Incremental Computing Approximations with the Dynamic Object set in Interval-valued Ordered Information System

Jianhang Yu, Weihua Xu∗†
School of Mathematics and Statistics
Chongqing University of Technology
Chongqing, 400054, P.R. China
yujh2013@foxmail.com, chxuwh@gmail.com

Abstract. Rough set theory has been successfully used in formation system for classification analysis and knowledge discovery. The upper and lower approximations are fundamental concepts of this theory. The new information arrives continuously and redundant information may be produced with the time in real-world application. So, then incremental learning is an efficient technique for knowledge discovery in a dynamic database, which enables acquiring additional knowledge from new data without forgetting prior knowledge, which need to be updated incrementally while the object set get varies over time in the interval-valued ordered information system. In this paper, we analyzed the updating mechanisms for computing approximations with the variation of the object set. Two incremental algorithms respectively for adding and deleting objects with updating the approximations are proposed in interval-valued ordered information system. Furthermore, extensive experiments are carried out on six UCI data sets to verify the performance of these proposed algorithms. And the experiments results indicate the incremental approaches significantly outperform non-incremental approaches with a dramatic reduction in the computational time.

Keywords: Approximations; Dynamic database; Incremental learning; Interval-valued ordered information system

∗Address for correspondence: No. 69 Hongguang Road, Banan District, Chongqing, P.R. China
†Also works: Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information, Nanjing University of Science and Technology, Nanjing, 210094, P.R. China and Key Laboratory of Oceanographic Big Data Mining & Application of Zhejiang Province, Zhejiang Ocean University, Zhoushan, Zhejiang 316022, P R China
1. Introduction

Rough set theory was proposed by Pawlak[28-30], is an extension of the classical set theory and could be regarded as a mathematical and soft computing tool to handle imprecision, vagueness and uncertainty in data analysis. This relatively new soft computing methodology has received great attention in recent years, and its effectiveness has been confirmed successful applications in many science and engineering fields, such as pattern recognition, data mining, image processing, medical diagnosis and so on[18,19]. Rough set theory is built on the basis of the classification mechanism, it is classified as the equivalence relation in a specific universe, and the equivalence relation constitutes a partition of the universe. A concept or more precisely the extension of a concept is represented by a subset of a universe of objects and is approximated by a pair of definable concepts of a logic language. The main idea of rough set theory is the use of a known knowledge in knowledge base to approximate the inaccurate and uncertain knowledge. It seems to be fundamental importance to artificial intelligence and cognitive sciences. The classical rough set is through the indiscernibility relation obtained equivalence classes and construct the lower and upper approximations. In many application fields the preference-ordered relation play an important role. To solve this problem, Greco et al. have proposed an extension of Pawlak’s rough set approach, which is called the Dominance-based rough set approach(DRSA)[11-15]. In DRSA, where condition attributes are criteria and classes and the dominance classes are sets of objects defined by using a dominance relation[4, 41].

In real-world application, data in information system are generated and collected dynamically, and the knowledge discovery by RST need to be updating accordingly[32]. The incremental technique is an effective method to updating knowledge by dealing with the new added-in data set without re-implementing the original data mining algorithm[26]. With respect to the different angles to recognize the dynamics in rough sets, there exist two main viewpoints. The first one is based on the view of information table. Since an information table consists of data objects, data attributes and data attribute values[25] recent researches focus on the three types of variations, namely, variation of objects[1,6,20-23,42], variation of attributes[24, 44], variation of attributes’ values[5]. The second one is based on the view of pre-topology[31]. The classification of dynamics in rough sets is divided into two aspects: synchronic dynamics and diachronic dynamics[7]. Furthermore, Ciucci[9] listed four main streamlines to investigate dynamics in rough sets, namely, lower and upper approximations[3, 42], reduce and rules[8], quality indexes[16, 20] and formal logical[17, 27]. To sum up, both viewpoints provide a basic and clear framework on dynamic studies of rough sets. Shan and Ziarko presented a discernibility-matrix based incremental methodology to find all maximally generalized rules. Bang and Bien proposed another incremental inductive learning algorithm to find a minimal set of rules for a decision table without recomputing all the set of instances when another instance set is added into the universe [1]. Tong and An developed an algorithm based on the -decision matrix for incremental learning rules. They listed seven cases that would happen when a new sample enters the system. Zheng and Wang developed a rough set and rule tree based incremental knowledge acquisition algorithm, RRITA, to update knowledge more quickly when new objects are added or removed from a given dataset[44]. Hu et al. constructed a novel incremental attribute reduction algorithm when new objects are added into a decision information system. Błaszczyński and Słowiński discussed the incremental induction of decision rules from dominance-based rough approximations to select the most interesting representatives in the final set of rules. Fan et al. proposed an approach of incremental rule induction based on rough sets[10]. In addition, Liu et al. proposed an incremental approach as well as its algorithm for inducing interesting knowledge
when objects change over time [20]. Then, Liu et al. further introduced the incremental matrix and presented a new optimization approach for knowledge discovery [21]. Followed by Liu’s work, Li et al. proposed an incremental approach for updating approximations in dominance-based rough sets [23]. Zhang et al. proposed a method for dynamic data mining based on neighborhood rough sets [42], and they further presented a parallel method for computing rough set approximations [43]. As an efficient data analysis technique, the rough set based incremental approaches have become one the hot topics on extraction of knowledge from changing data sets in recent decades and have achieved fruitful results.

However, mainly study on incremental computing approximations concerned in the certainly single-valued or set-valued information system, but little attention has been paid to the interval-valued information system and ordered information system. And they are very important type of data tables, and generalized models of single-valued information system. Xu and Qian et al. have did some studies in ordered information system [34-38]. In recent years, some problems of decision making have been investigated in the context of interval information system. Qian et al. introduced a dominance relation to interval information systems and interval decision tables and established a rough set approach based on dominance relation for decision-making analysis in the context of interval value [33]. Yang et al. investigated the interval-valued information system based on dominance relation [39, 40]. The upper and lower approximations are fundamental of studying in rough set theory. In this paper, we investigate the incremental approaches for updating approximations with dynamic object set in interval-valued ordered information system. We focus on updating approximations under the variation of the object set in interval-valued ordered information system. We proposed two incremental updating algorithms when the objectees are deleted or inserted, respectively. At last, the performances of two incremental algorithms are evaluated on several variety of data sets.

The remainder of this paper is organized as follows. In Section 2, some basic concepts of RST and interval-valued ordered information systems are simply introduced. In Section 3, the principles and some illustrated examples for incremental updating approximations with the variation of object set are presented. We proposed the incremental algorithms for computing approximations based on the updating principles in Section 4. In Section 5, performance evaluations are illustrated and the experiment results have exhibited. The paper ends with conclusions shown in Section 6.

2. Rough sets and interval-valued ordered information system

In this section, Some basic concepts and results of rough sets are outlined and we introduce a dominance relation to an interval information system, the rough set model of interval information system and some of their important properties were introduced. More details can refer to Literature [11-15,28,29,33,40].

For a non-empty set $U$, we call it the universe of discourse. The class of all subsets of $U$ is denoted by $P(U)$. For $X \in P(U)$, the equivalence relation $R$ in a Pawlak approximation space $(U, R)$ partitions the universe $U$ into disjoint subsets. Such a partition of the universe is a quotient set of $U$ and is denoted by $U/R = \{[x]_R | x \in U\}$, where $[x]_R = \{y \in U | (x, y) \in R\}$ is the equivalence class containing $x$ with respect to $R$. In the view of granular computing, equivalence classes are the basic building blocks for the representation and approximation of any subset of the universe of discourse. Each equivalence class may be viewed as a granule consisting of indistinguishable elements. The basic concept $X \in P(U)$, one
can characterize $X$ by a pair of upper and lower approximations which are
\[
\overline{R}(X) = \{ x \in U | [x]_R \cap X \neq \emptyset \}, \\
\underline{R}(X) = \{ x \in U | [x]_R \subseteq X \}.
\]

Here, $pos(X) = \overline{R}(X)$, $neg(X) = \neg \overline{R}(X)$, $bn(X) = \overline{R}(X) - \underline{R}(X)$ are called the positive region, negative region, and boundary region of $X$, respectively.

Pawlak upper and lower approximations divided the universe into three disjoint regions, namely, positive region, negative region and boundary region. These regions have qualitative semantics and reflect the positive certainty, negative certainty and uncertainty, respectively. While the Pawlak rough set has a severe limitation. The relationship between equivalence classes and the basic set are strict that there are no fault tolerance mechanisms. Quantitative information about the degree of overlap of the equivalence classes and the basic set is not taken into consideration. Therefore, neither wider relationships nor quantitative information can be utilized. In fact, there are some degrees of inclusion relations between sets, and the extent of overlap of sets is important information to consider in applications. The classical rough set model must be improved and expansions of the model that include quantification are of particular value.

An interval-valued information system is a quadruple $I = (U, AT, V, f)$, where $U$ is a finite non-empty set of objects and $AT$ is a finite non-empty set of attributes, $v = \cup_{a \in AT} \forall_a$ and $\forall_a$ is a domain of attribute $a$, $f : U \times AT \to V$ is a total function such that $f(x, a) \in \forall_a$ for every $a \in AT, x \in U$, called an information function, where $\forall_a$. is a set of interval numbers. Denoted by
\[
f(x, a) = [a^L(x), a^U(x)] = \{ p | a^L(x) \leq a^U(x), a^L(x), a^U(x) \in R \},
\]
we call it the interval number $x$ under the attribute $a$. In particular, $f(x, a)$ would degenerate into a real number if $a^L(x) = a^U(x).$ Under this consideration, we regard a single-valued information system as a special form of interval information system.

In particular decision-making analysis, we always consider a binary dominance relation between objects that are possibly dominant in terms of value of an attribute set in an interval information system. In general, an increasing preference and a decreasing preference are considered by a decision maker. If the domain of an attribute is ordered according to a decreasing or increasing preference, then the attribute is a criterion.

An interval-valued information system is called interval-valued ordered information system if all attributes are criterions, referred to as IvOIS. It is assumed that the domain of a criterion $a \in AT$ is completely pre-ordered by an outranking relation $\geq_a$ and $x \geq_a y$ means that $x$ is at least as good as $y$ with respect to the criterion $a$. For a subset of attribute $A \subseteq AT$, we define $x \geq_a y \iff \forall a \in A, x \geq_a y$. In other words, $x$ is at least as good as $y$ with respect to all attributes in $A$. In the following, we introduce a dominance relation that identifies dominance classes to an interval-valued ordered information system. In a given IvOIS, we say that $x$ dominates $y$ with respect to $A \subseteq AT$ if $x \geq_A y$, and denoted by $xR_A^\geq y$. That is
\[
R_A^\geq = \{(y, x) \in U \times U | y \geq_A x\}.
\]

It means that if $(x, y) \in R_A^\geq$, then $y$ dominates $x$ with respect to $A$. In other words, $y$ may have a better property than $x$ with respect to $A$ in reality. In the similar way, the relation $R_A^\leq$ (called a dominated relation) can be defined as follows:
\[
R_A^\leq = \{(y, x) \in U \times U | x \geq_A y\}.
\]
For $A \subset AT$ and $A = A_1 \cup A_2$, if the attributes set $A_1$ according to increasing preference and $A_2$ according to decreasing preference, then the two binary relations can be defined more precisely as follows:

$$R^\geq_A = \{(y, x) \in U \times U | a^U_1(y) \geq a^L_1(x), a^U_2(y) \geq a^L_2(x), \forall a_1 \in A_1;\}
$$

$$= \{(y, x) \in U \times U | (y, x) \in R^\geq_A \}. $$

$$R^\leq_A = \{(y, x) \in U \times U | a^L_1(y) \leq a^U_1(x), a^L_2(y) \leq a^U_2(x), \forall a_1 \in A_1;\}
$$

$$= \{(y, x) \in U \times U | (y, x) \in R^\leq_A \}. $$

Let $I^\geq = (U, AT, V, f)$ be an interval-valued ordered information system and $A \subset AT$, from the above definition of $R^\geq_A$ and $R^\leq_A$, the following properties can be easily obtained.

$$R^\geq_A = \bigcap_{a \in A} R^\geq_{(a)}, \quad R^\leq_A = \bigcap_{a \in A} R^\leq_{(a)}.$$  

And $R^\geq_A$, $R^\leq_A$ are reflexive, $R^\geq_A$, $R^\leq_A$ are asymmetric and $R^\geq_A$, $R^\leq_A$ are transitive.

The dominance class induced by the dominance relation $R^\geq_A$ is the set of objects dominating $x$, i.e.

$$[x]^\geq_A = \{ a^L_1(x), a^U_1(x) \} \forall a_1 \in A_1; a^L_2(x), a^U_2(x) \} \forall a_2 \in A_2 \},$$

and the set of objects dominated by $x$ as follows.

$$[x]^\leq_A = \{ a^L_1(x), a^U_1(x) \} \forall a_1 \in A_1; a^L_2(x), a^U_2(x) \} \forall a_2 \in A_2 \}.$$

Where $[x]^\geq_A$ describes the set of objects that may dominates $x$ and $[x]^\leq_A$ describes the set of objects that may dominated by $x$ in terms of $A$ in an interval-valued ordered information system, which are called the $A$-dominating set and the $A$-dominated set with respect to $x \in U$, respectively.

In many real application regions, one also can define the dominance relation on the universe with interval values through using others, the more details can be found in reference. Furthermore, no matter which dominance relation can be obtained similar to any one what have been investigated. Therefore, we just only adopt the dominance relation $R^\geq_A$ for studying interval-valued ordered information system in this paper. For simplicity and without any loss of generality, in the following we only consider attributes with increasing preference.

**Example 2.1.** An interval-valued ordered information system is presented in Table 1. It is a case of the diagnosis of myocardial infarction, where $U = \{x_1, x_2, \ldots, x_{10} \}$ representatives of ten different patients and $AT = \{a_1, a_2, \ldots, a_5 \}$ representatives of several enzymes related to the diagnosis of myocardial infarction. Where $a_1$ represents aspartate amino transferase (AST), $a_2$ represents Lactate dehydrogenase (LDH) and isoenzyme, $a_3$ represents Alfa hydroxybutyratdehydrogenase(α-HBDH), $a_4$ represents Creatine Kinase(CK), $a_5$ represents Creatine Kinase isoenzymes (CKMB). Compute the classification induced by the dominance relation $R^\geq_{AT}$. And the different decision attribute values mean different diagnosis results.
From the Table 1, using the above property can be obtained that

\[
\begin{align*}
[x_1]_{AT}^{\geq} &= \{x_1, x_5, x_7, x_8\}, \quad [x_2]_{AT}^{\geq} = \{x_2, x_7, x_9\}, \quad [x_3]_{AT}^{\geq} = \{x_3, x_9\}, \\
[x_4]_{AT}^{\geq} &= \{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}, \quad [x_5]_{AT}^{\geq} = \{x_5\}, \quad [x_6]_{AT}^{\geq} = \{x_6, x_8, x_9\}, \\
[x_7]_{AT}^{\geq} &= \{x_7\}, \quad [x_8]_{AT}^{\geq} = \{x_8\}, \quad [x_9]_{AT}^{\geq} = \{x_9\}, \quad [x_{10}]_{AT}^{\geq} = \{x_7, x_8, x_9, x_{10}\}.
\end{align*}
\]

So we can find that dominance classes in \(U/R_{AT}^{\geq}\) do not constitute a partition of \(U\) in general, but constitute a covering of \(U\).

### Table 1. An interval-valued ordered information system.

<table>
<thead>
<tr>
<th>(U)</th>
<th>(AST)</th>
<th>(LDH)</th>
<th>(\alpha - HBDH)</th>
<th>(CK)</th>
<th>(CKMB)</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>[10,40]</td>
<td>[100,240]</td>
<td>[105,195]</td>
<td>[5,195]</td>
<td>[0,24]</td>
<td>2</td>
</tr>
<tr>
<td>(x_2)</td>
<td>[10,30]</td>
<td>[80,210]</td>
<td>[80,180]</td>
<td>[10,190]</td>
<td>[0,24]</td>
<td>1</td>
</tr>
<tr>
<td>(x_3)</td>
<td>[12,45]</td>
<td>[105,248]</td>
<td>[100,210]</td>
<td>[7,203]</td>
<td>[0,23]</td>
<td>2</td>
</tr>
<tr>
<td>(x_4)</td>
<td>[5,30]</td>
<td>[60,80]</td>
<td>[90,160]</td>
<td>[0,180]</td>
<td>[0,10]</td>
<td>1</td>
</tr>
<tr>
<td>(x_5)</td>
<td>[10,46]</td>
<td>[110,246]</td>
<td>[105,195]</td>
<td>[6,198]</td>
<td>[0,26]</td>
<td>2</td>
</tr>
<tr>
<td>(x_6)</td>
<td>[10,30]</td>
<td>[90,200]</td>
<td>[96,206]</td>
<td>[5,180]</td>
<td>[0,10]</td>
<td>1</td>
</tr>
<tr>
<td>(x_7)</td>
<td>[13,60]</td>
<td>[100,240]</td>
<td>[115,200]</td>
<td>[20,260]</td>
<td>[5,30]</td>
<td>3</td>
</tr>
<tr>
<td>(x_8)</td>
<td>[10,50]</td>
<td>[120,260]</td>
<td>[115,210]</td>
<td>[8,196]</td>
<td>[5,28]</td>
<td>2</td>
</tr>
<tr>
<td>(x_9)</td>
<td>[16,80]</td>
<td>[140,260]</td>
<td>[102,300]</td>
<td>[40,320]</td>
<td>[10,60]</td>
<td>3</td>
</tr>
<tr>
<td>(x_{10})</td>
<td>[8,32]</td>
<td>[60,196]</td>
<td>[80,178]</td>
<td>[6,160]</td>
<td>[2,20]</td>
<td>1</td>
</tr>
</tbody>
</table>

Based on the above set, approximations will be considered with respect to a dominance relation \(R_{AT}^{\geq}\) in an interval-valued ordered information system. The original rough set approach proved to be very useful in dealing with inconsistency problems following from the information granulation. The original rough set idea is failing, however, when preference-orders of attributes domains are to be taken into account [2].

Let \(I^\geq = (U, AT, V, f)\) be an interval-valued ordered information system. For any \(X \subseteq U\) and \(A \subseteq AT\), the lower and upper approximations of \(X\) with respect to a dominance relation \(R_{AT}^{\geq}\) are defined as follows:

\[
\begin{align*}
\overline{R}_{AT}^{\geq}(X) &= \bigcup\{[x]_{AT}^{\geq} \mid [x]_{AT}^{\geq} \subseteq X\}, \\
\overline{R}_{AT}^{\geq}(X) &= \bigcup\{[x]_{AT}^{\geq} \mid [x]_{AT}^{\geq} \cap X \neq \emptyset\}.
\end{align*}
\]

From the definition, one can easily notice that \(\overline{R}_{AT}^{\geq}(X)\) is a set of objects that belong to \(X\) with certainty and \(\overline{R}_{AT}^{\geq}(X)\) is a set of objects that possibly belong to \(X\). It is similarly to Pawlak rough set that the \(Bn_{AT}(X) = \overline{R}_{AT}^{\geq}(X) \setminus \overline{R}_{AT}^{\geq}(X)\) denotes a boundary of the rough set. Moreover, one can easily obtain the following properties.

Let \(I^\geq = (U, AT, V, f)\) be an interval-valued ordered information system. For \(X, Y \subseteq U\), \(A \subseteq AT\) and \(R_{AT}^{\geq}\) a dominance relation, then following properties hold.

1. \(R_{AT}^{\geq}(\emptyset) = \overline{R}_{AT}^{\geq}(\emptyset) = \emptyset, \overline{R}_{AT}^{\geq}(U) = \overline{R}_{AT}^{\geq}(U) = U\);
Continued from Example 2.1. Consider the interval-valued ordered information system as Table 1. Let \( A = \{a_1, a_2, \ldots, a_5\} = AT \) and an object set \( X = D_2 = \{x_1, x_3, x_5, x_6, x_8\} \), compute the rough sets of \( D_2 \) approximated by \( U/R^{A_T}_A \). According to the definition of approximations and Example 2.1, the rough set \( \overline{R^{\tilde{A}}}_A(D_2) \) and \( \underline{R^{\tilde{A}}}_A(D_2) \) can be obtained as follows.

\[
\overline{R^{\tilde{A}}}_A(D_2) = \{x_5, x_8\}, \quad \underline{R^{\tilde{A}}}_A(D_2) = \{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}.
\]

So the boundary of the rough set is \( Bn_A(D_2) = \{x_1, x_3, x_4, x_6, x_7, x_9\} \).

3. Theories for incremental updating approximations in an IvOIS under the variation of objects

With the dynamic object set of interval-valued ordered information system, the structure of information granules in the information system may over time which leads to the change of knowledge induced by RST. In medical diagnosis the data does not usually remain a stable condition. Some objects will be deleted from the original information system with the patient’s cure or changing hospital and some objects will be inserted into the original information system with new patients arrive. So it is in other areas of science. Then the discovered knowledge may become invalid or some new implicit information may emerge in the whole updated information system. Rather than restarting from scratch by non-incremental or batch learning algorithm for each update, developing an efficient incremental algorithm to avoid unnecessary computations by utilizing the previous data structures or results are thus desired.

In this section, we investigate the variation of approximations of the dynamic IvOIS when the object set evolves over time while the attribute set remains constant. We assume the process for incremental update the approximations lasts two stages, namely, from time \( t \) to time \( t + 1 \). By considering the objects may enter into or get out of an information system at time \( t + 1 \) and we denote a dynamic IvOIS at time \( t \) as \( I^{2}_t = (U, AT \cup \{d\}, V, f) \), and at time \( t + 1 \) the original information system change into \( (I^{2}_{t+1}') = (U', AT \cup \{d\}, V', f) \) after insertion or deletion of objects. And we denote the decision classes and the \( A - \) dominating set as \( D_i \) and \( [x]^\sim_A \) respectively at time \( t \), which are denoted as \( D'_i \) and \( ([x]^\sim_A)' \) respectively at time \( t + 1 \). The lower and upper approximations of decision class \( D_i \) with
3.1. Theories for Incremental Computing Approximations with the Deletion of an Object in an IvOIS

Given an IvOIS $I^2 = (U, AT \cup \{d\}, V, f)$ at time $t$, the deletion of object $x^- \in U$ will change the original information system and information granules $[x]_A^\pm (x \in U$ and $A \subseteq AT)$ and the equivalence decision classes $D_i (i \in \{1, \ldots, r\})$. The approximations of $D_i$ will change accordingly. Here, we investigate the principles for updating approximations of $D_i$ as two cases: (1) The deleted object $x^-$ belongs to $D_i$. (2) The deleted object $x^-$ does not belong to $D_i$.

**Case 1.** The deleted object $x^-$ belongs to $D_i$, namely, $x^- \in D_i$.

**Proposition 3.1.** Let $I^2 = (U, AT \cup \{d\}, V, f)$ be an IvOIS and any $A \subseteq AT$. When $x^- \in D_i (i \in \{1, \ldots, r\})$ is deleted from $U$, we have the following properties about $R_A^\pm (D_i)$ and $R_A^\pm (D_i)'$.

(1) If $x^- \in R_A^- (D_i)$, then $R_A^- (D_i)' = R_A^- (D_i) - \{x^-\}$. Otherwise $R_A^- (D_i)' = R_A^- (D_i)$.

(2) $R_A^+ (D_i)' = (R_A^+ (D_i) - [x^-]_A^- \Delta^-_1) \cup \Delta^-_2$, where $\Delta^-_1 = \{x|x \in [x^-]_A^- \cap \Delta_2\}$ and $\Delta^-_2 = \cup_{x \in D_i - \{x^+\}} [x]_A^-$. 

**Proof:**

(1) If $x^- \in D_i$ is deleted from the universe $U$, we have $U = U - \{x^-\}$ and $D_i' = D_i - \{x^-\}$. So for any $x \in U'$, we have $([x]_A^+)' = [x]_A^+ - \{x^-\}$. If $[x]_A^+ \subseteq D_i$, then $([x]_A^+)' \subseteq D_i'$. It’s similar that if $[x]_A^- \not\subseteq D_i$ then $([x]_A^-)' \not\subseteq D_i$. So, from the definition of lower and upper approximations, we can get that for any $x \in U'$, if $x \in R_A^+ (D_i)$, then $x \in R_A^+ (D_i)'$ and if $x \not\in R_A^- (D_i)$ then $x \not\in R_A^+ (D_i)'$.

Hence, it is easy to obtain if $x^- \in R_A^- (D_i)$, then $R_A^- (D_i)' = R_A^- (D_i) - \{x^-\}$. Otherwise, the lower approximation of $D_i$ should be remain constant, i.e., $R_A^+ (D_i)' = R_A^+ (D_i)$.

(2) According to the definition, we have the $R_A^+ (D_i) = \cup_{x \in \Delta^-_1} [x]_A^- \cap D_i \neq \emptyset$. Thus when the object $x^- \in D_i$ is deleted from $U$, the $A$-dominating set $[x^-]_A^+$ should be removed from the upper approximation $R_A^+ (D_i)$, it’s mean $R_A^+ (D_i)' = R_A^+ (D_i) - [x^-]_A^- \Delta^-_1$. However, it may be exist $x \in D_i - \{x^-\}$ satisfies that $\Delta^-_1 = [x]_A^- \cap [x]_A^- \neq \emptyset$ and the object which $x \in [x]_A^-$, where $x \in (D_i - \{x^-\})$ should not be removed from $R_A^- (D_i)$. Therefore, we can obtain $R_A^- (D_i)' = (R_A^- (D_i) - [x^-]_A^- \Delta^-_1) \cup \Delta^-_2$, where $\Delta^-_2 = \cup_{x \in D_i - \{x^-\}} [x]_A^-$. Thus, the Proposition 3.1 is proved.

**Example 3.1.** *(Continued from Example 2.2)*. For Table 1, according to Proposition 3.1, we compute the lower approximation of $D_2$ by deleting $x_3$ and $x_5$, the upper approximation by deleting $x_6$ from $U$, where $D_2 = \{x_1, x_3, x_5, x_6, x_8\}$, respectively.
(1) Assume the object $x_3$ be deleted from Table 1, so $U' = U - \{x_3\}$. We can find the $x_3 \in D_2$ but $x_3 \notin R^\Delta_A(D_2)$. Therefore, $R^\Delta_A(D_2)' = \{x_5, x_8\}$. Let the object $x_5$ be deleted from Table 1, so $U' = U - \{x_5\}$. We have $x_5 \in D_2$ and $x_5 \in R^\Delta_A(D_2)$. Therefore, $R^\Delta_A(D_2)' = R^\Delta_A(D_2) - \{x_5\} = \{x_8\}$

(2) Let the object $x_6$ be deleted from Table 1, so $U' = U - \{x_6\}$. We have $x_6 \in D_2$, $\Delta^- = \{x_1, x_3, x_5, x_7, x_8, x_9\}$. Then $\Delta^- = \{x_8, x_9\}$ and $R^\Delta_A(D_2)' = (R^\Delta_A(D_2) - \{x_6\}) \cup \Delta^- = \{x_1, x_3, x_4, x_5, x_7, x_8, x_9\}$.

**Case 2.** The deleted object $x^-$ does not belongs to $D_i$, namely $x^- \notin D_i$.

**Proposition 3.2.** Let $I^\geq = (U, AT \cup \{d\}, V, f)$ be an IvOIS and any $A \subseteq AT$. When $x^- \notin D_i (i \in \{1, \ldots, r\})$ be deleted from $U$, we have the following properties about $R^\Delta_A(D_i)'$ and $\overline{R^\Delta_A(D_i)}$.

1. $R^\Delta_A(D_i)' = R^\Delta_A(D_i) \cup \Delta^-$. Where $\Delta^- = \{x|x \in (D_i - R^\Delta_A(D_i)), ([x]^\Delta_A' \subseteq D\}$, if $x^- \in [x]^\Delta_A$ then $([x]^\Delta_A') = [x]^\Delta_A - \{x^-\}$ otherwise $([x]^\Delta_A') = [x]^\Delta_A$.

2. If $x^- \in R^\Delta_A(D_i)$ then $R^\Delta_A(D_i)' = R^\Delta_A(D_i) - \{x^-\}$. Otherwise $R^\Delta_A(D_i)' = R^\Delta_A(D_i)$.

**Proof:**

(1) Based on definition, we have for any $x \in D_i$, if $x \in R^\Delta_A(D_i)$ then $[x]^\Delta_A \subseteq D_i$. When the object $x^- \notin D_i$ is deleted from the universe $U$, we have that $U' = U - \{x^-\}$ and $D_i = D_i$. So, $\forall x \in U'$, $([x]^\Delta_A') = [x]^\Delta_A - \{x^-\}$. It is easy to get that if $[x]^\Delta_A \subseteq D_i$ then $([x]^\Delta_A') \subseteq D_i$. Thus, $\forall x \in D_i \Rightarrow x \in R^\Delta_A(D_i)'$. On the other hand, for $\forall x \in D - R^\Delta_A(D_i)$, we can get $[x]^\Delta_A \subseteq D_i$. However, it may exist $x^- \in [x]^\Delta_A$ such that $([x]^\Delta_A') \subseteq D_i$ after the deletion of $x^-$. Then the $x$ should be added to $R^\Delta_A(D_i)'$, namely, $R^\Delta_A(D_i)' = R^\Delta_A(D_i) \cup \{x\}$. Therefore, we have $R^\Delta_A(D_i)' = R^\Delta_A(D_i) \cup \Delta^\Delta$.

(2) According to definition, we have $R^\Delta_A(D_i) = \bigcup\{[x]^\Delta_A|x \in D_i \cap D_i \neq \emptyset\}$. Since the deleted object $x^- \notin D_i$, there exists an object $x \in D_i$ stratifies $x^- \in [x]^\Delta_A$, if $x^- \in R^\Delta_A(D_i)$.

Thus, the proof is fulfilled. □

**Example 3.2.** (Continued from Example 2.2). For Table 1, according to Proposition 3.2, we compute the lower approximation of $D_2$ by deleting $x_{10}$ from $U$, the upper approximation of $D_2$ by deleting $x_9$ and $x_{10}$ from $U$.

(1) Assume the object $x_{10}$ be deleted from Table 1, so $U' = U - \{x_{10}\}$. We have $x_{10} \notin D_2$, $D_2 - R^\Delta_A(D_2) = \{x_1, x_3, x_6\}$, and $([x_1]_A^\Delta)', ([x_3]_A^\Delta)', ([x_6]_A^\Delta)' \not\subseteq D_2$. Therefore, $\Delta^- = \emptyset$ and $R^\Delta_A(D_2) = R^\Delta_A(D_2) \cup \Delta^- = \{x_5, x_8\}$.

(2) If the object $x_9$ be deleted from Table 1, then $U' = U - \{x_9\}$. We have $x_9 \notin D_2$ but $x_9 \in \overline{R^\Delta_A(D_2)}$. Therefore, $\overline{R^\Delta_A(D_2)'} = \overline{R^\Delta_A(D_2)} - \{x_9\} = \{x_1, x_3, x_4, x_5, x_6, x_7, x_8\}$. Let the object $x_{10}$ be deleted from Table 1, so $U' = U - \{x_{10}\}$. We have $x_{10} \notin D_2$ and $x_{10} \notin \overline{R^\Delta_A(D_2)}$. Therefore, $\overline{R^\Delta_A(D_2)'} = \overline{R^\Delta_A(D_2)} = \{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}$. 


3.2. Theories for Incremental Computing Approximations with the Insertion of a New Object in an IvOIS

Given an interval-valued ordered information system (IvOIS) \( I^\geq = (U, AT \cup \{d\}, V, f) \) at time \( t \), when the information system is updated by inserting a new object \( x^+ \) into the universe \( U \) at time \( t+1 \), where \( x^+ \) denotes the inserted object. There are two situations may occur: (1) \( x^+ \) forms a new decision class, namely, for any \( x \in U \), \( f(x, d) \neq f(x^+, d) \); (2) \( x^+ \) does not form a new decision class, namely, exist \( x \in U \), \( f(x, d) = f(x^+, d) \). The difference between the two situations is, in the first situation, in addition to updating the approximations of the equivalence classes, we need to compute the approximations for the new decision class. Firstly, for updating the approximations of the equivalence classes \( D_i \) where \( i \in \{1, \cdots, r\} \) when inserting an object \( x^+ \), we investigate the principles through two cases similar to the approach taken in the model of deletion: (1) The inserted object \( x^+ \) will belong to \( D_i \), it’s mean \( f(x, d) = f(x^+, d) \), where \( x \in D_i \); (2) The inserted object \( x^+ \) will not belong to \( D_i \), namely, \( f(x, d) \neq f(x^+, d) \), for any \( x \in D_i, i \in \{1, \cdots, r\} \).

To illustrate our incremental methods for updating approximations when inserting a new object \( x^+ \) into interval-valued ordered information system (IvOIS). We assume that the objects in Table 2 will be inserted into Table 1, the Table 2 are given as follows.

<table>
<thead>
<tr>
<th>( U )</th>
<th>( AST )</th>
<th>( LDH )</th>
<th>( \alpha - HBDH )</th>
<th>( CK )</th>
<th>( CKMB )</th>
<th>( d )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_{11} )</td>
<td>[10, 50]</td>
<td>[110, 250]</td>
<td>[115, 210]</td>
<td>[8, 195]</td>
<td>[5, 27]</td>
<td>2</td>
</tr>
<tr>
<td>( x_{12} )</td>
<td>[12, 45]</td>
<td>[105, 248]</td>
<td>[100, 210]</td>
<td>[7, 203]</td>
<td>[0, 23]</td>
<td>2</td>
</tr>
<tr>
<td>( x_{13} )</td>
<td>[10, 30]</td>
<td>[90, 200]</td>
<td>[96, 206]</td>
<td>[5, 195]</td>
<td>[3, 24]</td>
<td>2</td>
</tr>
<tr>
<td>( x_{14} )</td>
<td>[5, 30]</td>
<td>[60, 80]</td>
<td>[90, 160]</td>
<td>[0, 180]</td>
<td>[0, 10]</td>
<td>1</td>
</tr>
<tr>
<td>( x_{15} )</td>
<td>[30, 100]</td>
<td>[200, 600]</td>
<td>[100, 600]</td>
<td>[40, 800]</td>
<td>[10, 60]</td>
<td>4</td>
</tr>
</tbody>
</table>

**Case 1.** The inserted object \( x^+ \) will belong to \( D_i \), namely \( x^+ \in D_i, i \in \{1, \cdots, r\} \).

**Proposition 3.3.** Let \( I^\geq = (U, AT \cup \{d\}, V, f) \) be an IvOIS and any \( A \subseteq AT \). When the object \( x^+ \in D_i \) \( i \in \{1, \cdots, r\} \) be inserted into \( U \), we have the following properties about \( \overline{R^\geq_A(D_i)} \) and \( \overline{R^\geq_A(D_i)^{\prime}} \).

1. If \( [x^+]_{\overline{\mathcal{A}}} \subseteq D'_i \), where \( D'_i = D_i \cup \{x^+\} \) then \( \overline{R^\geq_A(D_i)^{\prime}} = \overline{R^\geq_A(D_i)} \cup \{x^+\} \). Otherwise, \( \overline{R^\geq_A(D_i)^{\prime}} = \overline{R^\geq_A(D_i)} \).

2. \( \overline{R^\geq_A(D_i)} = \overline{R^\geq_A(D_i)^{\prime}} = \overline{R^\geq_A(D_i)} \cup [x^+]_{\overline{\mathcal{A}}} \).

**Proof:**

1. According to definition, we have for any \( x \in D_i \), if \( [x]_{\overline{\mathcal{A}}} \subseteq D_i \), then \( x \in \overline{R^\geq_A(D_i)} \). Thus, when the object \( x^+ \) is inserted into \( U \), we have \( D'_i = D_i \cup \{x^+\} \). For any \( x \in D_i \), if \( x^+ \in [x]_{\overline{\mathcal{A}}} \) then \( ([x]_{\overline{\mathcal{A}}}') \subseteq D'_i \). That is, \( [x]_{\overline{\mathcal{A}}} \subseteq D_i \), then \( ([x]_{\overline{\mathcal{A}}}') \subseteq D'_i \). If \( [x]_{\overline{\mathcal{A}}} \not\subseteq D_i \), then \( ([x]_{\overline{\mathcal{A}}}') \not\subseteq D'_i \). Therefore, if \( [x]_{\overline{\mathcal{A}}} \subseteq D_i \), we have \( x^+ \in \overline{R^\geq_A(D_i)} \) and \( \overline{R^\geq_A(D_i)} = \overline{R^\geq_A(D_i)^{\prime}} \) and \( \overline{R^\geq_A(D_i)^{\prime}} = \overline{R^\geq_A(D_i)} \cup \{x^+\} \). Otherwise, \( \overline{R^\geq_A(D_i)} = \overline{R^\geq_A(D_i)^{\prime}} \).
(2) When the object $x^+$ be inserted into $U$ that $U' = U \cup \{x^+\}$. According to definition, we have $R^x_A(D_i)' = \bigcup_{x \in D_i} [x]_A\{x^+\}$. Since $D_i' = D_i \cup \{x^+\}$, then we have $R^x_A(D_i)' = R^x_A(D_i') \cup [x^+]_A$. Because for $\forall x \in U$ there $([x]_A)' = [x]_A \cup \{x^+\}$ or $([x]_A)' = [x]_A$ and the object $x^+ \in [x^+]_A$, we can obtain that $R^x_A(D_i)' = \bigcup_{x \in D_i} [x]_A \cup [x^+]_A = R^x_A(D_i) \cup [x^+]_A$.

Thus, the proof is finished. \hfill\(\square\)

Example 3.3. (Continued from Example 2.2). For Table 1, according to Proposition 3.3, we compute the lower approximation of $D_2$ when the objects $x_{11}$ and $x_{12}$ are inserted, the upper approximation of $D_2$ when the object $X_{13}$ in Table 2 inserted into the universe $U$, respectively.

(1) Let the object $x_{11}$ in Table 2 be inserted into Table 1, so the $U' = U \cup \{x_{11}\}$. Since $f(x_{11}, d) = 2$ then $D_2 = D_2 \cup \{x_{11}\}$. Because of $[x_{11}]_A = \{x_8, x_{11}\} \subseteq D_2$, we have $R^x_A(D_2) = R^x_A(D_2) \cup \{x_{11}\} = \{x_5, x_8, x_{11}\}$. Assume the object $x_{12}$ in Table 2 be inserted into Table 1 and $U' = U \cup \{x_{12}\}$. Since $f(x_{12}, d) = 2$ then $D_2 = D_2 \cup \{x_{12}\}$. Because of $[x_{12}]_A = \{x_3, x_9, x_{12}, x_{15}\} \not\subseteq D_2$ then we have $R^x_A(D_2)' = R^x_A(D_2) = \{x_5, x_8\}$.

(2) Assume the object $x_{13}$ in Table 2 be inserted into Table 1 and $U' = U \cup \{x_{13}\}$. Since $f(x_{13}, d) = 2$ then $D_2 = D_2 \cup \{x_{13}\}$ and $R^x_A(D_2)' = R^x_A(D_1) \cup [x_{13}]_A = \{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{13}\}$.

Case 2. The inserted object $x^+$ will not belong to $D_i$, namely, $x^+ \notin D_i, i \in \{1, \cdots, r\}$.

Proposition 3.4. Let $\mathcal{I} = (U, AT \cup \{d\}, V, f)$ be an IvOIS and any $A \subseteq AT$. When the object $x^+ \notin D_i (i \in \{1, \cdots, r\})$ be inserted into $U$, we have the following properties about $R^x_A(D_i)'$ and $R^x_A(D_i)$. If $\exists x \in D_i$ such that $x^+ \in [x]_A$, then $R^x_A(D_i)' = R^x_A(D_i) \cup \{x^+\}$. Otherwise $R^x_A(D_i)' = R^x_A(D_i)$.

Proof:

(1) When the object $x^+$ be inserted into $U$, since $f(x, d) = f(x^+, d)$ (x $\in D_i$) we have $U' = U \cup \{x^+\}$ and $D_i' = D_i$. For $\forall x \in D_i'$ there $([x]_A)' = [x]_A$ or $([x]_A)' = [x]_A \cup \{x^+\}$. We have if $[x]_A \not\subseteq D_i$ then $[x]_A \not\subseteq D_i'$. That is, if $x \notin R^x_A(D_i)$ then $x \notin R^x_A(D_i)'$. Hence, we only consider the object $x^+ \in R^x_A(D_i)$, namely, $D_i \subseteq [x]_A$. When the object $x^+$ be inserted into universe $U$, there may exist that $([x]_A)' = [x]_A \cup \{x^+\}$ and $([x]_A)'$ is not included by $D_i' = D_i$, namely, $x \notin R^x_A(D_i)'$. Therefore, we have $R^x_A(D_i)' = R^x_A(D_i) \cup \Delta_i^+$, where $\Delta_i^+ = \{x|x \in R^x_A(D_i), x^+ \in ([x]_A)'\}$.

(2) When the object $x^+$ is inserted into $U$, since $f(x, d) \neq f(x^+, d)$, we can obtain $U' = U \cup \{x^+\}$ and $D_i' = D_i$. Then, for $\forall x \in D_i'$, if $x^+ \in [x]_A$ then $([x]_A)' = [x]_A \cup \{x^+\}$. And we have $x^+ \in R^x_A(D_i)'$, that is, $R^x_A(D_i)' = R^x_A(D_i) \cup \{x^+\}$. Otherwise, if $\forall x \in D_i, x^+ \notin ([x]_A)'$, that is, $([x]_A)' = [x]_A$. Then we can get $R^x_A(D_i)' = R^x_A(D_i)$.

Thus, the proof is accomplished. \hfill\(\square\)

Example 3.4. (Continued from Example 2.2). For Table 1, according to Proposition 3.4, we compute the lower and upper approximations of $D_2$ when the object $x_{14}$ in Table 2 into the universe $U$. 

---

J. Yu, W. Xu / Incremental Computing Approximations with the Dynamic Object Set in IvOIS
(1) Assume the object $x_{14}$ in Table 2 insert into Table 1, and $U = U \cup \{x_{14}\}$. Since $f(x_{14}, d) = 1$ then $D_2' = D_2$ remain unchanged. Because of $\overline{R^2_A(D_2)} = \{x_5, x_8\}$, and $x_{14} \notin ([x_5]_A, ([x_8]_A)'$. Hence, we have $\Delta^+_1 = 0$ then $R^2_A(D_2)' = R^2_A(D_2) = \{x_5, x_8\}$.

(2) Let the object $x_{14}$ in Table 2 insert into Table 1 and $U = U \cup \{x_{14}\}$. Since $f(x_{14}, d) = 1$, then $D_2' = D_2$ remain unchanged. Because of $\overline{R^2_A(D_2)} = \{x_1, x_3, x_5, x_6, x_7, x_8, x_9\}$, and $x_{15} \in ([x_2]_A)'$, that is $x_{15} \in ([x_2]_A)'$. Hence, we have $R^2_A(D_2)' = R^2_A(D_2) \cup \{x_{15}\} = \{x_1, x_3, x_5, x_6, x_7, x_8, x_9, x_{15}\}$.

Based on above investigate, we can compute the lower and upper approximations of the existed equivalence decision classes $D_i$ where $i = 1, \cdots, r$ when inserting a new object into IvOIS. However, when a new object $x^+$ is inserted into the universe $U$, it might happen that $x^+$ will generate a new decision class, namely, $\forall x \in U$, $f(x, d) \neq f(x^+, d)$. Then the universe $U' = U \cup \{x^+\}$ will be divided into $r + 1$ partitions and $D_{r+1} = \{x^+\}$. At this point, in addition to updating the approximations of new decision class $D_{r+1}$.

**Proposition 3.5.** Let $I^2 = (U, AT \cup \{d\}, V, f)$ be an IvOIS and any $A \subseteq AT$. When the object $x^+$ be inserted into $U$, if for $\forall x \in U$, $f(x, d) \neq f(x^+, d)$, then the lower approximation of the new decision class $D_{r+1}$ can be computed by definition be shown as follows.

(1) If there $[x^+] \subseteq D_{r+1}$, where $D_{r+1} = \{x^+\}$, then $\overline{R^2_A(D_{r+1})} = [x^+]_A$. Otherwise $\overline{R^2_A(D_{r+1})} = \emptyset$.

(2) $\overline{R^2_A(D_{r+1})} = \bigcup \{[x]_A | x^+ \in [x]_A, x \in U\}$.

**Proof:**
It’s easy to prove according to the definition of approximations. 

**Example 3.5.** (Continued from Example 2.2). For Table 1, according to Proposition 3.5, we compute the lower and upper approximations of $D_{r+1}$ when the object $x_{15}$ in Table 2 be inserted into the universe $U$.

(1) Assume the object $x_{15}$ be inserted into universe $U$ in Table 2 then $U = U \cup \{x_{15}\}$. Since for $\forall x \in U$, $f(x, d) \neq f(x_{15}, d) = 4$ then $U/d = \{d_1, \cdots, d_{r+1}\} \cup \{x_{15}\}$ and $D_{r+1} = \{x_{15}\}$. Because of $[x]_A = \{x_{15}\}$, we have $\overline{R^2_A(D_{r+1})} = \{x_{15}\}$.

(2) Let the object $x_{15}$ in Table 2 insert into Table 1 and $U = U \cup \{x_{15}\}$. Since for $\forall x \in U$, $f(x, d) \neq f(x_{15}, d) = 4$, then $U/d = \{d_1, \cdots, d_{r+1}\}$ and $D_{r+1} = \{x_{15}\}$. Because of $[x]_A \subseteq \{x_{15}\}$, we have $\overline{R^2_A(D_{r+1})} = [x]_A \cup \overline{R^2_A(D_{r+1})}' = U'$.

4. **Non-incremental and incremental algorithms for computing approximations in an IvOIS with the dynamic object set**

In this section, we design the non-incremental and incremental algorithms on the variation of the object set in an IvOIS. Sometimes we call the non-incremental algorithm as statical algorithm or traditional algorithm.
4.1. The Non-incremental Algorithm for Computing Approximations in an IvOIS

The given Algorithm 1 is a statical(non-incremental) algorithm for computing the lower and upper approximations in an IvOIS when the object set in the information system is changed. First, we compute all the decision classes $U/d = \{D_1, D_2, \cdots, D_r\}$. Later, initialize all lower and upper approximations as empty set for every $D_i$, $i = 1, \cdots, r$. The step 4-5 compute all the $A-$dominating sets. Step 6-15 compute the lower and upper approximations in IvOIS based on the Dentition 2.2. At last, return the results. The computational complexity of Algorithm 1, as shown in Table 3.

Algorithm 1: An non-incremental algorithm for updating approximations in an IvOIS

<table>
<thead>
<tr>
<th>Step</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$O(</td>
</tr>
<tr>
<td>2-3</td>
<td>$O(\sum_{i=1}^{r}</td>
</tr>
<tr>
<td>4-5</td>
<td>$O(</td>
</tr>
<tr>
<td>6-15</td>
<td>$O(\sum_{i=1}^{r}</td>
</tr>
<tr>
<td>Total</td>
<td>$O(2</td>
</tr>
</tbody>
</table>

4.2. The Incremental Algorithm for Updating Approximations in an IvOIS when Deleting an Object from the Universe

The given Algorithm 2 is an incremental algorithm for updating the lower and upper approximations in an IvOIS when the object set be deleted from the universe $U$ in the interval-valued ordered informa-
tion system. Step 3-16 update the lower and upper approximations of the decision classed $D_i$, when the deleted object $x^-$ belongs to the decision classes. Among them, the step 4-8 update the lower approximations of $D_i$ by Proposition 3.1, step 9-16 update the upper approximations of $D_i$ by Proposition 3.1. Step 18-32 update the approximations of the decision classes $D_i$, where the deleted object $x^-$ does not belong to the decision classes $D_i$. Among them, the step 18-25 compute the lower approximations of $D_i$ by Proposition 3.2, step 26-32 compute the upper approximations of $D_i$ by Proposition 3.2. At last, return the result of approximations after deleting the object $x^-$. The computational complexity of Algorithm 2, as shown in Table 4. The flow-process diagram of Algorithm 2 as shown in Fig. 1.

Table 4. The computational complexity of Algorithm 2

<table>
<thead>
<tr>
<th>Step</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-9</td>
<td>$O(</td>
</tr>
<tr>
<td>10-16</td>
<td>$O(</td>
</tr>
<tr>
<td>18-25</td>
<td>$O(</td>
</tr>
<tr>
<td>26-30</td>
<td>$O(</td>
</tr>
<tr>
<td>Total</td>
<td>$O(\sum_{i=1}^r</td>
</tr>
</tbody>
</table>

![Flow-process diagram of Algorithm 2](image_url)

Fig. 1. The flow-process diagram of Algorithm 2.
Algorithm 2: An incremental algorithm for updating approximations in an IvOIS when deleting an object from the universe

\textbf{Input}:
(1) The original interval-valued ordered information system at time $t$: $I^x = (U, AT \cap \{d\}, V, f)$, where $A \subseteq AT$;
(2) The $A$–dominating sets $[x]_A^x$ at time $t$ for each $x \in U$ where $A \subseteq AT$ and the original decision equivalence classes $U/d = \{D_1, D_2, \cdots, D_r\}$, the $r$ is the number of the decision classes;
(3) The original lower and upper approximations at time $t$: $\underline{R}_A^x(D_i), \overline{R}_A^x(D_i), i = 1, \cdots, r$;
(4) The object will be deleted from $U$: $x^-$.

\textbf{Output}: The lower and upper approximations in an IvOIS at time $t + 1$ after deletion of $x^-$ from $U$: $\underline{R}_A^x(D_i)', \overline{R}_A^x(D_i)'$.

\begin{algorithm}
\begin{algorithmic}[1]
\State \textbf{begin}
\For{$i = 1, \cdots, r$}
\If{$x^- \notin D_i$}
\If{$x^- \in \overline{R}_A^x(D_i)$}
\State $\overline{R}_A^x(D_i)' = \overline{R}_A^x(D_i) - \{x^-\}$; // update the lower approximation by Proposition 3.1;
\Else
\State $\overline{R}_A^x(D_i)' = \overline{R}_A^x(D_i)$;
\EndIf
\EndIf
\EndIf
\For{$\forall x \in [x^-]_A^x$}
\If{$x \notin [x^-]_A^x$}
\State $\overline{R}_A^x(D_i)' = \overline{R}_A^x(D_i)' \cup \{x\}$; //update the upper approximation by Proposition 3.1;
\EndIf
\EndFor
\Else
\State $\underline{R}_A^x(D_i)' = \underline{R}_A^x(D_i)$; // update the lower approximation by Proposition 3.2;
\For{$\forall x \in (D_i) - \overline{R}_A^x(D_i)$}
\If{$x^- \notin [x]_A^x$}
\State $[x]_A^x = [x]_A^x - \{x^-\}$;
\EndIf
\If{$[x]_A^x \subseteq D_i$}
\State $\underline{R}_A^x(D_i)' = \underline{R}_A^x(D_i)' \cup \{x\}$;
\EndIf
\Else
\If{$x^- \in \overline{R}_A^x(D_i)$}
\State $\overline{R}_A^x(D_i)' = \overline{R}_A^x(D_i) - \{x^-\}$; //update the upper approximation by Proposition 3.2;
\Else
\State $\overline{R}_A^x(D_i)' = \overline{R}_A^x(D_i)$;
\EndIf
\EndIf
\EndFor
\EndIf
\EndFor
\Return $\underline{R}_A^x(D_i)', \overline{R}_A^x(D_i)'$.
\end{algorithmic}
\end{algorithm}
4.3. The Incremental Algorithm for Updating Approximations in an IvOIS when Inserting an Object into the Universe

The given Algorithm 3 is an incremental algorithm for updating the lower and upper approximations in an IvOIS when the object set is inserted into the universe $U$ in the information system. First, we should compute the $A$-dominating set with respect to $x^+$ is $[x^+]_A$. Step 2-22 update the approximations of decision classes $D_i$, when the inserted object $x^+$ will belong to the decision classes $D_i$. Step 5-9 compute the lower approximations of $D_i$ by Proposition 3.3. Step 10 compute the upper approximations of $D_i$ by Proposition 3.3. Step 11-22 update the approximations of the decision classes $D_i$, when the inserted object $x^+$ will not belong to the decision classes $D_i$. Step 11-16 compute the lower approximations of $D_i$ by Proposition 3.4. Step 17-22 update the approximations of $D_i$ by Proposition 3.4. Step 23-31 compute the approximation of new decision class by Proposition 3.5. Step 23-29 compute the lower approximation of $D_{r+1}$ by Proposition 3.5. Step 30 compute the upper approximation of $D_{r+1}$ by Proposition 3.5. At last, return the approximations after inserting object $x^+$. The computational complexity of Algorithm 3, as shown in Table 5. The flow-process diagram of Algorithm 3 as shown in Fig. 2.

Fig. 2. The flow-process diagram of Algorithm 3.

<table>
<thead>
<tr>
<th>Step</th>
<th>Computational Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$O(</td>
</tr>
<tr>
<td>5-9</td>
<td>$O(</td>
</tr>
<tr>
<td>12-16</td>
<td>$O(</td>
</tr>
<tr>
<td>17-22</td>
<td>$O(</td>
</tr>
<tr>
<td>23-30</td>
<td>$O(</td>
</tr>
<tr>
<td>Total</td>
<td>$O(</td>
</tr>
</tbody>
</table>
Algorithm 3: An incremental algorithm for updating approximations in an IvOIS when inserting an object into the universe $U$.

**Input**: 
1. The original interval-valued ordered information system at time $t : I^U = (U, AT \cup \{d\}, V, f)$; 
2. The $A$–dominating sets $[x_A]_A^2$ at time $t$ for each $x \in U$ where $A \subseteq AT$ and the original decision equivalence classes $U/d = \{D_1, D_2, \ldots, D_r\}$, the $r$ is the number of the decision classes; 
3. The original lower and upper approximations at time $t : R_A^U(D_i), R_A^U(D_i)\bar{\prime}$, $i = 1, \ldots, r$; 
4. The object will be inserted into $U$: $x^+$.

**Output**: The lower and upper approximations in an IvOIS at time $t + 1$ after the insertion of $x^+$ into $U$: $R_A^U(D_i)\prime + 1$, $R_A^U(D_i)\prime + 1$.

```
begin
compute: the $A$–dominating set with respect to $x^+$: $[x_A]_A^2$;
for $i = 1, \ldots, r$ do
if $x^+ \in D_i$ then
  $D_i = D_i \cup \{x^+\}$;
if $[x^+]_A^2 \subseteq D_i$ then
  $R_A^U(D_i)\prime = R_A^U(D_i)\prime + 1 \cup \{x^+\}$;  // update the lower approximation by Proposition 3.3;
else
  $R_A^U(D_i)\prime = R_A^U(D_i)\prime$;
end
end
else
let : $R_A^U(D_i)\prime - 1 = R_A^U(D_i)\prime$;
for each $x \in R_A^U(D_i)\prime$ do
  if $x^+ \in [x_A]_A^2$ then
    $R_A^U(D_i)\prime = R_A^U(D_i)\prime - \{x\}$;  // update the lower approximation by Proposition 3.4;
end
end
for each $x \in D_i$ do
  if $x^+ \in [x_A]_A^2$ then
    $R_A^U(D_i)\prime = R_A^U(D_i)\prime \cup \{x^+\}$;  // update the upper approximation by Proposition 3.4;
end
end
if $\forall x \in U, f(x, d) \neq f(x^+, d)$ then
generate: a new decision class $D_{r+1}$;
if $[x^+]_A^2 \subseteq D_{r+1}$ then
  $R_A^U(D_{r+1}) = \{x^+\}$;  // update the lower approximation by Proposition 3.5;
else
  $R_A^U(D_{r+1}) = \emptyset$;
end
end
$R_A^U(D_{r+1}) = \cup\{[x_A]_A^2 | x^+ \in [x_A]_A^2, x \in U\}$;  // update the upper approximation by Proposition 3.5;
return : $R_A^U(D_i)\prime + 1, R_A^U(D_i)\prime + 1, R_A^U(D_{r+1})$;
end
```
5. Case study

In this section, in order to evaluate the performance of the proposed incremental algorithms, we conduct a series of experiments to compare the computational time between the non-incremental (statistical) algorithm and incremental algorithms for computing approximations based on standard data sets where from the UC Irvine Machine Learning Database Repository (http://archive.ics.uci.edu/ml/datasets.html), named “Energy efficiency”, “Airfoil Self-Noise”, “Wine Quality-red”, “Wine Quality-white”, “Letter Recognition”, “Spoken Arabic Digit” and the characteristics of the data sets are summarized in Table 7. This experimental computing program is running on a personal computer with following hardware and software as Table 6.

![Table 6. Experiment platform.](image)

<table>
<thead>
<tr>
<th>Name</th>
<th>Model</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel i3-370</td>
<td>2.40GHz</td>
</tr>
<tr>
<td>Memory</td>
<td>Samsung DDR3</td>
<td>2GB, 1067MHz</td>
</tr>
<tr>
<td>HardDisk</td>
<td>West Data</td>
<td>500GB</td>
</tr>
<tr>
<td>System</td>
<td>Windows7</td>
<td>32bit</td>
</tr>
<tr>
<td>Platform</td>
<td>VC++</td>
<td>6.0</td>
</tr>
</tbody>
</table>

Note that, the attributes characteristics of the six datasets in Table 7 consist of integer or real number. We construct the interval-valued information tables by utilizing multiply error precision $\alpha$, namely, the attribute value of $\forall x_i \in U$, at $\forall a_j \in AT$ is $V_{x_i,a_j}$, we can let it express as $[(1-\alpha) \times V_{x_i,a_j}, (1+\alpha) \times V_{x_i,a_j}]$ then we can utilize the interval-valued information tables in our experiments. In different engineering areas may use different error precision. In this paper, we set the error precision $\alpha = 0.05$.

![Table 7. Experiment data sets.](image)

<table>
<thead>
<tr>
<th>No.</th>
<th>Data set name</th>
<th>Abbreviation</th>
<th>Objects</th>
<th>Attributes</th>
<th>Decision classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Energy efficiency</td>
<td>EE</td>
<td>768</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Airfoil Self – Noise</td>
<td>AS</td>
<td>1503</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Wine Quality – red</td>
<td>WQ-r</td>
<td>1599</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>Wine Quality – white</td>
<td>WQ-w</td>
<td>4898</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>Letter Recognition</td>
<td>LR</td>
<td>8084</td>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>6</td>
<td>Spoken ArabicDigit</td>
<td>SAD</td>
<td>8800</td>
<td>13</td>
<td>3</td>
</tr>
</tbody>
</table>

Before the experiment let the original data in each dataset are equally divided into twenty parts. We let the original data set as the training data at time $t$, and randomly choose serval parts (from 5% to 50%) as the immigrating objects which will be deleted from the system at time $t + 1$. Another experiment, we choose the 80% as the training data set at time $t$, and the remaining 20% as the test data set which will be inserted into the system at $t + 1$. Each test choose a part enter into the system (from 10% to 100% of the test data) at time $t + 1$. 

![image]
Generally, we perform the experimental analysis with applying the non-incremental algorithm along with our proposed incremental algorithms when the objects inserting into or deleting from the information system, respectively. The size of updated objects which inserting into or deleting from the universe should be different, namely, updated ratio, that is, the ration of the numbers of updating data and original data. Here, in order to analyze the influence of the updated ratio on the efficiency of algorithms, we compare the computational time of the non-incremental and incremental algorithms with different updated ratios. It’s mean for each data sets, we conduct the comparison experiments with same original data size, but different updated ratios included deleting ratios and inserting ratios.

5.1. A Comparison of Computational Efficiency Between Non-incremental and Incremental Algorithm with the Deletion of the Object Set

To compare the efficiency of non-incremental algorithm (Algorithm 1) and incremental algorithm (Algorithm 2) for computing lower and upper approximations when deleting the objects from the data sets. We compute the time of the two algorithms on the given datasets in Table 7 with the different updating ratio (from 10% to 100%), but same sizes of the original data, we show the experimental results in Table 8. And more detailed changing trendline of each of two algorithms with the increasing updating ratio of data sets are presented in Fig. 3.

Table 8. A comparison of non-incremental and incremental algorithm versus different updating rates when deleting objects.

<table>
<thead>
<tr>
<th>Del.(%)</th>
<th>EE</th>
<th>AS</th>
<th>WQ-r</th>
<th>WQ-w</th>
<th>LR</th>
<th>SAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>5%</td>
<td>0.092</td>
<td>0.001</td>
<td>0.423</td>
<td>0.001</td>
<td>1.077</td>
<td>0.030</td>
</tr>
<tr>
<td>10%</td>
<td>0.094</td>
<td>0.001</td>
<td>0.381</td>
<td>0.001</td>
<td>0.981</td>
<td>0.048</td>
</tr>
<tr>
<td>15%</td>
<td>0.062</td>
<td>0.001</td>
<td>0.375</td>
<td>0.048</td>
<td>0.888</td>
<td>0.096</td>
</tr>
<tr>
<td>20%</td>
<td>0.064</td>
<td>0.020</td>
<td>0.327</td>
<td>0.045</td>
<td>0.795</td>
<td>0.093</td>
</tr>
<tr>
<td>25%</td>
<td>0.062</td>
<td>0.024</td>
<td>0.279</td>
<td>0.045</td>
<td>0.654</td>
<td>0.093</td>
</tr>
<tr>
<td>30%</td>
<td>0.056</td>
<td>0.028</td>
<td>0.234</td>
<td>0.048</td>
<td>0.624</td>
<td>0.141</td>
</tr>
<tr>
<td>35%</td>
<td>0.052</td>
<td>0.026</td>
<td>0.195</td>
<td>0.048</td>
<td>0.516</td>
<td>0.141</td>
</tr>
<tr>
<td>40%</td>
<td>0.046</td>
<td>0.028</td>
<td>0.189</td>
<td>0.093</td>
<td>0.420</td>
<td>0.186</td>
</tr>
<tr>
<td>45%</td>
<td>0.036</td>
<td>0.030</td>
<td>0.141</td>
<td>0.096</td>
<td>0.375</td>
<td>0.234</td>
</tr>
<tr>
<td>50%</td>
<td>0.032</td>
<td>0.030</td>
<td>0.141</td>
<td>0.093</td>
<td>0.327</td>
<td>0.282</td>
</tr>
</tbody>
</table>

In each sub-figure(a)-(f) of Fig. 3, the $x$–coordinate pertains to the ratio of the numbers of the deleting data and original data, while the $y$–coordinate concerns the computational time. According to the experimental results in Table 8 and Fig. 3, we can see, for the non-incremental algorithm, the computational time for computing approximations with deletion of the objects from the universe $U$ is decreasing monotonically along with the increase of ratios, the size of the universe $U$ decrease gradually. On the contrary, for the incremental algorithm, we can see that the computational efficiency for computing approximations is changing smoothly along with the increase of deleting ratios.
Fig. 3. A comparison of non-incremental (Algorithm 1) and incremental (Algorithm 2) algorithms versus different updating rates when deleting objects.
It’s easy to get the incremental algorithm always performs faster than the non-incremental algorithm for computing approximations. It must be noted that there is a threshold depending on the data set. Different data sets have different thresholds. Once the delete ratio over the threshold, namely, the deleted data set is bigger than the remaining data set maybe the incremental algorithm is slower than the non-incremental. So, the incremental algorithm is very efficient especially when need to delete the data set is far smaller than the original data set. Data set is larger when stronger regularity.

5.2. A Comparison of Computational Efficiency Between Non-incremental and Incremental Algorithm with the Insertion of the Object Set

Similar to the experiment schemes for comparing the efficiencies between non-incremental and incremental algorithms when deleting the objects from the universe \( U \), we also adopt such schemes to compare the performance of algorithms on the case of inserting the objects into the universe \( U \). We compute the two algorithms (Algorithm 1 and Algorithm 3) on the six UCI data sets in Table 7 with the changing of updating ratios for each data sets. The experimental results are shown in Table 6. More detailed change trend line of each two algorithms with the increasing ratio of data sets are given in Fig. 4.

Table 9. A comparison of non-incremental and incremental algorithm versus different updating rates when inserting objects.

<table>
<thead>
<tr>
<th>Ins. (%)</th>
<th>EE</th>
<th>AS</th>
<th>WQ-r</th>
<th>WQ-w</th>
<th>LR</th>
<th>SAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>0.090</td>
<td>0.001</td>
<td>0.327</td>
<td>0.001</td>
<td>0.795</td>
<td>0.001</td>
</tr>
<tr>
<td>20%</td>
<td>0.094</td>
<td>0.001</td>
<td>0.327</td>
<td>0.001</td>
<td>0.843</td>
<td>0.001</td>
</tr>
<tr>
<td>30%</td>
<td>0.092</td>
<td>0.001</td>
<td>0.327</td>
<td>0.001</td>
<td>0.891</td>
<td>0.001</td>
</tr>
<tr>
<td>40%</td>
<td>0.094</td>
<td>0.001</td>
<td>0.375</td>
<td>0.024</td>
<td>0.936</td>
<td>0.030</td>
</tr>
<tr>
<td>50%</td>
<td>0.092</td>
<td>0.003</td>
<td>0.375</td>
<td>0.045</td>
<td>0.984</td>
<td>0.048</td>
</tr>
<tr>
<td>60%</td>
<td>0.092</td>
<td>0.004</td>
<td>0.420</td>
<td>0.048</td>
<td>1.029</td>
<td>0.045</td>
</tr>
<tr>
<td>70%</td>
<td>0.094</td>
<td>0.004</td>
<td>0.420</td>
<td>0.045</td>
<td>1.029</td>
<td>0.048</td>
</tr>
<tr>
<td>80%</td>
<td>0.094</td>
<td>0.012</td>
<td>0.468</td>
<td>0.045</td>
<td>1.077</td>
<td>0.048</td>
</tr>
<tr>
<td>90%</td>
<td>0.094</td>
<td>0.012</td>
<td>0.468</td>
<td>0.048</td>
<td>1.125</td>
<td>0.054</td>
</tr>
<tr>
<td>100%</td>
<td>0.094</td>
<td>0.012</td>
<td>0.468</td>
<td>0.048</td>
<td>1.218</td>
<td>0.048</td>
</tr>
</tbody>
</table>

In each sub-figure(a)-(f) of Fig. 4, the \( x \)-coordinate pertains to the ratio of the numbers of the inserted objects and test data, while the \( y \)-coordinate concerns the computational time. According to the experimental results in Table 9 and Fig. 4, we can see, for the non-incremental algorithm, the computational time for computing approximations with insertion of the objects into the universe \( U \) is increasing monotonically along with the increase of ratios. On the contrary, for the incremental algorithm, we can see that the computational efficiency for computing approximations is changing smoothly along with the increase of inserting ratios. It’s easy to get the incremental algorithm always performs faster than the non-incremental algorithm for computing approximations. So, the incremental algorithm is efficiency when the objects insert into the universe, especially the original data set is an big data set and when the changing data set relatively small is very efficiency.
Fig. 4. A comparison of non-incremental (Algorithm 1) and incremental (Algorithm 3) algorithm versus different updating rates when inserting objects.
6. Conclusions

The incremental technique is an very effective approach to maintain knowledge in the dynamic environment. In this paper, we proposed incremental methods for updating lower and upper approximations in an IvOIS when the information system is updated by inserting or deleting object set, respectively. And two algorithms as for updating approximations when the information system is updated by inserting or deleting object set. Experimental studies pertaining to six UCI data sets showed that the incremental algorithms can improve the computational efficiency for updating approximations when the object in the information system varies over time. In real-world application, an interval-valued information system may be updated by changing granularity or attributes or all of the elements in the information system will change as time goes by under the dynamic environment. In the future, the variation of attributes and the domain of attributes values in an IvOIS or the granulations’ coarsening and refinement will also be taken into consideration in terms of incremental updating approximations and knowledge discovery.

Acknowledgements

This work is supported by Natural Science Foundation of China (No. 61472463 and No. 61402064), National Natural Science Foundation of CQ CSTC (No. cstc 2013jcyjA40051), Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information (Nanjing University of Science and Technology), Ministry of Education (No. 30920140122006), Graduate Innovation Foundation of Chongqing University of Technology (No.YCX2014236), and Graduate Innovation Foundation of CQ (No.CYS15223).

References


