion or removal of object</td>
</tr>
<tr>
<td>(\overline{P}\left(C_{i_j}^{p}\right), \overline{P}\left(C_{i_j}^{p, g}\right))</td>
<td>The column vector of (\overline{P}\left(C_{i_j}^{p}\right)) before and after insertion or removal of object</td>
</tr>
<tr>
<td>(\overline{P}\left(C_{i_j}^{d}\right), \overline{P}\left(C_{i_j}^{d, e}\right))</td>
<td>The column vector of (\overline{P}\left(C_{i_j}^{d}\right)) before and after insertion or removal of object</td>
</tr>
<tr>
<td>(\text{ones}(n, 1), \text{ones}(1, n))</td>
<td>The (n)-dimensional row and column vector with all the components equal to 1</td>
</tr>
<tr>
<td>(</td>
<td>·</td>
</tr>
<tr>
<td>([A:B])</td>
<td>A column vector which is constructed though concatenation of column vector (A) and (B)</td>
</tr>
<tr>
<td>(A[m:n])</td>
<td>Column sub-vector which consist of rows (m-n) of column vector (A)</td>
</tr>
</tbody>
</table>

Table 1 | Description of symbols.
2.2. The Basic Concepts of Dominance-Based Rough Set Approach

Some basic concepts of RST and DRSA are briefly reviewed in this subsection [1,10–12].

An IS is a quadruple IS = (U, C ∪ {d}, V, f), where U = {u1, u2, ..., un} is a non-empty finite set of objects, called the universe. C is a non-empty finite set of condition attributes and d is a decision attribute with C ∩ {d} = φ. V = Vc ∪ Vd, where V is the domain of all attributes, Vc is the domain of all condition attributes and Vd is the domain of the decision attribute d; f is a mapping from U × (C ∪ {d}) to V.

In an IS, OB ∈ 2U ∧ OB ≠ φ and P ∈ 2C ∧ P ≠ φ. Where 2U and 2C is the power set of U and C respectively. If there is a preference (decreasing or increasing) relation on two elements x and y of OB for all a ∈ P, denoted by xdpy. Dp is a dominance relation on universe U with respect to the attribute set P.

\[ D_p = \{(x, y) \in U \times U | f(x, a) \geq f(y, a), \forall a \in P\}. \] (1)

In Ref. [10], the granules of knowledge of element x used in DRSA for approximation of the unions Cl↑p and Cl↓p, namely, the P-dominating set and P-dominated set of element x were defined, respectively, as follows:

\[ D_p^+(x) = \{y \in U | xdpy\}. \] (2)

\[ D_p^-(x) = \{y \in U | xdpy\}. \] (3)

The equivalence classes partition of universe U by the decision attribute d is denoted as U/IND(d), called set of decision classes. Let U/IND(d) = {Cln, n ∈ T}, T = {1, ..., m}, m denotes the number of decision classes. ∀r, s ∈ T such that r > s, it means the objects from Clr are preferred to the objects from Cls. In DRSA, an upward union and a downward union of decision classes, denoted as Cl↑p = u′>n Cl↑p and Cl↓p = u′<n Cl↓p respectively, are no other than the concepts to be approximated and the definitions of approximation were given in Ref. [10].

Definition 1. Assume that IS is an information system, P ⊆ C, n ∈ T. The definitions of lower and upper approximations of Cl↑p and Cl↓p are respectively as follows:

\[ P(Cl↑p) = \{x \in U | D_p^+(x) \subseteq Cl↑p\}. \] (4)

\[ P(Cl↓p) = \{x \in U | D_p^+(x) \cap Cl↓p \neq \phi\}. \] (5)

\[ P(Cl↑p) = \{x \in U | D_p^+(x) \subseteq Cl↑p\}. \] (6)

\[ P(Cl↓p) = \{x \in U | D_p^+(x) \cap Cl↓p \neq \phi\}. \] (7)

Three-way decisions, proposed by Yao, is a general and effective human heuristic way to problem solving and information processing, their aim is to make fast, low cost and/or high benefit decisions in solving problems with uncertainty and imprecision [38,39,43,44]. The essential ideas of three-way decisions are described in term of a ternary classification according to evaluations of a set of criteria [38]. The two basic tasks of three-way decisions, which gives rise to a trisecting-and-acting model of three-way decisions, are trisecting and acting. Trisecting, namely, tri-partition is to divide a whole into three pair-wise disjoint parts or regions and acting is to develop an appropriate strategy on each part or region [39]. In many real applications, trisecting-and-acting model can turn complexity into simplicity [39] due to its firm cognitive basis and appropriate strategies and it would be a simple, general and flexible model.

2.4. The Boolean Column Vector

The basic concepts of Boolean column vector as well as the operations on Boolean Column vector (including the operations we proposed) are introduced in this subsection [45–47]. Definition 2 is quoted from Ref. [45], Definition 3 is quoted from Ref. [46] and Definition 7 is quoted from Ref. [47]. The vector inner product will be used in analyses on incremental updates of approximations of unions of decision classes, the other operations of vector will be used in the incremental update matrix algorithms of the unions’ approximations at insertion or deletion of an object in Section 6.

Definition 2. Assume that U = {u1, u2, ..., un} is a non-empty finite set of objects, called the universe, and X ⊆ U. Then the column vector representation of subset X is as follows:

\[ X_{n \times 1} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_{1} \\ x_{2} \\ \vdots \\ x_{n} \end{bmatrix}^T. \] (8)

where \( x_i = \begin{cases} 1 & u_i \in X \\ 0 & u_i \notin X, \end{cases} \)

In Formula (8), the location of object \( u_i \) in the universe U is i, denoted as Loc(\( u_i \), U) = i.

Example 1. Assume that U = {u1, u2, u3, u4, u5, u6, u7, u8} and A = {u1, u3, u5, u7, u8}, B = {u2, u3, u4, u5, u7}, then

\[ A = [a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8]^T = [1, 0, 1, 0, 1, 0, 1, 1]^T. \]

\[ B = [b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8]^T = [0, 1, 1, 1, 0, 1, 0, 0]^T. \]

Example 2. Suppose U = {u2, u1, u7, u4, u8, u6, u3, u5}, then Loc(\( u_7 \), U) = 3, Loc(\( u_8 \), U) = 5, Loc(\( u_3 \), U) = 8.

2.4.1. Operations on Boolean column vector

Assume set X, Y and Z are subsets of universe U and their corresponding n-dimensional Boolean column vectors are \( X, Y, Z \) respectively and \( X = \begin{bmatrix} x_1, x_2, ..., x_n \end{bmatrix}^T, Y = \begin{bmatrix} y_1, y_2, ..., y_n \end{bmatrix}^T, Z = \begin{bmatrix} z_1, z_2, ..., z_n \end{bmatrix}^T \).
Definition 3. Let
\[ [X, Y] = X^T \cdot Y = \sum_{i=1}^{m} x_i \cdot y_i. \]  
(9)

\([X, Y]\) is called inner product of column vector.

The following formulas are easy to gained from Definition 3.

\[ [X, Y] = 0 \iff X \cap Y = \emptyset. \]  
(10)

\[ [X, Y] = |X| \iff X \subseteq Y. \]  
(11)

\[ [X, Y] = |Y| \iff Y \subseteq X. \]  
(12)

Vector Inner product can be used to judge the intersection or inclusion between two sets.

Example 3. (Continuation of Example 1)

\[ [A, B] = \sum_{i=1}^{8} a_i \cdot b_i = 3 \text{ and } A \cap B \neq \emptyset. \]

Taking into account the characteristics of Boolean column vector, we defined their operations corresponding to the intersection, union and difference of two sets respectively as follows:

Definition 4. Let
\[ Z = X \cdot * Y. \]  
(13)

where \( z_i = \begin{cases} 1 & x_i = 1 \land y_i = 1 \\ 0 & \text{others} \end{cases} \) \( i = 1, 2, \ldots, n \)

\(*\) is called operator of column vector multiplication. It can be used to calculate the intersection of two sets.

Example 4. (Continuation of Example 1)

Let \( D = A \cdot * B = [0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0]. \) So \( D = \{u_3, u_5, u_7\} \) and set \( D \) is the intersection of set \( A \) and set \( B \).

Definition 5. Let
\[ Z = X + Y. \]  
(14)

where \( z_i = \begin{cases} 0 & x_i = 0 \land y_i = 0 \\ 1 & \text{others} \end{cases} \) \( i = 1, 2, \ldots, n \)

\( Z \) is called summation of \( X \) and \( Y \). It can be used to calculate the union of two sets.

Example 5. (Continuation of Example 1)

Let \( D = A + B = [1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1]. \) So \( D = \{u_1, u_2, u_3, u_4, u_5, u_7, u_8\} \) and set \( D \) is the union of set \( A \) and set \( B \).

Definition 6. Let
\[ Z = X - Y. \]  
(15)

where \( z_i = \begin{cases} 1 & x_i = 1 \land y_i = 0 \\ 0 & \text{others} \end{cases} \) \( i = 1, 2, \ldots, n \)

\( Z \) is called difference of \( X \) and \( Y \). It can be applied to calculate the difference of two sets.

Example 6. (Continuation of Example 1)

Let \( D = A - B = [1, 0, 0, 0, 0, 0, 0, 0, 1, 0]. \) So \( D = \{u_1, u_3\} \) and set \( D \) is the subtraction of set \( A \) and set \( B \).

Definition 7. Assume that \( X \) and \( Y \) are two non-empty sets, the inclusion degree of \( X \) in \( Y \) is denote as \( C(X, Y) \).
\[ C(X, Y) = \frac{|X \cap Y|}{|X|} = \frac{|X, Y|}{|X, X|}. \]  
(16)

It is obviously that \( 0 \leq C \leq 1 \).

Inclusion degree can be used to judge the relationship of two sets. The following formulas are easy to gained from Definitions 7 and 3.

\[ C(X, Y) = 0 \iff [X, Y] = 0 \iff X \cap Y = \emptyset. \]  
(17)

\[ C(X, Y) = 1 \iff [X, Y] = |X| \iff X \subseteq Y. \]  
(18)

\[ C(Y, X) = 1 \iff [X, Y] = |Y| \iff Y \subseteq X. \]  
(19)

Example 7. (Continuation of Example 1)

\[ C(A, B) = \frac{|A \cap B|}{|A|} = \frac{|A, B|}{|A, A|} = \frac{3}{5} = 0.6 \text{ and } A \cap B \neq \emptyset. \]

3. THE RE-JUDGEMENT OF EXISTING RELATIONSHIP BETWEEN SUBSETS UNDER VARIATIONS OF OBJECTS

According to the definitions of approximations, the judgement of inclusion relationship or intersection relationship between two subsets of universe is a prerequisite for calculating approximations in RST. However, the pre-existing relationship between two subsets will change after new objects are inserted into the universe or the original objects are removed from the universe, so the relationship between two subsets remains to be re-judged. In this paper, Boolean column vector is employed to denote the subsets of the universe, and the problem that judge one object belongs to approximations of upward (downward) unions or not can be transformed into the operations of Boolean column vectors and the subsequent simple numerical comparison. In short, the vector approach of dynamic maintenance and update of approximations at variation of objects is that of incremental maintenance and update of approximations via operations on column vectors. Therefore, the research of this section is the foundation of follow-up analyses on incremental update of approximations from perspective of trisecting and acting.

Suppose that set \( A \) and set \( B \) are two non-empty subsets of the universe \( U \). \( A' \) and \( B' \) denote updated \( A \) and updated \( B \), respectively, after \( x^+ \) is inserted into \( U \) or \( x^- \) is removed from \( U \) respectively.

The Lemma 1 and Lemma 2 can be applied to re-judge the relationship between set \( A \) and set \( B \) after object \( x^+ \) is inserted into universe \( U \).

Lemma 1. Assume that \( A \not\subseteq B \) (i.e., \( 0 < |A, B| < \min \left( |A|, |B| \right) \)), then
\[ A \cap B \neq \emptyset \Rightarrow A' \cap B' \neq \emptyset \land A' \not\subseteq B'. \]  
(20)

\[ A \cap B = \emptyset \land x^+ \in A' \cap B' \Rightarrow A' \cap B' \neq \emptyset. \]  
(21)

The equivalent expressions of column vector are as follows:

\[ |A, B| \neq 0 \Rightarrow 0 < |A', B'| < \min \left( |A'|, |B'| \right). \]  
(22)

\[ |A, B| = 0 \land x^+ \in A' \cap B' \Rightarrow |A', B'| = 1. \]  
(23)
Lemma 2. Assume that $A \subseteq B$ (i.e., $[A, B] = |A|$), the inclusion relationship between $A$ and $B$ will be changed after $x^+$ is inserted into $U$ if and only if $x^+ \in A^\prime \land x^+ \notin B^\prime$.

$$A \subseteq B \land x^+ \in A^\prime \land x^+ \notin B^\prime \Rightarrow A^\prime \notin B^\prime.$$ (24)

Its equivalent expression of column vector’s inner product is as follows:

$$[A, B] = |A| \land x^+ \in A^\prime \land x^+ \notin B^\prime \Rightarrow [A^\prime, B^\prime] \neq |A'|.$$ (25)

Proof. The above conclusions can be driven easily from connotation of the inclusion relation between two sets.

Lemma 2 can be demonstrated by the Venn diagram in the Fig. 1.

Similarly, the Lemma 3 and Lemma 4 can be applied to re-judge the relationship between set $A$ and set $B$ after object $x^-$ is removed from universe $U$.

Lemma 3. Assume that $A \not\subseteq B$, then

$$x^- \in A \cap B \Rightarrow A^\prime \not\subseteq B.$$ (26)

$$A \cap B = \phi \lor A \cap B = \{x^\prime\} \Rightarrow A^\prime \cap B^\prime = \phi.$$ (27)

The equivalent expressions of column vector are as follows:

$$[A, B] \neq 0 \land x^- \notin A \cap B \Rightarrow [A^\prime, B^\prime] \neq 0.$$ (28)

$$[A, B] = 0 \lor A \cap B = \{x^\prime\} \Rightarrow [A^\prime, B^\prime] = 0.$$ (29)

Lemma 4. Assume that $A \subseteq B$, then

$$A \subseteq B \Rightarrow A^\prime \subseteq B^\prime.$$ (30)

Its equivalent expression of column vector’s inner product is as follows:

$$[A, B] = |A| \Rightarrow [A^\prime, B^\prime] = |A'|.$$ (31)

Proof. The above conclusions can be driven easily from connotation of the inclusion relation. As shown in the Venn diagram in Fig. 2, three cases are considered, i.e., $x^- \in A$, $x^- \in A^C \cap B$ and $x^- \in A^C \cap B^\prime$.

Lemma 4 demonstrates that the removal of single object from the universe will not alter the existing inclusion relationship between two subsets.

4. THE ANALYSES ON INCREMENTAL UPDATE OF APPROXIMATIONS WITH THREE-WAY DECISIONS

In this section, a trisecting-and-acting model is presented to update approximations in DRSA incrementally and then the methods of tri-partition of universe at insertion or removal of one object are discussed.

4.1. A Trisecting-and-Acting Model for Incremental Update of Approximations in DRSA

The incremental update method of approximations is just the method in which updated approximations can be acquired on the basis of the approximations prior to update by some certain operations and it is not necessary to recompute from scratch. As for the incremental update using three-way decisions, the certain operations refer to trisecting and acting.

We take the update of $P(C_{x^+})$ under the insertion of one object $x^+$ as an example to demonstrate the divide-and-conquer method in trisecting-and-acting model for incremental update of approximations.

**Figure 1** | The change of inclusion relation between two subsets after insertion of object $x^+$.

**Figure 2** | The inclusion relation between two subsets remains unchanged after deletion of object $x^-$. 
Above all, the objects in the universe \( U' \) can be partitioned into three pair-wise disjoint parts, as shown in Fig. 3.

The first part consists of all the objects which do not belong to the original approximation and belong to the updated approximation simultaneously, denoted as \( N \), shown in Formula (32).

\[
N = \{ x \in U | x \not\in P(Cl_n^0) \land x \in P(Cl_n^{\geq}) \}. \tag{32}
\]

The second part consists of all the objects which belong to the approximation before update and no longer belong to the approximation after update, denoted as \( M \), shown in Formula (33).

\[
M = \{ x \in U | x \in P(Cl_n^0) \land x \not\in P(Cl_n^{\geq}) \}. \tag{33}
\]

The rest objects of the universe constitute the third part, i.e., the third part is composed of all the objects except for the first part and the second part, denoted as \( O \) and shown in Formula (34).

\[
O = \left\{ x \mid x \not\in P(Cl_n^0) \land x \not\in P(Cl_n^{\geq}) \lor x \in P(Cl_n^0) \land x \in P(Cl_n^{\geq}) \right\} \tag{34}
\]

It’s worth noting that the inserted object \( x^+ \) is a special object. \( x^+ \) may be belongs to either the set \( N \) or the set \( O \) and we need to determine which set \( x^+ \) belong to.

Secondly, the different strategy will be developed and adopted in the different part after trisection. It is obviously that the strategies should be developed and adopted only in the set \( N \) and the set \( M \) after tri-partition of universe.

So the incremental update of \( P(Cl_n^0) \) can be finished by the following three steps:

1. Computes the set \( M \) and the set \( N \).
2. Subtracts the set \( M \) from \( P(Cl_n^0) \), assume that the intermediate result is \( P' = P(Cl_n') \).
3. Computes the union set of \( P' \) and the set \( N \), so the updated approximation \( P(Cl_n^{\geq}) \) is gained.

Thus, the incremental update of \( P(Cl_n^0) \) is more appropriately formulated as the following formula:

\[
P(Cl_n^{\geq}) = P(Cl_n^0) - M \cup N. \tag{35}
\]

The incremental update of \( P(Cl_n^0) \) follows Formula (35) after tri-partition of universe. And the incremental update of \( P(Cl_n^0) \), \( P(Cl_n^n) \) and \( P(Cl_n^{\geq}) \) are similar with that of \( P(Cl_n^0) \).

As for the case of object’s deletion, the process of incremental update of approximations is similar with that of object’s insertion, so it is not discussed again.

In the following subsections, under the unified frame of trisecting-and-acting model, only methods of tri-partition are discussed for updating on dynamic environment of insertion and removal of one object, respectively. What’s more, it can be simplified as a series of calculational formulas of the set \( M \) and \( N \). The inner product of Boolean column vectors is employed to judge the relationship between two sets in the proof of the theorems.

### 4.2. The Update of Approximations Using Trisecting-and-Acting Model at Insertion of Object

Supposing \( x^+ \) is the inserted object and \( Loc(x^+, U) = i \).

#### 4.2.1. The update of unions of decision classes

Supposing that \( Cl_n^i \) is the decision class to which inserted object \( x^+ \) belongs.

1. The update of upward unions of decision classes

   The updated upper unions of decision classes is as following formula:

   \[
   Cl_n^{\geq} = \begin{cases} 
   Cl_n^i \cup \{ x^+ \} & n' \geq n \\
   Cl_n^i & n' < n 
   \end{cases} \tag{36}
   \]
The corresponding column vector of the updated upward unions of decision classes can be denoted as
\[
C^u_n = \begin{cases} 
C_n^u (1 : i - 1); 1; C_n^u (i + 1 : m) & n' \geq n \\
C_n^u (1 : i - 1); 0; C_n^u (i + 1 : m) & n' < n
\end{cases}.
\] (37)

2. The update of downward unions of decision classes
The updated downward unions of decision classes is as following formula:
\[
C^l_n \leq \begin{cases} 
C_n^l \cup \{x^+\} & n' \leq n \\
C_n^l & n' > n
\end{cases}.
\] (38)

The corresponding column vector of the updated upward union set can be denoted as
\[
C^l_n \leq \begin{cases} 
C_n^l (1 : i - 1); 1; C_n^l (i + 1 : m) & n' \leq n \\
C_n^l (1 : i - 1); 0; C_n^l (i + 1 : m) & n' > n
\end{cases}.
\] (39)

It is obviously that the dimension of updated column vector will increase to \(n + 1\) after removal of \(x^+\).

4.2.2. Trisection of universe at insertion of object
The calculations of the set \(N\) and \(M\) are discussed in Theorems 1–4 respectively. The computational results lead to tri-partition of universe naturally.

- The computational formula of the set \(M\) and \(N\) for incremental update of \(\overline{P}(C_n^l)\)

**Theorem 1.**

\[
M = \begin{cases} 
\phi & n' \geq n \\
\{u|u \in D^+_n (x^+) \land u \in P(C_n^l)\} & n' < n
\end{cases}.
\] (40)

\[
N = \begin{cases} 
x^+ & n' \geq n \land D^+_n (x^+) \subseteq C^l_n \\
\phi & n' < n
\end{cases}.
\] (41)

**Proof.**

Case 1: If \(D^+_n (x^+) \subseteq C^l_n\), i.e.,
\[\{C_n^l \cup D^+_n (x^+)\} = \{D^+_n (x^+)\}\], we have \(x^+ \in P(C_n^l)\).

Case 2: Assume that \(v \notin P(C_n^l)\), then \(D^+_n (v) \subseteq C^l_n\), we have \([C_n^l \cup D^+_n (v)] = \{D^+_n (v)\}\), and \([C_n^l \cup D^+_n (v)] = \{D^+_n (v)\}\) still holds after \(x^+\) is inserted into \(U\) (Lemma 1).

To sum up, \(N = \{x^+\}\) while \(D^+_n (x^+) \subseteq C^l_n\).

Case 3: Assume that \(u \notin P(C_n^l)\), then \(D^+_n (v) \subseteq C^l_n\), we have \([C_n^l \cup D^+_n (v)] = \{D^+_n (v)\}\), for \(x^+ \in C^l_n\), \([C_n^l \cup D^+_n (v)] = \{D^+_n (v)\}\) still holds after \(x^+\) is inserted into \(U\) (Lemma 2).

So, \(M = \phi\).

Case 1: For \(x^+ \notin C^l_n\) and \(x^+ \notin D^+_n (x^+)\), we have \([C_n^l \cup D^+_n (x^+)] = \{D^+_n (x^+)\}\), so \(x^+ \notin N\).

Case 2: Assume that \(v \notin P(C_n^l)\), then \(D^+_n (v) \subseteq C^l_n\), we have \([C_n^l \cup D^+_n (v)] \neq \{D^+_n (v)\}\), and \([C_n^l \cup D^+_n (v)] \neq |D^+_n (v)|\) still holds after \(x^+\) is inserted into \(U\) (Lemma 1).

So, \(N = \phi\).

Case 3: Assume that \(u \notin P(C_n^l)\), then \(D^+_n (u) \subseteq C^l_n\), we have \([C_n^l \cup D^+_n (u)] = |D^+_n (u)|\). For \(x^+ \notin C^l_n\) and if \(x^+ \in D^+_n (u)\), i.e., \(u \in D^+_n (x^+)\), then \([C_n^l \cup D^+_n (u)] \neq |D^+_n (u)|\) (Lemma 2).

So, \(M = \{u|u \in D^+_n (x^+) \land u \in P(C_n^l)\}\) \[42\]

\[
N = \begin{cases} 
v | v \in D^+_n (x^+) \land v \notin P(C_n^l) & n' \geq n \\
\{x^+\} & n' < n \land D^+_n (x^+) \cap C^l_n \neq \phi
\end{cases}.
\] (43)

**Proof.**

Assume that \(v \notin P(C_n^l)\), then \(D^+_n (v) \cap C^l_n \neq \phi\), and \(D^+_n (v) \cap C^l_n \neq \phi\) still holds after \(x^+\) is inserted (Lemma 1).

So, \(M = \phi\).

Case 1: For \(x^+ \in D^+_n (x^+)\), we have \(D^+_n (x^+) \cap C^l_n \neq \phi\), i.e.,
\[\{D^+_n (x^+) \cap C^l_n \neq \phi\}, \text{ so } x^+ \notin N\).

Case 2: Assume that \(v \notin P(C_n^l)\), then \(D^+_n (v) \cap C^l_n \neq \phi\), and \(D^+_n (v) \cap C^l_n \neq \phi\) still holds after \(x^+\) is inserted into \(U\) (Lemma 1).

So, \(N = \{v|v \notin P(C_n^l) \land v \in D^+_n (x^+)\}\) \[44\]

\[
N = \begin{cases} 
\phi & n' \geq n \\
\{x^+\} & n' < n \land D^+_n (x^+) \cap C^l_n \neq \phi
\end{cases}.
\] (45)

**Proof.**

The proof of Theorem 3 is similar to that of Theorem 1.

The computational formula of the set \(M\) and \(N\) for incremental update of \(\overline{P}(C_n^l)\)

**Theorem 3.**

\[
M = \begin{cases} 
\phi & n' \geq n \\
\{u|u \in D^+_n (x^+) \land u \in P(C_n^l)\} & n' < n
\end{cases}.
\] (44)

\[
N = \begin{cases} 
\phi & n' \geq n \\
\{x^+\} & n' < n \land D^+_n (x^+) \cap C^l_n \neq \phi
\end{cases}.
\] (45)
Theorem 4.

\[ M = \phi. \]  
\[ N = \begin{cases} 
\{x^+\} & n' > n \land D_p^+(x^+) \cap Cl_n^\leq \neq \phi \\
\{v \in D_p^-(x^+) \land v \notin \bar{P}(Cl_n^\leq)\} & n' \leq n.
\end{cases} \]

Proof.

The proof of Theorem 4 is similar to that of Theorem 2.

4.3. The Update of Approximations Using Trisecting-and-Acting Model at Deletion of Object

Supposing \( x^- \) is the deleted object and \( \text{Loc}(x^-, U) = i \).

4.3.1. The update of unions of decision classes

Supposing that \( Cl_n^\leq \) is the decision class to which \( x^- \) belongs.

1. The update of upward unions of decision classes

The updated upward unions of decision classes is as following formula:

\[ Cl_n^\geq = \begin{cases} 
Cl_n^\leq - \{x^-\} & n' \geq n \\
Cl_n^\leq & n' < n.
\end{cases} \]

The corresponding column vector of the updated upward union set can be denoted as

\[ Cl_n^\geq = \left[ Cl_n^\leq (1 : i - 1); Cl_n^\leq (i + 1 : m) \right]. \]

2. The update of downward unions of decision classes

The updated downward unions of decision classes is as following formula:

\[ Cl_n^\leq = \begin{cases} 
Cl_n^\leq - \{x^-\} & n' \leq n \\
Cl_n^\leq & n' > n.
\end{cases} \]

The corresponding column vector of the updated upward unions can be denoted as

\[ Cl_n^\leq = \left[ Cl_n^\leq (1 : i - 1); Cl_n^\leq (i + 1 : m) \right]. \]

It is obviously that the dimension of updated column vector will decrease to \( n-1 \) after removal of \( x^- \).

4.3.2. Trisection of universe at deletion of object

The calculations of the set \( N \) and \( M \) are discussed in Theorem 5–8, respectively. The computational results lead to tri-partition of universe naturally.

- The computational formula of the set \( M \) and \( N \) for incrementally update of \( \bar{P}(Cl_n^\leq) \)

Theorem 5.

\[ M = \begin{cases} 
\{x\} & n' \geq n \land x^- \in P(Cl_n^\leq) \\
\phi & n' < n
\end{cases} \]

\[ N = \begin{cases} 
v \notin P(Cl_n^\leq) & n' \geq n \\
v \notin \bar{P}(Cl_n^\leq) \land v \notin D_p^+(x^+) & n' < n
\end{cases} \]

Proof.

\[ M = \begin{cases} 
\{u \mid u \in P(Cl_n^\leq) \land u \notin P(Cl_n^\geq)\} \\
\{u \mid D_p^+(u) \subseteq Cl_n^\leq \land D_p^+(u) \subseteq Cl_n^\geq\}
\end{cases} \]

\[ u \in P(Cl_n^\leq) \text{ will still hold after } x^- \text{ is removed (Lemma 4)} \]

When \( n' \geq n \), we have \( x^- \in Cl_n^\leq \). If \( x^- \notin P(Cl_n^\leq) \), then \( M = \{x\} \).

\[ N = \begin{cases} 
v \notin P(Cl_n^\leq) & n' \geq n \\
v \notin \bar{P}(Cl_n^\leq) \land v \notin D_p^+(x^+) & n' < n
\end{cases} \]

When \( n' \geq n \), we have \( x^- \in Cl_n^\leq \), \( D_p^+(v) \subseteq Cl_n^\leq \), will still hold after \( x^- \) is removed (lemma 3), so \( N = \phi \).

When \( n' < n \), we have \( x^- \notin Cl_n^\leq \). If \( x^- \in D_p^+(v) \), i.e., \( v \in D_p^+(x^+) \) and \( D_p^+(v) \subseteq Cl_n^\leq \), then \( v \notin P(Cl_n^\leq) \).

So \( N = \begin{cases} 
v \notin P(Cl_n^\leq) \land v \notin D_p^+(x^+) & n' \geq n \\
v \notin \bar{P}(Cl_n^\leq) \land v \notin D_p^+(v) & n' < n
\end{cases} \).

- The computational formula of the set \( M \) and \( N \) for incrementally update of \( \bar{P}(Cl_n^\leq) \)

Theorem 6.

\[ M = \begin{cases} 
\{x\} & n' \geq n \\
\{x\} \cup \{u \mid u \in P(Cl_n^\leq) \land D_p^+(x^+) \land D_p^+(u) \land Cl_n^\geq = \phi\} & n' < n
\end{cases} \]

\[ N = \phi. \]

Proof.

\[ M = \begin{cases} 
\{u \mid u \in P(Cl_n^\leq) \land u \notin P(Cl_n^\geq)\} \\
\{u \mid D_p^+(u) \land Cl_n^\leq \neq \phi \land D_p^-(u) \land Cl_n^\leq = \phi\}
\end{cases} \]

When \( n' \geq n \), we have \( x^- \in Cl_n^\leq \), so \( D_p^+(x^+) \cap Cl_n^\leq = \phi \), i.e.,

\( x^- \in M \). And if \( u \in D_p^+(x^+) \), i.e., \( x^- \in D_p^+(u) \) and \( D_p^-(u) \cap Cl_n^\leq = \phi \), then \( u \notin \bar{P}(Cl_n^\leq) \).

To sum up, \( M = \{x\} \cup \{u \mid u \in P(Cl_n^\leq) \land D_p^+(x^+) \land D_p^+(u) \land Cl_n^\geq = \phi\} \).
When \( n' < n \), we have \( x' \notin C_{n'}^\leq, D_{n'}^\geq(u) \cap C_{n'}^\geq \neq \emptyset \) still hold after \( x' \) is removed. If \( D_{n'}^\geq(x') \cap C_{n'}^\geq \neq \emptyset \), then \( M = \{x'\}. \)

\[
N = \{v \notin \overline{P(C_{n'}^\leq)} \land u \in \overline{P(C_{n'}^\geq)}\}
\]

\[
= \{v \notin D_{n'}^\geq(v) \cap C_{n'}^\geq = \emptyset \land D_{n'}^\geq(v) \cap C_{n'}^\geq \neq \emptyset \}.
\]

For \( D_{n'}^\geq(v) \cap C_{n'}^\geq = \emptyset \) will still hold after \( x' \) is removed (Lemma 3). So, \( N = \emptyset. \)

- The computational formula of the set \( M \) and \( N \) for incrementally update of \( P(C_{n'}^\leq) \)

**Theorem 7.**

\[
M = \begin{cases} 
\{x'\} & n' \leq n \land x' \in \overline{P(C_{n'}^\leq)} \\
\emptyset & n' > n
\end{cases}.
\]

(56)

**Proof.**

The proof of Theorem 7 is similar to that of Theorem 5.

- The computational formula of the set \( M \) and \( N \) for incremental update of \( P(C_{n'}^\leq) \)

**Theorem 8.**

\[
M = \begin{cases} 
\{x'\} \cup \{u \in \overline{P(C_{n'}^\leq)} \cap D_{n'}^\geq(x') \land D_{n'}^\geq(u) \cap C_{n'}^\geq = \emptyset\} & n' \leq n \\
\{x'\} & n' > n
\end{cases}.
\]

(58)

**Proof.**

The proof of Theorem 8 is similar to that of Theorem 6.

### 5. ILLUSTRATIVE EXAMPLE

Consider the following example (See Table 2). A set of 16 objects is described by the set of 3 attributes (criteria) \( C = \{c_1, c_2, c_3\}. \) The decision attribute \( d \) classifies objects into three decision classes \( C_{l1}, C_{l2} \) and \( C_{l3} \) which are preference-ordered according to increasing class number.

\[
\begin{align*}
C_{l1}^\leq &= \{o_1, o_2, o_3, o_{10}, o_{14}\}, \\
C_{l2}^\leq &= \{o_1, o_2, o_3, o_4, o_5, o_6, o_9, o_{10}, o_{10}, o_{12}, o_{13}, o_{14}, o_{15}\}, \\
C_{l3}^\leq &= \{o_1, o_2, o_4, o_5, o_6, o_7, o_9, o_{11}, o_{11}, o_{12}, o_{13}, o_{15}, o_{16}\}, \\
C_{l3}^\geq &= \{o_7, o_8, o_{11}, o_{16}\}.
\end{align*}
\]

Let \( P = C \), the approximations of upward and downward unions of decision classes are

\[
\begin{align*}
\overline{P(C_{l1}^\leq)} &= \{o_1, o_2, o_{14}\}, \\
\overline{P(C_{l2}^\leq)} &= \{o_1, o_2, o_3, o_{10}, o_{14}, o_{15}\}, \\
\overline{P(C_{l2}^\leq)} &= \{o_1, o_2, o_3, o_4, o_5, o_6, o_9, o_{10}, o_{10}, o_{12}, o_{13}, o_{14}, o_{15}\}, \\
\overline{P(C_{l3}^\leq)} &= \{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_9, o_{10}, o_{11}, o_{12}, o_{13}, o_{15}, o_{16}\}, \\
\overline{P(C_{l3}^\leq)} &= \{o_7, o_8, o_{11}, o_{16}\}.
\end{align*}
\]

Now we consider the following two cases at time \( t + 1 \):

- A new object \( o_{17} \) inserts the IS (See Table 3).

\[
\begin{align*}
C_{l1}^{\leq t+1} &= \{o_1, o_2, o_3, o_{10}, o_{14}\}, \\
C_{l2}^{\leq t+1} &= \{o_1, o_2, o_3, o_4, o_5, o_6, o_9, o_{10}, o_{12}, o_{13}, o_{14}, o_{15}, o_{17}\}, \\
C_{l3}^{\leq t+1} &= \{o_4, o_5, o_6, o_7, o_9, o_{11}, o_{12}, o_{13}, o_{15}, o_{16}, o_{17}\}.
\end{align*}
\]

#### Table 2 | Illustrative data table.

<table>
<thead>
<tr>
<th>Object</th>
<th>( c_1 )</th>
<th>( c_2 )</th>
<th>( c_3 )</th>
<th>( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( o_1 )</td>
<td>0.8</td>
<td>0.5</td>
<td>1.5</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_2 )</td>
<td>1.0</td>
<td>1.5</td>
<td>8</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_3 )</td>
<td>1.0</td>
<td>3.0</td>
<td>5</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_4 )</td>
<td>2.5</td>
<td>4.0</td>
<td>11</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_5 )</td>
<td>1.2</td>
<td>1.0</td>
<td>7</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_6 )</td>
<td>1.9</td>
<td>4.5</td>
<td>14</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_7 )</td>
<td>2.8</td>
<td>5.5</td>
<td>15</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_8 )</td>
<td>2.4</td>
<td>4.0</td>
<td>13</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_9 )</td>
<td>0.6</td>
<td>3.0</td>
<td>6</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_{10} )</td>
<td>2.0</td>
<td>2.5</td>
<td>6</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_{11} )</td>
<td>2.4</td>
<td>3.5</td>
<td>9</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_{12} )</td>
<td>1.6</td>
<td>3.0</td>
<td>12</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_{13} )</td>
<td>1.8</td>
<td>5.0</td>
<td>9.5</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_{14} )</td>
<td>0.6</td>
<td>3.0</td>
<td>6</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_{15} )</td>
<td>2.0</td>
<td>2.5</td>
<td>6</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_{16} )</td>
<td>2.4</td>
<td>3.5</td>
<td>9</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_{17} )</td>
<td>1.8</td>
<td>2.8</td>
<td>3.5</td>
<td>( C_{l2} )</td>
</tr>
</tbody>
</table>

#### Table 3 | Insertion of one object.

<table>
<thead>
<tr>
<th>Object</th>
<th>( c_1 )</th>
<th>( c_2 )</th>
<th>( c_3 )</th>
<th>( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( o_1 )</td>
<td>0.8</td>
<td>0.5</td>
<td>1.5</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_2 )</td>
<td>1.0</td>
<td>1.5</td>
<td>8</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_3 )</td>
<td>1.0</td>
<td>3.0</td>
<td>5</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_4 )</td>
<td>2.5</td>
<td>4.0</td>
<td>11</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_5 )</td>
<td>1.2</td>
<td>1.0</td>
<td>7</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_6 )</td>
<td>1.9</td>
<td>4.5</td>
<td>14</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_7 )</td>
<td>2.8</td>
<td>5.5</td>
<td>15</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_8 )</td>
<td>2.4</td>
<td>4.0</td>
<td>13</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_9 )</td>
<td>0.6</td>
<td>3.0</td>
<td>6</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_{10} )</td>
<td>2.0</td>
<td>2.5</td>
<td>6</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_{11} )</td>
<td>2.4</td>
<td>3.5</td>
<td>9</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_{12} )</td>
<td>1.6</td>
<td>3.0</td>
<td>12</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_{13} )</td>
<td>1.8</td>
<td>5.0</td>
<td>9.5</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_{14} )</td>
<td>0.6</td>
<td>3.0</td>
<td>6</td>
<td>( C_{l2} )</td>
</tr>
<tr>
<td>( o_{15} )</td>
<td>2.0</td>
<td>2.5</td>
<td>6</td>
<td>( C_{l1} )</td>
</tr>
<tr>
<td>( o_{16} )</td>
<td>2.4</td>
<td>3.5</td>
<td>9</td>
<td>( C_{l3} )</td>
</tr>
<tr>
<td>( o_{17} )</td>
<td>1.8</td>
<td>2.8</td>
<td>3.5</td>
<td>( C_{l2} )</td>
</tr>
</tbody>
</table>
The $D^+_p(o_{17})$ and $D^-_p(o_{17})$ should be calculated firstly.

$$D^+_p(o_{17}) = \{o_1, o_{14}, o_{17}\}.$$  

$$D^-_p(o_{17}) = \{o_4, o_6, o_7, o_8, o_{11}, o_{13}, o_{16}, o_{17}\}.$$  

According to Theorems 1–4, we have

$$P(\mathbb{C}_1^{o_t}) = P(\mathbb{C}_1^{o_t}) - M \cup N$$  

$$= P(\mathbb{C}_1^{o_t}) - \phi \cup \phi$$  

$$= \{o_1, o_2, o_{14}\}.$$  

$$\mathcal{P}(\mathbb{C}_1^{o_t}) = \mathcal{P}(\mathbb{C}_1^{o_t}) - M \cup N$$  

$$= \mathcal{P}(\mathbb{C}_1^{o_t}) - \phi \cup \{o_{17}\}$$  

$$= \{o_1, o_2, o_3, o_5, o_6, o_9, o_{10}, o_{12}, o_{13}, o_{14}, o_{16}, o_{17}\}.$$  

$$P(\mathbb{C}_2^{o_t}) = P(\mathbb{C}_2^{o_t}) - M \cup N$$  

$$= P(\mathbb{C}_2^{o_t}) - \phi \cup \{o_{17}\}$$  

$$= \{o_4, o_5, o_6, o_7, o_9, o_{10}, o_{11}, o_{12}, o_{13}, o_{14}, o_{16}, o_{17}\}.$$  

$$\mathcal{P}(\mathbb{C}_2^{o_t}) = \mathcal{P}(\mathbb{C}_2^{o_t}) - M \cup N$$  

$$= \mathcal{P}(\mathbb{C}_2^{o_t}) - \phi \cup \{o_{17}\}$$  

$$= \{o_5, o_6, o_7, o_8, o_{10}, o_{11}, o_{12}, o_{13}, o_{14}, o_{16}, o_{17}\}.$$  

According to Theorem 5–8, we have

$$P(\mathbb{C}_3^{o_t}) = P(\mathbb{C}_3^{o_t}) - M \cup N$$  

$$= P(\mathbb{C}_3^{o_t}) - \phi \cup \phi$$  

$$= \{o_7, o_8, o_{16}\}.$$  

$$\mathcal{P}(\mathbb{C}_3^{o_t}) = \mathcal{P}(\mathbb{C}_3^{o_t}) - M \cup N$$  

$$= \mathcal{P}(\mathbb{C}_3^{o_t}) - \phi \cup \phi$$  

$$= \{o_7, o_8, o_{16}\}.$$  

The object $o_{10}$ gets out of the IS (See Table 4).

$\mathbb{C}_1^{o_t} = \{o_1, o_2, o_3, o_{14}\}.$  

$\mathbb{C}_2^{o_t} = \{o_1, o_2, o_3, o_4, o_5, o_6, o_9, o_{10}, o_{12}, o_{13}, o_{14}, o_{15}\}.$  

$\mathbb{C}_3^{o_t} = \{o_4, o_5, o_6, o_7, o_8, o_{11}, o_{12}, o_{13}, o_{15}, o_{16}\}.$  

$D^+_p(o_{10}) = \{o_1, o_{10}, o_{14}, o_{15}\}.$  

$D^-_p(o_{10}) = \{o_4, o_7, o_8, o_{10}, o_{11}, o_{16}\}.$  

According to Theorem 5–8, we have

$$P(\mathbb{C}_1^{o_t}) = P(\mathbb{C}_1^{o_t}) - M \cup N$$  

$$= P(\mathbb{C}_1^{o_t}) - \phi \cup \phi$$  

$$= \{o_1, o_2, o_{14}\}.$$  

According to the preceding methods using trisecting and acting in update of approximations at object’s insertion or removal, their corresponding incremental update algorithms are proposed. All the algorithms are described by matrix considering that representation and calculation of matrix are intuitive and concise.

6. Incremental Update Matrix Algorithm of the Union Set’s Approximations at Insertion or Deletion of One Object
The Algorithm 1 is finished.

Assume that the number of objects, number of attributes and number of dominance classes are \(n_1, n_2\) and \(m\), respectively. The time complexity of Algorithm 1 is equal to \(O(m \cdot (8 + 2 \cdot n_2) \cdot n_1)\). Thus it approximately equal to \(O(m \cdot n_2 \cdot n_1)\).

**Algorithm 1:** A matrix-based incremental algorithm for updating approximations of unions of decision classes when a new object adds to an information system.

**Input:**
1. \(D_p\) at time \(t\).
2. \(\forall n \in T, P\left(\mathcal{C}_{n}^p\right), \bar{P}\left(\mathcal{C}_{n}^-\right), P\left(\mathcal{C}_{n}^c\right), \bar{P}\left(\mathcal{C}_{n}^-\right)\) at time \(t\).
3. An inserted object \(x^*\).

**Output:**
\(\forall n \in T, P\left(\mathcal{C}_{n}^p\right), \bar{P}\left(\mathcal{C}_{n}^-\right), P\left(\mathcal{C}_{n}^c\right), \bar{P}\left(\mathcal{C}_{n}^-\right)\).

**Begin**
Compute \(D_p^x (x^+)\) and \(D_p^- (x^+)\).
Update \(D_p\) at time \(t\) to \(D_p\) at time \(t + 1\). Insert \(D_p^x (x^+)\) and \(D_p^- (x^+)\).
After the \((i-1)\)th column and after the \((i-1)\)th row.
Update \(C_{n}^{p}\) and \(C_{n}^{-}\) at time \(t\) to \(C_{n}^{p}\) and \(C_{n}^{-}\) at time \(t + 1\).
For \(n = 1, \ldots, m\) do
If \((n < g)\) Then// \(g\) is the index of the class which the object \(x^+\) is // assigned
If \(\left[\left|D_p (x^+), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = |D_p^x (x^+)\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) + x^+\) \nEnd.
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) + D_p^- (x^+)\) \nEnd.
End.
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) - D_p^x (x^+)\) \n\P\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) + x^+\) \nEnd.
End.
End.
End.
If \((n = g)\) Then
If \(\left[\left|D_p (x^+), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = |D_p^x (x^+)\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) + x^+\) \nEnd.
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) + D_p^- (x^+)\) \nEnd.
End.
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) - D_p^x (x^+)\) \n\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) + x^+\) \nEnd.
End.
End.
End.
If \((n > g)\) Then
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) - A_n\) \nEnd.
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) + x^+\) \nEnd.
If \(\left[\left|D_p (x), \bar{P}\left(\mathcal{C}_{n}^-\right)\right| = 0\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) + x^+\) \nEnd.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
The Algorithm 2 is finished.

Similar to Algorithm 1, the time complexity of Algorithm 2 is approximately equal to \(O(m \cdot n_2 \cdot n_1)\).

The non-incremental algorithm for computing the approximations is given below in order to compare its performance with that of the incremental algorithm in the next subsection.

**Algorithm 2:** A matrix-based incremental algorithm for updating approximations of unions of decision classes when an object gets out of an information system.

**Input:**
1. \(D_p\) at time \(t\).
2. \(\forall n \in T, P\left(\mathcal{C}_{n}^p\right), \bar{P}\left(\mathcal{C}_{n}^-\right), P\left(\mathcal{C}_{n}^c\right), \bar{P}\left(\mathcal{C}_{n}^-\right)\) at time \(t\).
3. An deleted object \(x^-\).

**Begin**
Update \(D_p\) at time \(t\) //delete the \(i\)th column and the \(i\)th row Update \(C_{n}^{p}\) and \(C_{n}^{-}\) at time \(t\).
For \(n = 1, \ldots, m\) do
If \((n < g)\) Then// \(g\) is the index of the class which the object \(x^-\) //is assigned
If \(\left[\left|D_p (x^-), \mathcal{C}_{n}^-\right| = |D_p^x (x^-)\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) - x^-\) \nEnd.
\(\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) - x^-\).
End.
For each \(u \in \bar{P}\left(\mathcal{C}_{n}^p\right) \cap D_p^x (x^-)\)
If \(\left[\left|D_p^x (u), \mathcal{C}_{n}^-\right| = 0\right]\), Then \(\bar{P}\left(\mathcal{C}_{n}^p\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^p\right) - u\) \nEnd.
End.
End.
End.
End.
For each \(u \in D_p^x (x^-) \cap D_p^x (x^-)\)
If \(\left[\left|D_p^x (u), \mathcal{C}_{n}^-\right| = 0\right]\), Then \(\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) + u\) \nEnd.
End.
End.
End.
End.
If \((n = g)\) Then
If \(\left[\left|D_p (x^-), \mathcal{C}_{n}^-\right| = 0\right]\), Then \(P\left(\mathcal{C}_{n}^p\right) \leftarrow P\left(\mathcal{C}_{n}^p\right) - x^-\) \n\bar{P}\left(\mathcal{C}_{n}^-\right) \leftarrow \bar{P}\left(\mathcal{C}_{n}^-\right) - x^-\)
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
End.
If \((n > g)\) Then

For each \(u \not\in P (C_{n}^{u}) \land u \in D_{p}(x')\)

\[\text{If } \left[D_{p}^{u'}(u), C_{n}^{u'}\right] \Rightarrow \left[D_{p}^{u'}(u)\right] \text{, Then } P(C_{n}^{u}) \leftarrow P(C_{n}^{u}) + u\]

End.
End.

End.

End.

End.

End.

End.

End.

End.

End.

End.

End.

The Algorithm 3 is finished.

The time complexity of Algorithm 3 is equal to \(O(m \cdot (n1^2 - (n2 + 8 \cdot n1))\). It is approximately equal to \(O(m \cdot n1^2 \cdot n2)\).

Algorithms 1–3 have the same space complexity due to the same memory space.

Algorithm 3: A matrix-based algorithm for computing approximations of unions of decision classes.

**Input:**

\(IS = \{U, C \cup d, V, \mathbb{f}\}, U/\mathbb{H}(d)\).

**Output:**

\(\forall n \in T, P(C_{n}^{u}), \overline{P}(C_{n}^{u}), \overline{P}(C_{n}^{u}), \overline{P}(C_{n}^{u})\).

**Begin**

Computing \(D_{p}, C_{n}^{u}, C_{n}^{u}\) by using IS.

Computing matrix multiplications \(D_{p} \cdot C_{n}^{u}, D_{p}^{u} \cdot C_{n}^{u}, D_{p} \cdot C_{n}^{u}\) and \(D_{p}^{u} C_{n}^{u}\). Denote by \(M_{1}, M_{2}, M_{3}\) and \(M_{4}\) respectively.

Computing Column Vector by sum of row \(\sum(D_{n}^{u}, 2)\) and \(\sum(D_{n}, 2)\).

\[P(C_{n}^{u}) \leftarrow \left(M_{1} \geq \sum(D_{n}^{u}, 2)\right)\]

\[\overline{P}(C_{n}^{u}) \leftarrow (M_{2} \geq \text{ones}(n, 1))\]

\[\overline{P}(C_{n}^{u}) \leftarrow (M_{3} \geq \sum(D_{n}, 2))\]

\[\overline{P}(C_{n}^{u}) \leftarrow (M_{4} \geq \text{ones}(n, 1))\]

**End.**

7. THE TEST AND EVALUATION IN UCI DATA SET

Three data sets are selected (i.e., ‘Wine’ data set, ‘Car evaluation’ data set and ‘Abalone’ data set) from the machine learning data repository UCI (http://archive.ics.uci.edu/ml/) to test the performance of Algorithms 1 and 2 proposed in this paper and the existing Algorithm 3 (non-incremental updating algorithm) in order to verify the effectiveness of the proposed algorithms. Descriptions of the selected data sets are shown in Table 5.

Experimental Platform: CPU Intel Core i7-4510U (2.00 GHz), 8.0G Memory, Windows 8 operation system, Matlab7.0 development tool. Experimental method is as follows:

To show the time efficiency of dynamic algorithm (Algorithms 1 and 2) and compare with the existing static algorithm (Algorithm 3), each of selected data sets is divided into 10 sub-data sets. The generation of sub-data set follows four principles, take the ‘wine’ data set as an example to explain how to generate sub-data set from whole data set. Firstly, the ‘wine’ data set is divided into 10 subsets, each sub-data set is named as wine 1, wine 2, wine 3, wine 4, … and wine 10, respectively. Secondly, the size of wine 1, wine 2, wine 3, … and wine 9 is one-tenth, two-tenths, three-tenths, … and nine-tenths of that of ‘wine’ data set, respectively, wine 10 is the copy of wine data set. Thirdly, there exists the following inclusion relation among these ten sub-data sets, the wine 10 contains wine 9, wine 9 contains wine 8, …, wine 3 contains wine 2 and wine 2 contains wine 1. Fourthly, the number of objects in a certain class of each sub-data set should be proportional to the size of the sub-data set in order to keep the original proportion of distribution of each class in whole data set. The distributions of each class in sub-data set of ‘wine’ are shown in Table 6.

The running time of the proposed Algorithms 1 and 2 and the existing Algorithm 3 can be gained by executing the corresponding programs on each of these 10 sub-data sets. Concerned programs which update four approximations of DRSA while object set varies are all developed on Matlab7.0 platform.

The experimental results are depicted in Fig. 4, where the x-coordinate pertains to the test sub-data sets, while y-coordinate concerns the computing time of updating approximations. The following conclusions can be drawn from the experimental results on three UCI data sets:

i. The incremental updating matrix algorithms (Algorithms 1 and 2) have more effectiveness compared to the non-incremental updating matrix algorithm (Algorithm 3), showing that the time consume in incremental updating algorithm is obviously less that of the non-incremental ones.

ii. For the same data set, the computational times of the incremental updating matrix algorithms and the non-incremental updating matrix algorithm all grow up with the increasing size

<table>
<thead>
<tr>
<th>Table 5</th>
<th>A descriptions on the selected three data sets.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Number of Attributes</td>
</tr>
<tr>
<td>Wine</td>
<td>13</td>
</tr>
<tr>
<td>Car evaluation</td>
<td>6</td>
</tr>
<tr>
<td>Abalone</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 6</th>
<th>The distribution of classes in each subset of Wine.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-Data Set</td>
<td>Total Objects</td>
</tr>
<tr>
<td>Wine 1</td>
<td>18</td>
</tr>
<tr>
<td>Wine 2</td>
<td>36</td>
</tr>
<tr>
<td>Wine 3</td>
<td>54</td>
</tr>
<tr>
<td>Wine 4</td>
<td>72</td>
</tr>
<tr>
<td>Wine 5</td>
<td>90</td>
</tr>
<tr>
<td>Wine 6</td>
<td>107</td>
</tr>
<tr>
<td>Wine 7</td>
<td>125</td>
</tr>
<tr>
<td>Wine 8</td>
<td>142</td>
</tr>
<tr>
<td>Wine 9</td>
<td>160</td>
</tr>
<tr>
<td>Wine 10</td>
<td>178</td>
</tr>
</tbody>
</table>
Figure 4 | The curve of time-consuming for updating of approximations at insertion or deletion of a one objects.
of sub-data set. Furthermore, Algorithms 1 and 2 are much more faster than Algorithm 3, and the difference between incremental algorithms and the non-incremental algorithm are getting larger and larger while the size of sub-data set increases.

iii. The running times of Algorithm 3 increases sharply with the increasing size of data while the running times of Algorithms 1 and 2 increase very slowly, demonstrating that the larger the size of data, the greater the advantage of Algorithms 1 and 2.

iv. For the incremental update algorithm, the running time of object's insertion is more less than that of object's deletion in the same sub-data set. The reason is that there are more loop structure in Algorithm 2, which lead to more time consume.

It can be seen from the above four conclusions that the incremental update approaches of approximations with trisecting-and-acting model of three-way decisions are feasible and outperform the existing non-incremental approach.

8. CONCLUSIONS

Incrementally updating approximations in rough sets is a critical issue for knowledge update, maintenance and data mining related task in dynamic IS. In this paper, considering the dynamic scenario of one object's insertion or deletion, an incremental updated mechanism of approximations in DRSA is introduced from the perspective of three-way decisions and three update steps are derived from the updated mechanism. Then we suggested a concrete incremental updating approach of approximations in DRSA while the object set varies, in which Boolean column vectors are used as an expression tool of subsets as well as a calculation tool of approximations. The proposed methods can effectively update the four approximations of DRSA on the basis of the prior approximations. With a numerical example and some test results on UCI data set, we can conclude that the proposed incremental vector method for updating the approximations of DRSA is feasible and can effectively reduce the computational time in comparison to the existing non-incremental vector method when the set of objects changes. One of the further work is to investigate the approaches for updating approximations of DRSA under variations of attribute and attribute value and study the corresponding vector-based incremental algorithms.

CONFLICT OF INTEREST

The authors declare no conflicts of interest.

AUTHORS’ CONTRIBUTIONS

All authors have contributed to this paper. The individual responsibilities and contribution of all authors can be described as follows: The idea of this paper was put forward by Lei Wang, he also wrote the paper. Min Li summarized the existing work of the research problem and presented the illustrative example in Section 5. The works in Section 3 were done by Jun Ye. The submission and revision of this paper was completed by Lei Wang and Xiang Yu. The tests on UCI data sets were performed by Ziqi Wang and Shaobo Deng. The other works of this paper were done by Lei Wang.

Funding Statement

This work was supported by the fund of Development Plan of Jiangxi Province of China for the Young and Middle-Aged Teachers in Universities to Pursue Study Abroad as a Visiting Scholar (Jiangxi Education Administration Letter [2016] No. 169), the National Natural Science Foundation of China (Nos. 61562061, 61363047 and 61703199), and the fund of Science and Technology Project of Ministry of Education of Jiangxi province of China (No. GJJ170995).

ACKNOWLEDGMENTS

The author would like to thank reviewers of the paper for their constructive comments which greatly improve the quality of this paper. This work was supported by the fund of Development Plan of Jiangxi Province of China for the Young and Middle-Aged Teachers in Universities to Pursue Study Abroad as a Visiting Scholar (Jiangxi Education Administration Letter [2016] No. 169), the National Natural Science Foundation of China (No. 61562061, No. 61363047, No. 61703199), and the fund of Science and Technology Project of Ministry of Education of Jiangxi province of China (No. GJJ170995).

REFERENCES


