International Journal of Approximate Reasoning 55 (2014) 1764–1786

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

A rough set-based incremental approach for learning knowledge in dynamic incomplete information systems

Dun Liu [a,*], Tianrui Li [b], Junbo Zhang [b]

[a] School of Economics and Management, Southwest Jiaotong University, Chengdu 610031, China
[b] School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China

ARTICLE INFO

Article history:
Received 5 January 2013
Received in revised form 28 February 2014
Accepted 27 May 2014
Available online 2 June 2014

Keywords:
Rough set theory
Incomplete information systems
Incremental learning
Knowledge discovery

ABSTRACT

With the rapid growth of data sets nowadays, the object sets in an information system may evolve in time when new information arrives. In order to deal with the missing data and incomplete information in real decision problems, this paper presents a matrix based incremental approach in dynamic incomplete information systems. Three matrices (support matrix, accuracy matrix and coverage matrix) under four different extended relations (tolerance relation, similarity relation, limited tolerance relation and characteristic relation), are introduced to incomplete information systems for inducing knowledge dynamically. An illustration shows the procedure of the proposed method for knowledge updating. Extensive experimental evaluations on nine UCI datasets and a big dataset with millions of records validate the feasibility of our proposed approach.

1. Introduction

Data mining (also known as knowledge discovery in databases) is the process of extracting patterns from large data sets to analyze data from different perspectives and summarize them into useful information. Many data mining approaches, such as association rule mining [27], sequential pattern mining [25], text mining [32] and temporal data mining [40], are utilized to discover potentially valuable patterns, associations, trends, sequences and dependencies [26]. However, with the fast growth of data sets in real-life applications, it brings a big challenge to quickly acquire the useful information with dynamic data mining techniques.

As an efficient data analysis' technique, the rough set based incremental approaches have become one of the hot topics on extraction of knowledge from the changing data sets in recent decades. 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 (items, records or instances), data attributes (features) and data attribute values [50] recent researches mainly focus on the three types of its variations, namely, variation of objects [1,8,23,36,42–44, 52,68,71,80,82,83], variation of attributes [6,9,37–39,47,53,81,85] and variation of attributes' values [7,45,46,48]. The second one is based on the view of pre-topology [57]. The classification of dynamics in rough sets is divided into two aspects: synchronic dynamics (knowledge evolves in time) and diachronic dynamics (changes from one point of view to another) [10]. Furthermore, Ciucci [10,13] listed four main streamlines to investigate dynamics in rough sets, namely, lower and

* Corresponding author.
E-mail addresses: newton83@163.com (D. Liu), trli@swjtu.cn (T. Li), JunboZhang86@163.com (J. Zhang).

http://dx.doi.org/10.1016/j.ijar.2014.05.009
0888-613X/

upper approximations [4,11,63,64,80–82], reducts and rules [12,24,65,77], quality indexes [20,21,42–44] and formal logic [29,56,66]. To sum up, both viewpoints provide a basic and clear framework on dynamic studies of rough sets.

However, many databases with thousands of items in real-life applications lead to lots of challenges. The variation of objects significantly affects the knowledge updating. Then, a series of effective algorithms on the incremental learning of approximations, reducts and rules were proposed for knowledge updating to improve the computational efficiency. Shan and Ziarko presented a discernibility-matrix based incremental methodology to find all maximally generalized rules [68]. Bang and Bien proposed another incremental inductive learning algorithm to find a minimal set of rules for a decision table without recalculating all the set of instances when another instance 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 [71]. Zheng and Wang developed a rough set and rule tree based incremental knowledge acquisition algorithm, RRIA, to update knowledge more quickly when new objects are added or removed from a given dataset [85]. Hu et al. constructed a novel incremental attribute reduction algorithm when new objects are added into a decision information system [24]. Blaszczynski and Slowinski discussed the incremental induction of decision rules from dominance-based rough approximations to select the most interesting representatives in the final set of rules [3]. Fan et al. proposed an approach of incremental rule induction based on rough sets [14]. In addition, Liu et al. proposed an incremental approach as well as its algorithm for inducing interesting knowledge when objects change over time [42,43]. Then, Liu et al. further introduced the incremental matrix and presented a new optimization approach for knowledge discovery [44]. Followed by Liu's work, Li et al. proposed an incremental approach for updating approximations in dominance-based rough sets [36]. Zhang et al. proposed a method for dynamic data mining based on neighborhood rough sets [80], and they further presented a parallel method for computing rough set approximations [82,84]. In our studies, we mainly consider the missing data in real databases, and the rough set-based incremental approach for inducing knowledge in incomplete information systems (IIS) is carefully investigated.

This paper is to focus on the dynamic knowledge discovery based on rough set theory with the variation of the object set in IIS. The rest of the paper is organized as follows. We provide the basic concepts of rough sets under IIS in Section 2. We introduce the support matrix, the accuracy matrix and the coverage matrix to illuminate the incremental approach, the related updating strategies and algorithms for learning knowledge in IIS when objects change is given in Section 3. An example is employed to illustrate the proposed model and experimental results on nine UCI datasets and a big dataset with millions of records are presented in Section 4. The paper ends with conclusions and further research topics in Section 5.

2. Preliminaries

In this section, we first briefly review the concepts of rough sets as well as their extensions in IIS from [17–19,30,31,34,35,41,49,51,58,59,61–64,69,70,75]. Then, we introduce some necessary concepts of knowledge discovery from [5,15,42–44].

2.1. Pawlak rough set model

For an approximation space $\mathcal{K} = (U, R)$, let U be a finite and non-empty set called the universe and $R \subseteq U \times U$ a binary relation on U. $\mathcal{K} = (U, R)$ is characterized by an information system $S = (U, A, V, f)$. S is called a decision table with $A = C \cup D$ and $C \cap D = \emptyset$, where C denotes the condition attribute set and D denotes the decision attribute set. $V = \bigcup_{a \in A} V_a$, V_a is a domain of the attribute a. $f : U \times A \to V$ is an information function such that $f(x, a) \in V_a$ for every $x \in U$, $a \in A$.

Each non-empty subset $B \subseteq C$ determines a binary indiscernibility relation R_B as follows.

$$R_B = \big\{(x, y) \in U \times U \bigm| f(x, a) = f(y, a),\ \forall a \in B\big\}.$$

R_B is an equivalence relation. It constitutes a partition of U, denoted as U/B. The equivalence relation R_B divides U into several disjoint subsets named equivalence classes given by: $U/R_B = \{[x]_B \mid x \in U\}$. Suppose there are two elements $x, y \in U (x \neq y)$ indistinguishable under R_B, we say x and y belong to the same equivalence class. The equivalence class including x is denoted as $[x]_{R_B}$, where $[x]_{R_B} = \{y \in U \mid (x, y) \in R_B\}$. For simplicity, U/R_B is denoted as U/B.

Pawlak rough set model is based on the equivalence relation. The elements in an equivalence class satisfy reflexive, symmetric and transitive properties. It also does not allow the missing data and requires the information table should be complete. However, missing data appears frequently in real-life applications, *e.g.*, in the survey sampling, it may arise out of poorly designed questionnaires, non-response by an interviewer, or errors made by the interviewer [19]. Lipski discussed the semantic issues connected with incomplete information databases [41]. Ibrahim et al. did a comparative review on four missing-data methods for generalized linear models [28]. Leung et al. dealt with knowledge acquisition in incomplete information systems using rough set theory [33]. Saar-Tsechansky et al. investigated three methods for applying classification trees to instances with missing values [67]. Therefore, many extended models of Pawlak rough sets were presented to deal with missing data in IIS.

2.2. Extended rough set models

The early extension of Pawlak rough sets to deal with incomplete data was presented by Kryszkiewicz under a tolerance relation [30]. The key point introduced in this method is to assign a value of "null" as a value of "everything is possible" in IIS [39]. An information system $S = (U, A, V, f)$ is called an IIS if V_a contains a null value "*" for at least one attribute $a \in A$ [30,31].

Definition 1. (See [30].) Given an IIS $S = (U, A, V, f)$. Let $B \subseteq A$ be a subset of attributes. The tolerance relation, denoted by TR_B, is defined as:

$$\forall x, y, TR_B(x, y) \iff \forall a \in B : \big(a(x) = a(y)\big) \vee \big(a(y) = *\big) \vee \big(a(x) = *\big).$$

Obviously, the tolerance relation TR is reflexive and symmetric, but not transitive.

Followed by Kryszkiewicz's definition, Stefanowski et al. extended the model of rough sets by using of a non-symmetric similarity relation [69,70]. In this approach, objects are described "incompletely" due to our imperfect knowledge and definitely impossible descriptions of all their attributes [39].

Definition 2. (See [69].) Given an IIS $S = (U, A, V, f)$. Let $B \subseteq A$ be a subset of attributes. The similarity relation, denoted by SR_B, is defined as:

$$\forall x, y, SR_B(x, y) \iff \forall a \in B : \big(a(x) \neq *, \big(a(x) = a(y)\big)\big).$$

Obviously, the similarity relation SR is reflexive and transitive, but not symmetric. One object x can be considered similar to another object y only if they have the same known values [39].

However, both tolerance relation TR and similarity relation SR have their limitations in applications [74,75]. For example, let $x_1 = (*, *, 1, 0, 0, *, 1)$ and $y_1 = (1, 0, *, *, *, 0, *)$. We have $x_1 TR y_1$, but x_1 and y_1 are obviously not similar. In addition, let $x_2 = (0, *, 1, 0, 0, 0, 1)$ and $y_2 = (0, 0, 1, 0, *, 0, 1)$. Clearly $x_2 SR y_2$ does not hold, but x_2 and y_2 are nearly the same. Observed by limitations of these kinds of extensions for IIS, Wang proposed a limited tolerance relation LR.

Definition 3. (See [74].) Given an IIS $S = (U, A, V, f)$. Let $B \subseteq A$ be a subset of attributes. The limited tolerance relation, denoted by LR_B, is defined as:

$$\begin{aligned}&\forall x, y, LR_B(x, y) \iff \\ &\quad \forall a \in B : \big(a(x) = a(y) = *\big) \vee \big(\big(P_B(x) \cap P_B(y) \neq \emptyset\big) \wedge \big(a(x) \neq *\big) \wedge \big(a(y) \neq *\big) \rightarrow \big(a(x) = a(y)\big)\big),\end{aligned}$$

where $P_B(x) = \{a \mid a \in B \wedge a(x) = *\}$. Obviously, the limited tolerance relation LR is reflexive and symmetric, but not transitive.

With above definitions, the null value "*" can be replaced by any value in V_a, which is called "do not care" conditions in [19]. Grzymala-Busse pointed out that there exists another explanation of missing value, namely, an attribute value is lost [19]. In this scenario, the missing value is not comparable with any value in V_a. Then he proposed a new extension of rough sets in terms of the characteristic relation under the assumption that some of the missing attribute values are lost (*i.e.*, were erased but currently are unavailable) and some are "do not care" conditions (*i.e.*, they are redundant or unnecessary to make a decision or to classify a case) [19,39]. For simplicity, all the missing values are denoted by "$*$" or "?", where "do not care" condition is denoted by "$*$", and the lost value is denoted by "?".

Definition 4. (See [19].) Given an IIS $S = (U, A, V, f)$. Let $B \subseteq A$ be a subset of attributes. The characteristic relation, denoted by CR_B, is defined as:

$$\forall x, y, CR_B(x, y) \iff \forall a \in B : \big(a(x) \neq ?\big) \wedge \big(\big(a(x) = a(y)\big) \vee \big(a(x) = *\big) \vee \big(a(y) = *\big)\big).$$

Obviously, the characteristic relation CR is reflexive, but not symmetric and transitive. The characteristic relation is the generalization of tolerance and similarity relation. Furthermore, Grzymala-Busse believed there exists the third type of missing attribute value called an "attribute-concept value" beside the two former explanations by considering the semantics interpretation to a specific concept [20], which brought a strategy for inducing better rule sets [22].

Fig. 1 (originally from [78,79]) clearly illustrates the relations between Pawlak rough sets and extended rough sets. The four relations in Definitions 1–4 are indicated, and the comparisons of the four relations are outlined by the different levels of the structure in Fig. 1.

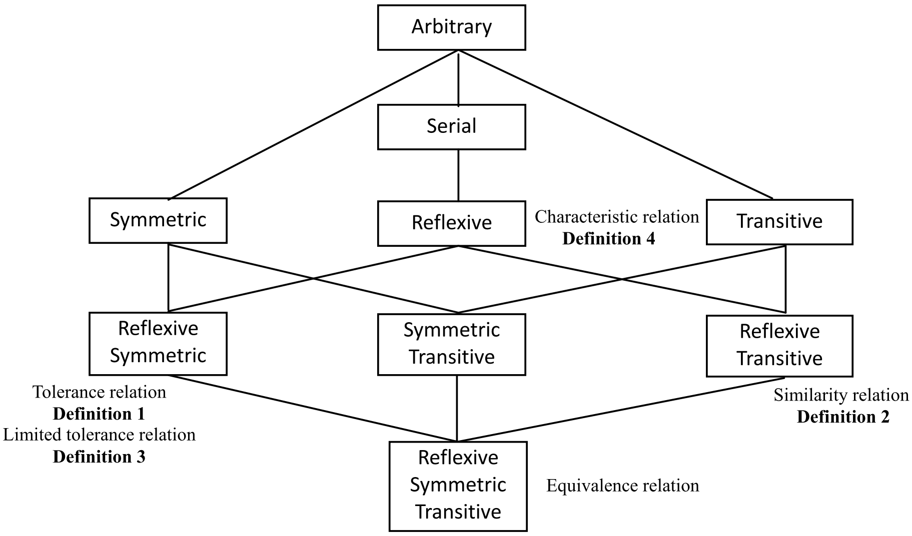

Fig. 1. The relation framework among Pawlak rough set model and extended rough set models.

2.3. The necessary concepts in knowledge discovery

Rough set theory proposed by Pawlak is an excellent mathematical tool to deal with inconsistency and ambiguity information, it focuses on the approximation of a concept by using blocks induced by a binary relation [58,59]. One may approximate a set by a pair of lower and upper approximations or equivalently by the positive, boundary, and negative regions. The lower approximation is the union of those equivalence classes that are included in the set and the upper approximation is the union of those that have a non-empty overlap with the set. Intuitively, the knowledge generated by the approximations may include useful information. The motivation of our incremental strategies for knowledge discovery is based on the approximation updating.

As another representation of approximations in rough sets, we present the basic decision rule synthesis to describe the objects in approximations as follows.

Definition 5. (See [2].) For a decision table $S = (U, C \cup D, V, f)$ with $C = \{a_1, a_2, \cdots, a_p\}$ and $D = \{d\}$, $\forall x \in U$, we generate a decision rule in the following way: as the predecessor of the decision rule we take the conjunction $(a_1 = a_1(x)) \wedge \cdots \wedge (a_p = a_p(x))$ and as the successor of the rule we take decision attribute d with value $d(x)$. Hence, the constructed decision rule for the object x is of the form:

$$\big(a_1 = a_1(x)\big) \wedge \cdots \wedge \big(a_p = a_p(x)\big) \rightarrow d = d(x).$$

In Definition 5, the formula can be equally written as: $X_i \rightarrow D_j$ if $x \in X_i$ and $x \in D_j$. Here, we do not consider the minimal patterns for updating decision rules in this paper. Finding the minimal patterns for rules dynamically is an another important work, namely, incremental attribute reduction in rough sets. As Ciucci mentioned in [10,13], we only consider one dynamics of quality indexes in our following discussions.

In order to extract useful information with knowledge discovery technology, many scholars gave their opinions to define the interestingness of information with different viewpoints. With the perspective of data mining, Han and Kamber suggested that the useful information is usually induced by certain patterns, which represent in the form of association rules. The rule support and rule confidence are utilized to measure the rule interestingness [26]. The rule support reflects the usefulness of discovered rules and the rule confidence reflects the certainty of discovered rules. Association rules are considered interesting if they satisfy both a minimum support threshold and a minimum confidence threshold [26]. In addition, Wong and Ziarko utilized two measures, namely, the confidence and resolution factors for inductive learning [76]. Michalski et al. proposed an approach based on conditional probabilities as in the AQ15 data mining system [55]. Grzymala-Busse proposed an approach based on specificity, strength and support as in the LERS data mining system [16]. Pawlak introduced two factors, certainty and coverage factors of decision rules, to express how exact is our knowledge (data) about the considered reality [60]. Tsumoto argued the accuracy and coverage measure of the degree of sufficiency and necessity, respectively, and the simplest probabilistic model is the one that only uses classification rules that have high accuracy and high coverage [72,73]. To sum up, the three parameters of support, accuracy (certainty in [60]) and coverage can be used as the most important factors to discover the knowledge.

Definition 6. (See [43,44,60].) Given a decision table $S = (U, C \cup D, V, f)$, $U/C = \{X_1, X_2, \cdots, X_m\}$ is a partition of objects under the condition attributes of C, where X_i $(i = 1, 2, \cdots, m)$ is an equivalence class of condition attributes; $U/D = \{D_1, D_2, \cdots, D_n\}$ is a partition of objects under the decision attributes of D, where D_j $(j = 1, 2, \cdots, n)$ is an equivalence class of decision attributes. $\forall X_i \in U/C$, $\forall D_j \in U/D$, the support, accuracy and coverage of a rule $X_i \to D_j$ are defined respectively as follows.

$$\text{Support of } X_i \to D_j\text{:} \quad Supp(D_j|X_i) = |X_i \cap D_j|;$$

$$\text{Accuracy of } X_i \to D_j\text{:} \quad Acc(D_j|X_i) = |X_i \cap D_j|/|X_i|;$$

$$\text{Coverage of } X_i \to D_j\text{:} \quad Cov(D_j|X_i) = |X_i \cap D_j|/|D_j|,$$

where $|X_i|$ and $|D_j|$ denote the cardinality of set X_i and D_j, respectively. For a decision rule $X_i \to D_j$, X_i is the predecessor of the decision rule, D_j is the successor of the decision rule according to Definition 5.

By considering the massive data in real-life applications, we utilize the strategy of matrices to simplify the problem. The support matrix, the accuracy matrix, the coverage matrix as well as their propositions are defined as follows [43,44].

$$Supp(D|X) = \begin{pmatrix} Supp(D_1|X_1) & Supp(D_2|X_1) & \cdots & Supp(D_n|X_1) \\ Supp(D_1|X_2) & Supp(D_2|X_2) & \cdots & Supp(D_n|X_2) \\ \vdots & \vdots & \vdots & \vdots \\ Supp(D_1|X_m) & Supp(D_2|X_m) & \cdots & Supp(D_n|X_m) \end{pmatrix} \tag{1}$$

$$Acc(D|X) = \begin{pmatrix} Acc(D_1|X_1) & Acc(D_2|X_1) & \cdots & Acc(D_n|X_1) \\ Acc(D_1|X_2) & Acc(D_2|X_2) & \cdots & Acc(D_n|X_2) \\ \vdots & \vdots & \vdots & \vdots \\ Acc(D_1|X_m) & Acc(D_2|X_m) & \cdots & Acc(D_n|X_m) \end{pmatrix} \tag{2}$$

$$Cov(D|X) = \begin{pmatrix} Cov(D_1|X_1) & Cov(D_2|X_1) & \cdots & Cov(D_n|X_1) \\ Cov(D_1|X_2) & Cov(D_2|X_2) & \cdots & Cov(D_n|X_2) \\ \vdots & \vdots & \vdots & \vdots \\ Cov(D_1|X_m) & Cov(D_2|X_m) & \cdots & Cov(D_n|X_m) \end{pmatrix} \tag{3}$$

Proposition 1. *(See [43,44].)* $Supp(D_j|X_i) \geq 0$, $\forall X_i \in U/C$, $\forall D_j \in U/D$, $i = 1, 2, \cdots, m$, $j = 1, 2, \cdots, n$.

Proposition 2. *(See [43,44].)* $0 \leq Acc(D_j|X_i) \leq 1$ *and* $\sum_{j=1}^{n} Acc(D_j|X_i) = 1$, $\forall X_i \in U/C$, $i = 1, 2, \cdots, m$.

Proposition 3. *(See [43,44].)* $0 \leq Cov(D_j|X_i) \leq 1$ *and* $\sum_{i=1}^{m} Cov(D_j|X_i) = 1$, $\forall D_j \in U/D$, $j = 1, 2, \cdots, n$.

According to Definition 5, the relations among the three matrices in (1)–(3) can be written as: $Acc(D_j|X_i) = \frac{Supp(D_j|X_i)}{\sum_{j=1}^{n} Supp(D_j|X_i)}$ and $Cov(D_j|X_i) = \frac{Supp(D_j|X_i)}{\sum_{i=1}^{m} Supp(D_j|X_i)}$, namely, the "support" can be expressed by both "accuracy" and "coverage". Followed by Pawlak and Tsumoto's ideas [60,72,73], we choose the two factors, accuracy and coverage, to describe the knowledge in this paper. The definition of knowledge is displayed as follows.

Definition 7. (See [43,44].) $\forall X_i$ $(i = 1, 2, \cdots, m)$, $\forall D_j$ $(j = 1, 2, \cdots, n)$, if $Acc(D_j|X_i) \geq \alpha$ and $Cov(D_j|X_i) \geq \beta$ hold, we call the rule $X_i \to D_j$ a kind of knowledge where $\alpha \in (0.5, 1)$ and $\beta \in (0, 1)$.

The values α and β are dependent on a problem itself. Generally, we choose the decision rules with high accuracy and high coverage [44].

3. Incremental approaches for knowledge discovery based on extended rough sets

In this section, we focus on discussing the knowledge updating process in IIS when the object set evolves over time while the attribute set remains unchanged. Followed by the incremental approaches in [43,44], the incremental strategies on adding and deleting of objects in IIS are firstly investigated. To illustrate our method clearly, we divide the work into three parts. We provide several basic assumptions of the incremental approach for inducing knowledge in Section 3.1. We introduce a matrix updating strategy to achieve the knowledge incremental learning process in Section 3.2. An incremental model as well as its analysis procedure is proposed. We provide a matrix based incremental algorithm for learning knowledge in Section 3.2.

3.1. Assumptions and basic structure for approaches of inducing knowledge incrementally

In our following discussions, we assume the process for incremental learning knowledge lasts two stages, namely, from time t to time $t+1$. By considering the objects may enter into or get out of a decision table at time $t+1$, Liu et al. suggested there exist two scenarios, namely, adding and deleting of objects [43,44]. In the first scenario, adding objects leads to the expansion of the universe. Similarly, in the second scenario, deleting objects brings on the contraction of the universe. In short, both adding and deleting processes reflect the coarsening or refining of the object set in an information table [44].

To describe the incremental knowledge discovery procedure in IIS, we denote an incomplete decision table at time t as $IDT = (U, C \cup D, V, f)$. $U/C = \{X_1, X_2, \cdots, X_m\}$ stands for the predecessor of the decision rules. The objects in X_i $(i = 1, 2, \cdots, m)$ are indiscernible; $U/D = \{D_1, D_2, \cdots, D_n\}$ generated by D stands for the successor of the decision rule. The objects in D_j $(j = 1, 2, \cdots, n)$ are indiscernible. However, with a binary relation R in an incomplete decision table (R can be set as TR, SR, LR or CR), X_i can form a new set X_i^R with $X_i^R = \{y \in U \mid (x, y) \in R, x \in X_i\}$, and $\{X_1^R, X_2^R, \cdots, X_m^R\}$ forms a cover of U with R. At time $t+1$, some objects enter the decision table while some go out. Therefore, the original incomplete decision table IDT will change into $IDT' = (U', C' \cup D', V', f')$, $U'/C' = \{X_1', X_2', \cdots, X_{m'}'\}$ and $U'/D' = \{D_1', D_2', \cdots, D_{n'}'\}$. Similarly, X_i' can form a new set $X_i'^R$ with $X_i'^R = \{y \in U \mid (x, y) \in R, x \in X_i'\}$, and $\{X_1'^R, X_2'^R, \cdots, X_{m'}'^R\}$ forms a cover of U' with R. For simplicity, the relation between IDT and IDT' can be written as $U' \neq U$, $C' = C$, $D' = D$ on the variation of objects.

With these stipulations, the following work focuses on the incremental strategies, models and algorithms for inducing knowledge under adding and deleting of objects in IIS.

3.2. The incremental model for knowledge discovery in IIS

In order to illustrate the incremental model, the variation of objects can be divided into three parts: (1) One object enters or gets out of the incomplete decision table at time $t+1$; (2) The objects enter or get out of the incomplete decision table at time $t+1$; (3) The objects enter and get out of the incomplete decision table simultaneously at time $t+1$. In the following, we firstly discuss the scenarios that adding an object and deleting of an object, respectively. Then, we discuss the scenarios that adding objects and deleting objects, respectively. At last, we discuss the scenario of adding and deleting of objects simultaneously.

3.2.1. Adding one object

Suppose an object $\bar{x}$ enters the incomplete decision table $IDT = (U, C \cup D, V, f)$ at time $t+1$, and $U' = U \cup \{\bar{x}\}$. Adding $\bar{x}$ may lead to following results.

(1) If $\exists X_i \subseteq U, x \in X_i, \forall a \in C, f(\bar{x}, a) = f(x, a)$, then $X_i' = X_i \cup \{\bar{x}\}$. In this scenario, we have:

$$X_i'^R = \begin{cases} X_i^R \cup \{\bar{x}\} & (x, \bar{x}) \in R, x \in X_i \\ X_i^R & (x, \bar{x}) \notin R, x \in X_i, \end{cases} \quad i = 1, 2, \cdots, m \tag{4}$$

(2) If $\forall X_i \subseteq U, x \in X_i, \exists a \in C, f(\bar{x}, a) \neq f(x, a)$, then $\bar{x}$ generates a new class $X_{m+1}' = \{\bar{x}\}$. We have:

$$\begin{aligned} X_i'^R &= \begin{cases} X_i^R \cup \{\bar{x}\} & (x, \bar{x}) \in R, x \in X_i \\ X_i^R & (x, \bar{x}) \notin R, x \in X_i, \end{cases} \quad i = 1, 2, \cdots, m \\ X_{m+1}'^R &= \begin{cases} \{\bar{x}\} \cup (\bigcup_{i=1}^m X_i) & \exists x \in X_i, (\bar{x}, x) \in R \\ \{\bar{x}\} & \forall x \in X_i, (\bar{x}, x) \notin R, \end{cases} \quad i = 1, 2, \cdots, m \end{aligned} \tag{5}$$

(3) If $\exists D_j \subseteq U$, $x \in D_j$, $f(\bar{x}, d) = f(x, d)$, then $D_j' = D_j \cup \{\bar{x}\}$. We have:

$$D_j' = \begin{cases} D_j \cup \{\bar{x}\} & \bar{x} \in D_j \\ D_j & \bar{x} \notin D_j, \end{cases} \quad j = 1, 2, \cdots, n \tag{6}$$

(4) If $\forall D_j \subseteq U$, $x \in D_j$, $f(\bar{x}, d) \neq f(x, d)$, then $\bar{x}$ generates a new class $D_{n+1}' = \{\bar{x}\}$. We have:

$$D_j' = \begin{cases} D_j & j = 1, 2, \cdots, n \\ \{\bar{x}\} & j = n+1 \end{cases} \tag{7}$$

From (4)–(7), adding one object $\bar{x}$ may affect the variation of multiple classes $X_i'^R$ $(i \in 1, 2, \cdots, m+1)$ and one class D_j $(j \in 1, 2, \cdots, n+1)$.

3.2.2. Deleting one object

Suppose an object $\tilde{x}$ gets out the incomplete decision table $IDT=(U,C\cup D,V,f)$ at time $t+1$, and $U'=U-\{\tilde{x}\}$. Deleting $\tilde{x}$ may lead to following results.

(1) Suppose $\tilde{x}\in X_i$, the deletion of $\tilde{x}$ leads to $X_i'=X_i-\{\tilde{x}\}$, then

$$X_i'^R=\begin{cases} X_i^R-\{\tilde{x}\} & (x,\tilde{x})\in R, x\in X_i \\ X_i^R & (x,\tilde{x})\notin R, x\in X_i \end{cases} \tag{8}$$

where $i=1,2,\cdots,m$. Specially, if $X_i=\{\tilde{x}\}$, then the deletion of $\tilde{x}$ will eliminate the class X_i, namely, $X_i'=\{\tilde{x}\}-\{\tilde{x}\}=\emptyset$, $X_i'^R=\emptyset$.

(2) Suppose $\tilde{x}\in D_j$, the deletion of $\tilde{x}$ leads to $D_j'=D_j-\{\tilde{x}\}$, then

$$D_j'=\begin{cases} D_j-\{\tilde{x}\} & \tilde{x}\in D_j \\ D_j & \tilde{x}\notin D_j \end{cases} \tag{9}$$

where $j=1,2,\cdots,n$. Specially, if $D_j=\{\tilde{x}\}$, then the deletion of $\tilde{x}$ will eliminate the class D_j, namely, $D_j'=\{\tilde{x}\}-\{\tilde{x}\}=\emptyset$.

From (8)–(9), deleting of one object $\tilde{x}$ may affect the variation of multiple classes $X_i'^R$ $(i\in 1,2,\cdots,m)$ and one class D_j $(j\in 1,2,\cdots,n)$.

3.2.3. Adding objects

Suppose M new objects U_{im} enter the system at time $t+1$. Adding objects U_{im} generates two partitions: $U_{im}/C'=\{X'_{l_1},\cdots,X'_{l_p};X'_{l'_1},\cdots,X'_{l'_q}\}$ and $U_{im}/D'=\{D'_{r_1},\cdots,D'_{r_{p'}};D'_{r'_1},\cdots,D'_{r'_{q'}}\}$, where $\{l_1,\cdots,l_p\}\subseteq\{1,2,\cdots,m\}$, $\{l'_1,\cdots,l'_q\}\cap\{1,2,\cdots,m\}=\emptyset$; $\{r_1,\cdots,r_{p'}\}\subseteq\{1,2,\cdots,n\}$, $\{r'_1,\cdots,r'_{q'}\}\cap\{1,2,\cdots,n\}=\emptyset$. Hence, let $U'=U\cup U_{im}$. We have $U'/C'=\{X'_1,X'_2,\cdots,X'_m;X'_{l'_1},\cdots,X'_{l'_q}\}$ and $U'/D'=\{D'_1,D'_2,\cdots,D'_n;D'_{r'_1},\cdots,D'_{r'_{q'}}\}$.

$$X_i'=\begin{cases} X_i\cup X_i' & i\in\{l_1,l_2,\cdots,l_p\} \\ X_i' & i\in\{l'_1,l'_2,\cdots,l'_q\} \\ X_i & \text{else} \end{cases}$$

$$D_j'=\begin{cases} D_j\cup D_j' & j\in\{r_1,r_2,\cdots,r_{p'}\} \\ D_j' & j\in\{r'_1,r'_2,\cdots,r'_{q'}\} \\ D_j & \text{else} \end{cases}$$

$$X_i'^R=\begin{cases} X_i^R\cup(\bigcup_{j=l_1,j=l'_1}^{l_p,l'_q}X_j') & i\in\{1,2,\cdots,m\}, j\in\{l_1,\cdots,l_p;l'_1,l'_2,\cdots,l'_q\}; x\in X_i, \exists y\in X_j', (x,y)\in R; \\ X_i^R & i\in\{1,2,\cdots,m\}, j\in\{l_1,\cdots,l_p;l'_1,l'_2,\cdots,l'_q\}; x\in X_i, \forall y\in X_j', (x,y)\notin R; \\ X_i'\cup(\bigcup_{j=1}^{m}X_j') & i\in\{l'_1,l'_2,\cdots,l'_q\}, j\in\{1,2,\cdots,m;l'_1,l'_2,\cdots,l'_q\}; x\in X_i', \exists y\in X_j', (x,y)\in R; \\ X_i' & i\in\{l'_1,l'_2,\cdots,l'_q\}, j\in\{1,2,\cdots,m\,l'_1,l'_2,\cdots,l'_q\}; x\in X_i',\ \forall y\in X_j', (x,y)\notin R. \end{cases} \tag{10}$$

We design an incremental algorithm for updating knowledge when objects add to IIS, which is shown in Algorithm 1.

3.2.4. Deleting objects

Suppose N objects U_{em} $(U_{em}\subseteq U, |U_{em}|<|U|)$ get out of system at time $t+1$. Deleting objects U_{em} generates two partition: $U_{em}/C'=\{X'_{l_1},\cdots,X'_{l_g}\}$ and $U_{em}/D'=\{D'_{r_1},\cdots,D'_{r_{g'}}\}$, where $\{l_1,\cdots,l_g\}\subseteq\{1,2,\cdots,m\}$; $\{r_1,\cdots,r_{g'}\}\subseteq\{1,2,\cdots,n\}$, $U'=U-U_{em}$, then

$$X_i'=\begin{cases} X_i-X_i' & i\in\{l_1,l_2,\cdots,l_g\} \\ X_i & \text{else} \end{cases}$$

$$D_j'=\begin{cases} D_j-D_j' & j\in\{r_1,r_2,\cdots,r_{g'}\} \\ D_j & \text{else} \end{cases}$$

$$X_i'^R=\begin{cases} X_i^R-(X_i^R\cap U_{em}) & X_i^R\cap U_{em}\neq\emptyset \\ X_i^R & X_i^R\cap U_{em}=\emptyset. \end{cases} \tag{11}$$

Similarly, we design an incremental algorithm for updating knowledge when objects delete from IIS, which is shown in Algorithm 2.

Algorithm 1: The incremental algorithm for updating knowledge when objects add to IIS.

Input:
(1) An incomplete decision table $S = (U, C \cup D, V, F)$;
(2) A binary relation R;
(3) Two thresholds α and β.
Output:
Knowledge at time t and $t+1$, respectively.

% Calculate the support matrix $Supp^{(t)}(D|X)$, accuracy matrix $Acc^{(t)}(D|X)$ and coverage matrix $Cov^{(t)}(D|X)$ at time t and output knowledge.
for $i = 1$ to m **do**
    for $j = 1$ to n **do**
        Calculate X_i^R and D_j, and generate $Supp^{(t)}(D_j|X_i)$, $Acc^{(t)}(D_j|X_i)$, $Cov^{(t)}(D_j|X_i)$ for the decision rule $X_i \to D_j$ at time t
    end
end
for $i = 1$ to m **do**
    for $j = 1$ to n **do**
        if $Acc^{(t)}(D_j|X_i) \geq \alpha$ and $Cov^{(t)}(D_j|X_i) \geq \beta$ **then**
            Output the decision rule $X_i \to D_j$
        end
    end
end
% Considering $\bar{x}$ ($\bar{x} \in U_{im}$) enters into the system.
for $\bar{x} = U_{im}^1$ to U_{im}^M **do**
    for $i = 1$ to m, $l = 0$ **do**
        for $j = 1$ to n, $r = 0$ **do**
            while $\bar{x} \notin X_i$ **do**
                if $\bar{x} \notin D_j$ **then**
                    Generate new classes X_{m+1} and D_{n+1}, $X_{m+1} = \bar{x}$, $D_{n+1} = \{\bar{x}\}$. Then l++, r++;
                else
                    Generate a new class X_{m+1}, $X_{m+1} = \{\bar{x}\}$, $D_j = D_j \cup \{\bar{x}\}$. Then l++.
                end
            end
            while $\bar{x} \in X_i$ **do**
                if $\bar{x} \notin D_j$ **then**
                    Generate a new class D_{n+1}, $D_{n+1} = \{\bar{x}\}$, $X_i = X_i \cup \{\bar{x}\}$. Then r++;
                else
                    $X_i = X_i \cup \{\bar{x}\}$, $D_j = D_j \cup \{\bar{x}\}$.
                end
            end
        end
    end
end
% Calculate the support matrix $Supp^{(t+1)}(D'|X')$, accuracy matrix $Acc^{(t+1)}(D'|X')$ and coverage matrix $Cov^{(t+1)}(D'|X')$ at time $t+1$ and output knowledge.
for $i = 1$ to $m+l$ **do**
    for $j = 1$ to $n+r$ **do**
        Calculate $X_i'^R$ and D_j' according to (10) and (11), and generate $Supp^{(t+1)}(D_j'|X_i')$, $Acc^{(t+1)}(D_j'|X_i')$, $Cov^{(t+1)}(D_j'|X_i')$ for the decision rule $X_i' \to D_j'$ at time $t+1$
    end
end
for $i = 1$ to $m+l$ **do**
    for $j = 1$ to $n+r$ **do**
        if $Acc^{(t+1)}(D_j'|X_i') \geq \alpha$ and $Cov^{(t+1)}(D_j'|X_i') \geq \beta$ **then**
            Output the decision rule $X_i' \to D_j'$
        end
    end
end

3.2.5. Adding and deleting objects

Followed by our discussion in Sections 3.2.1–3.2.4, suppose M new objects $U_{im} = \{U_{im}^1, U_{im}^2, \cdots, U_{im}^M\}$ enter the system and N objects $U_{em} = \{U_{em}^1, U_{em}^2, \cdots, U_{em}^N\}$ ($U_{em} \subseteq U$, $|U_{em}| < |U|$) get out of system at time $t+1$ simultaneously, and $U' = U \cup U_{im} - U_{em}$, $|U'| = |U| + M - N$. We assume the M new objects will form l new conditional classes $X'_{m+1}, X'_{m+2}, \cdots, X'_{m+l}$ and r new decision classes $D'_{n+1}, D'_{n+2}, \cdots, D'_{n+r}$. So at time $t+1$, $U'/C' = \{X'_1, X'_2, \cdots, X'_m, X'_{m+1}, \cdots, X'_{m+l}\}$ and $U'/D' = \{D'_1, D'_2, \cdots, D'_n, D'_{n+1}, \cdots, D'_{n+r}\}$. Note that X_i and X'_i actually describe the same class. The only difference of them is that their cardinalities are different, and this is also true for D_j and D'_j [44].

Algorithm 2: The incremental algorithm for updating knowledge when objects delete from IIS.

Input:
(1) An incomplete decision table $S=(U, C\cup D, V, F)$;
(2) A binary relation R;
(3) Two thresholds α and β.
Output:
Knowledge at time t and $t+1$, respectively.

% Calculate the support matrix $Supp^{(t)}(D|X)$, accuracy matrix $Acc^{(t)}(D|X)$ and coverage matrix $Cov^{(t)}(D|X)$ at time t and output knowledge.
for $i=1$ to m **do**
    for $j=1$ to n **do**
        Calculate X_i^R and D_j, and generate $Supp^{(t)}(D_j|X_i)$, $Acc^{(t)}(D_j|X_i)$, $Cov^{(t)}(D_j|X_i)$ for the decision rule $X_i \to D_j$ at time t
    end
end
for $i=1$ to m **do**
    for $j=1$ to n **do**
        if $Acc^{(t)}(D_j|X_i)\ge\alpha$ and $Cov^{(t)}(D_j|X_i)\ge\beta$ **then**
            Output the decision rule $X_i \to D_j$
        end
    end
end
% Considering $\tilde{x}$ ($\tilde{x}\in U_{em}$) gets out of the system at time $t+1$.
for $\tilde{x}=U_{em}^1$ to U_{em}^N **do**
    for $i=1$ to m **do**
        for $j=1$ to n **do**
            if $\tilde{x}\in X_i, \tilde{x}\in D_j$ **then**
                $X_i=X_i-\{\tilde{x}\}$, $D_j=D_j-\{\tilde{x}\}$;
            else
                $X_i=X_i$, $D_j=D_j$.
            end
        end
    end
end
% Calculate the support matrix $Supp^{(t+1)}(D'|X')$, accuracy matrix $Acc^{(t+1)}(D'|X')$ and coverage matrix $Cov^{(t+1)}(D'|X')$ at time $t+1$ and output knowledge.
for $i=1$ to $m+l$ **do**
    for $j=1$ to $n+r$ **do**
        Calculate ${X'_i}^R$ and D'_j according to (10) and (11), and generate $Supp^{(t+1)}(D'_j|X'_i)$, $Acc^{(t+1)}(D'_j|X'_i)$, $Cov^{(t+1)}(D'_j|X'_i)$ for the decision rule $X'_i \to D'_j$ at time $t+1$
    end
end
for $i=1$ to $m+l$ **do**
    for $j=1$ to $n+r$ **do**
        if $Acc^{(t+1)}(D'_j|X'_i)\ge\alpha$ and $Cov^{(t+1)}(D'_j|X'_i)\ge\beta$ **then**
            Output the decision rule $X'_i \to D'_j$
        end
    end
end

For clarity, we introduce the support matrix, the accuracy matrix and the coverage matrix to describe the knowledge updating procedure. At time t, the support matrix describes the cardinal number distribution for $X_i \to D_j$ with a binary relation R in IIS. To simplify the model, we expand the original $m\times n$ matrix into $(m+l)\times(n+r)$ matrix, and the row from $m+1$ to $m+l$, the column from $n+1$ to $n+r$ all set to zero. The support matrix at time t can be written as:

$$Supp^{(t)}(D|X)=\begin{pmatrix} |X_1^R\cap D_1| & |X_1^R\cap D_2| & \cdots & |X_1^R\cap D_n| & 0 & \cdots & 0 \\ |X_2^R\cap D_1| & |X_2^R\cap D_2| & \cdots & |X_2^R\cap D_n| & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ |X_m^R\cap D_1| & |X_2^R\cap D_2| & \cdots & |X_m^R\cap D_n| & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{12}$$

In addition, the accuracy matrix and the coverage matrix at time t can be written as:

$$Acc^{(t)}(D|X) = \begin{pmatrix} \frac{|X_1^R \cap D_1|}{\sum_{j=1}^n |X_1^R \cap D_j|} & \frac{|X_1^R \cap D_2|}{\sum_{j=1}^n |X_1^R \cap D_j|} & \cdots & \frac{|X_1^R \cap D_n|}{\sum_{j=1}^n |X_1^R \cap D_j|} & 0 & \cdots & 0 \\ \frac{|X_2^R \cap D_1|}{\sum_{j=1}^n |X_2^R \cap D_j|} & \frac{|X_2^R \cap D_2|}{\sum_{j=1}^n |X_2^R \cap D_j|} & \cdots & \frac{|X_2^R \cap D_n|}{\sum_{j=1}^n |X_2^R \cap D_j|} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|X_m^R \cap D_1|}{\sum_{j=1}^n |X_m^R \cap D_j|} & \frac{|X_m^R \cap D_2|}{\sum_{j=1}^n |X_m^R \cap D_j|} & \cdots & \frac{|X_m^R \cap D_n|}{\sum_{j=1}^n |X_m^R \cap D_j|} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{13}$$

$$Cov^{(t)}(D|X) = \begin{pmatrix} \frac{|X_1^R \cap D_1|}{\sum_{i=1}^m |X_i^R \cap D_1|} & \frac{|X_1^R \cap D_2|}{\sum_{i=1}^m |X_i^R \cap D_2|} & \cdots & \frac{|X_1^R \cap D_n|}{\sum_{i=1}^m |X_i^R \cap D_n|} & 0 & \cdots & 0 \\ \frac{|X_2^R \cap D_1|}{\sum_{i=1}^m |X_i^R \cap D_1|} & \frac{|X_2^R \cap D_2|}{\sum_{i=1}^m |X_i^R \cap D_2|} & \cdots & \frac{|X_2^R \cap D_n|}{\sum_{i=1}^m |X_i^R \cap D_n|} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|X_m^R \cap D_1|}{\sum_{i=1}^m |X_i^R \cap D_1|} & \frac{|X_m^R \cap D_2|}{\sum_{i=1}^m |X_i^R \cap D_2|} & \cdots & \frac{|X_m^R \cap D_n|}{\sum_{i=1}^m |X_i^R \cap D_n|} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{14}$$

According to (12)–(14), the matrices updating process depends on two aspects: the updating of X_i^R and D_j. Furthermore, with the insightful gain from (10) and (11), $X_i'^R$ $(i = 1, 2, \cdots, m+l)$ and D_j' $(j = 1, 2, \cdots, n+r)$ can be easily calculated by considering both adding and deleting of objects. For simplicity, we denote $\Delta X_i'^R$ $(i = 1, 2, \cdots, m+l)$ and $\nabla X_i'^R$ $(i = 1, 2, \cdots, m)$ as the objects which enter and get out of X_i^R with the binary relation R in IIS at time $t+1$, respectively. Similarly, we denote $\Delta D_j'$ $(j = 1, 2, \cdots, n+r)$ and $\nabla D_j'$ $(j = 1, 2, \cdots, n)$ as the objects which enter and get out of D_j at time $t+1$, respectively.

$$Supp^{(t+1)}(D'|X') = \begin{pmatrix} |X_1'^R \cap D_1'| & |X_1'^R \cap D_2'| & \cdots & |X_1'^R \cap D_n'| & \cdots & |X_1'^R \cap D_{n+r}'| \\ |X_2'^R \cap D_1'| & |X_2'^R \cap D_2'| & \cdots & |X_2'^R \cap D_n'| & \cdots & |X_2'^R \cap D_{n+r}'| \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ |X_m'^R \cap D_1'| & |X_2'^R \cap D_2'| & \cdots & |X_m'^R \cap D_n'| & \cdots & |X_m'^R \cap D_{n+r}'| \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ |X_{m+l}'^R \cap D_1'| & |X_{m+l}'^R \cap D_2'| & \cdots & |X_{m+l}'^R \cap D_n'| & \cdots & |X_{m+l}'^R \cap D_{n+r}'| \end{pmatrix} \tag{15}$$

where $X_i'^R \cap D_j'$ in (15) is calculated as:

$$X_i'^R \cap D_j' = \begin{cases} (X_i^R \cup \Delta X_i'^R - \nabla X_i'^R) \cap (D_j \cup \Delta D_j' - \nabla D_j'): & i \in \{1, 2, \cdots, m\};\ j \in \{1, 2, \cdots, n\} \\ \Delta X_i'^R \cap (D_j \cup \Delta D_j' - \nabla D_j'): & i \in \{m+1, m+2, \cdots, m+l\};\ j \in \{1, 2, \cdots, n\} \\ (X_i^R \cup \Delta X_i'^R - \nabla X_i'^R) \cap \Delta D_j': & i \in \{1, 2, \cdots, m\};\ j \in \{n+1, n+2, \cdots, n+r\} \\ \Delta X_i'^R \cap \Delta D_j': & i \in \{m+1, m+2, \cdots, m+l\};\ j \in \{n+1, n+2, \cdots, n+r\} \end{cases} \tag{16}$$

In the same way, the accuracy matrix and the coverage matrix at time $t+1$ is written as:

$$Acc^{(t+1)}(D'|X') = \begin{pmatrix} \frac{|X_1'^R \cap D_1'|}{\sum_{j=1}^{n+r} |X_1'^R \cap D_j'|} & \frac{|X_1'^R \cap D_2'|}{\sum_{j=1}^{n+r} |X_1'^R \cap D_j'|} & \cdots & \frac{|X_1'^R \cap D_n'|}{\sum_{j=1}^{n+r} |X_1'^R \cap D_j'|} & \cdots & \frac{|X_1'^R \cap D_{n+r}'|}{\sum_{j=1}^{n+r} |X_1'^R \cap D_j'|} \\ \frac{|X_2'^R \cap D_1'|}{\sum_{j=1}^{n+r} |X_2'^R \cap D_j'|} & \frac{|X_2'^R \cap D_2'|}{\sum_{j=1}^{n+r} |X_2'^R \cap D_j'|} & \cdots & \frac{|X_2'^R \cap D_n'|}{\sum_{j=1}^{n+r} |X_2'^R \cap D_j'|} & \cdots & \frac{|X_2'^R \cap D_{n+r}'|}{\sum_{j=1}^{n+r} |X_2'^R \cap D_j'|} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|X_m'^R \cap D_1'|}{\sum_{j=1}^{n+r} |X_m'^R \cap D_j'|} & \frac{|X_m'^R \cap D_2'|}{\sum_{j=1}^{n+r} |X_m'^R \cap D_j'|} & \cdots & \frac{|X_m'^R \cap D_n'|}{\sum_{j=1}^{n+r} |X_m'^R \cap D_j'|} & \cdots & \frac{|X_m'^R \cap D_{n+r}'|}{\sum_{j=1}^{n+r} |X_m'^R \cap D_j'|} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|X_{m+l}'^R \cap D_1'|}{\sum_{j=1}^{n+r} |X_{m+l}'^R \cap D_j'|} & \frac{|X_{m+l}'^R \cap D_2'|}{\sum_{j=1}^{n+r} |X_{m+l}'^R \cap D_j'|} & \cdots & \frac{|X_{m+l}'^R \cap D_n'|}{\sum_{j=1}^{n+r} |X_{m+l}'^R \cap D_j'|} & \cdots & \frac{|X_{m+l}'^R \cap D_{n+r}'|}{\sum_{j=1}^{n+r} |X_{m+l}'^R \cap D_j'|} \end{pmatrix} \tag{17}$$

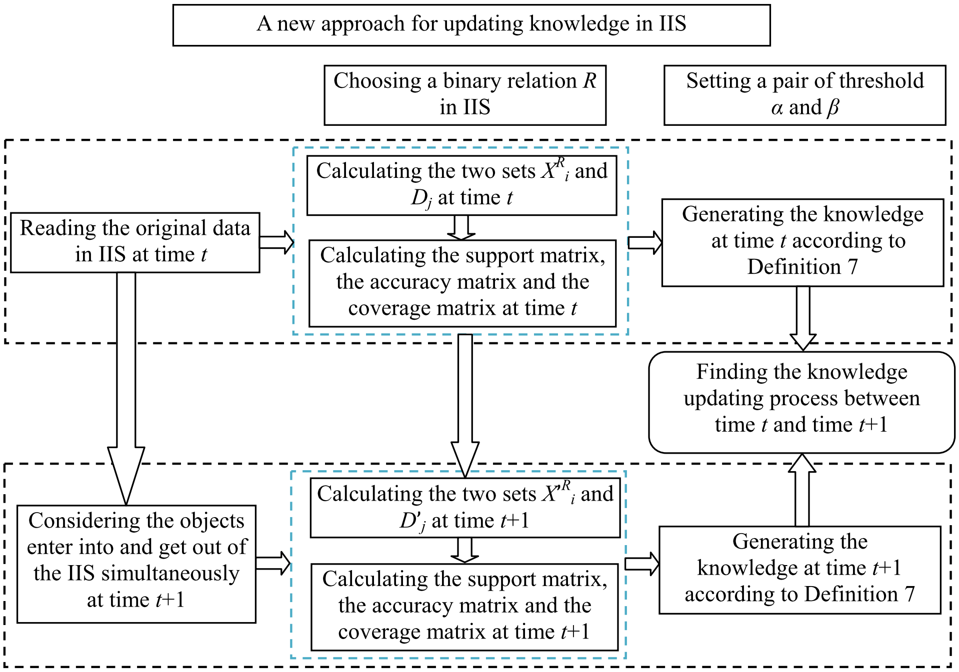

Fig. 2. A new approach for updating knowledge in IIS.

$$Cov^{(t+1)}(D'|X') = \begin{pmatrix} \frac{|X'^R_1 \cap D'_1|}{\sum_{i=1}^{m+l}|X'^R_1 \cap D'_j|} & \frac{|X'^R_1 \cap D'_2|}{\sum_{i=1}^{m+l}|X'^R_1 \cap D'_j|} & \cdots & \frac{|X'^R_1 \cap D'_n|}{\sum_{i=1}^{m+l}|X'^R_1 \cap D'_j|} & \cdots & \frac{|X'^R_1 \cap D'_{n+r}|}{\sum_{i=1}^{m+l}|X'^R_1 \cap D'_j|} \\ \frac{|X'^R_2 \cap D'_1|}{\sum_{i=1}^{m+l}|X'^R_2 \cap D'_j|} & \frac{|X'^R_2 \cap D'_2|}{\sum_{i=1}^{m+l}|X'^R_2 \cap D'_j|} & \cdots & \frac{|X'^R_2 \cap D'_n|}{\sum_{i=1}^{m+l}|X'^R_2 \cap D'_j|} & \cdots & \frac{|X'^R_2 \cap D'_{n+r}|}{\sum_{i=1}^{m+l}|X'^R_2 \cap D'_j|} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|X'^R_m \cap D'_1|}{\sum_{i=1}^{m+l}|X'^R_m \cap D'_j|} & \frac{|X'^R_m \cap D'_2|}{\sum_{i=1}^{m+l}|X'^R_m \cap D'_j|} & \cdots & \frac{|X'^R_m \cap D'_n|}{\sum_{i=1}^{m+l}|X'^R_m \cap D'_j|} & \cdots & \frac{|X'^R_m \cap D'_{n+r}|}{\sum_{i=1}^{m+l}|X'^R_m \cap D'_j|} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{|X'^R_{m+l} \cap D'_1|}{\sum_{i=1}^{m+l}|X'^R_{m+l} \cap D'_j|} & \frac{|X'^R_{m+l} \cap D'_2|}{\sum_{i=1}^{m+l}|X'^R_{m+l} \cap D'_j|} & \cdots & \frac{|X'^R_{m+l} \cap D'_n|}{\sum_{i=1}^{m+l}|X'^R_{m+l} \cap D'_j|} & \cdots & \frac{|X'^R_{m+l} \cap D'_{n+r}|}{\sum_{i=1}^{m+l}|X'^R_{m+l} \cap D'_j|} \end{pmatrix} \tag{18}$$

The relations of support, accuracy and coverage between time t and time $t+1$ are clearly displayed in matrices (12)–(18), which give an intuitive interpretation for knowledge discovery with a decision rule $X_i \to D_j$ when setting a pair of thresholds α and β. To sum up, the concrete steps of the incremental approach for updating knowledge are briefly listed as follows. The simple flowchart of our proposed approach for updating interesting knowledge in IIS is shown in Fig. 2.

Step 1: Calculating the support matrix, the accuracy matrix and the coverage matrix at time t according to Definition 6 as $Supp^{(t)}(D|X) = (Supp^{(t)}(D_j|X_i))_{m\times n}$, $Acc^{(t)}(D|X) = (Acc^{(t)}(D_j|X_i))_{m\times n}$ and $Cov^{(t)}(D|X) = (Cov^{(t)}(D_j|X_i))_{m\times n}$, respectively.

Step 2: Calculating the updating support matrix $Supp^{(t+1)}(D|X) = (Supp^{(t+1)}(D_j|X_i))_{(m+l)\times(n+r)}$ by (10), (11) and (15) at time $t+1$. Then, calculating the updating accuracy matrix $Acc^{(t+1)}(D'|X') = (Acc^{(t+1)}(D'_j|X'_i))_{(m+l)\times(n+r)}$ and the coverage matrix $Cov^{(t+1)}(D'|X') = (Cov^{(t+1)}(D'_j|X'_i))_{(m+l)\times(n+r)}$ at time $t+1$.

Step 3: Generating the decision rules $X_i \to D_j$ $(i = 1, 2, \cdots, m;\ j = 1, 2, \cdots, n)$ at time t and $X'_i \to D'_j$ $(i = 1, 2, \cdots, m+l;\ j = 1, 2, \cdots, n+r)$ at time $t+1$. Therefore, we may trace the change of knowledge at different stages according to the variation of matrices by setting a reasonable pair of thresholds according to Definition 7.

Furthermore, compared with the approach for objects updating strategy in complete information systems [43,44], the differences and similarities on dynamic maintenance between incomplete decision table and complete decision table are summarized in Table 1.

Table 1 lists five main indexes to illustrate the differences and similarities between incomplete decision table and complete decision table. The essential difference of the incremental strategy in the two types of decision tables is based on the relation. In the complete decision table, the predecessor X_i is generated by a partition U/C with an equivalence relation [42–44]; Whereas, in the incomplete decision table, the predecessor X_i^R is generated by a cover set with a binary relation R which mentioned in Section 3.1. However, the successor D_j is generated by a partition U/D in both complete decision table and incomplete decision table. Furthermore, adding or deleting an object at time $t+1$ may change one row and one

Table 1
The differences and similarities on dynamic maintenance between complete decision table and incomplete decision table.

Indexes	Incremental approach in complete decision table	Incremental approach in incomplete decision table
Relation based	Equivalence relation	A binary relation *TR*, *SR*, *LR* or *CR*
The predecessor X_i	Form a partition	Form a cover
The successor D_j	Form a partition	Form a partition
Adding an object $\bar{x}$	Changing one row and one column of three matrices [44]	Changing multiple rows and multiple columns of three matrices
Deleting an object $\tilde{x}$	Changing one row and one column of three matrices [44]	Changing multiple rows and multiple columns of three matrices

Table 2
An incomplete decision table about the medical diagnosis for identifying the cold.

U	a_1	a_2	a_3	a_4	d	*Num*
x_1	?	0	1	0	0	10
x_2	0	1	1	0	0	15
x_3	0	1	1	0	1	5
x_4	*	1	1	1	1	20
x_5	0	1	?	1	1	15
x_6	1	1	1	*	1	25
x_7	1	1	1	*	2	10
x_8	1	*	2	2	2	5

column of the original support matrix, accuracy matrix and coverage matrix at time t in a complete decision table [44], but it may change multiple rows and multiple columns of the three original matrices in an incomplete decision table.

4. An illustration and experimental evaluations

In this section, we demonstrate our incremental approach by using an example of medical diagnosis at first. Then, we compare our incremental algorithm with the non-incremental strategy by using the UCI databases. The experimental results are presented at the end of this section.

4.1. An illustration of the proposed approaches in medical diagnosis

Table 2 is an incomplete decision table $S = (U, A, V, f)$ about the medical diagnosis of patients collected for identifying the cold in a hospital. Suppose $U = \{x_1, x_2, \cdots, x_8\}$ stands for the 8 types of patients. We choose the symptoms {fever, cough, headache and nose snivel} as condition attributes described by the set $C = \{a_1, a_2, a_3, a_4\}$. The clinical decision making is considered as the decision attribute $D = \{d\}$. "?" and "*" mean missing data ("?" denotes lost value and "*" denotes "do not care" condition). For simplicity and clarity, we combine the same descriptions for the objects in the original data set as "*Num*", where *Num* stands for the cardinality of patients. Here, we only care about the characteristic relation *CR* in Table 2 since the analysis results with other binary relations *TR*, *SR* and *LR* in IIS can be easily obtained when we simply replace "?" by "*" with the criteria according to Definitions 1, 2 and 3.

The meaning of the values for every attribute is shown as follows.

Fever (a_1): 0 = No; 1 = Yes.

Cough (a_2): 0 = No; 1 = Yes.

Headache (a_3): 0 = No; 1 = A Little; 2 = Serious.

Nose snivel (a_4): 0 = No; 1 = A Little; 2 = Serious.

Clinical decision making (d): 0 = Have no cold; 1 = Need further investigate; 2 = Have cold.

In Table 2, $U/C = \{X_1, X_2, \cdots, X_6\}$, $U/D = \{D_1, D_2, D_3\}$, where $X_1 = \{x_1\}$, $X_2 = \{x_2, x_3\}$, $X_3 = \{x_4\}$, $\{X_4\} = \{x_5\}$, $X_5 = \{x_6, x_7\}$, $X_6 = \{x_8\}$; $D_1 = \{x_1, x_2\}$, $D_2 = \{x_3, x_4, x_5, x_6\}$, $D_3 = \{x_7, x_8\}$. According to Definition 4, we have:

$CR_C(x_1) = \{x_1, x_4, x_8\} \cap \{x_1, x_2, x_3, x_4, x_6, x_7\} \cap \{x_1, x_2, x_3, x_6, x_7\} = \{x_1\}$;

$CR_C(x_2) = CR_C(x_3) = \{x_2, x_3, x_4, x_5\} \cap \{x_2, x_3, x_4, x_5, x_6, x_7, x_8\} \cap \{x_1, x_2, x_3, x_4, x_6, x_7\} \cap \{x_1, x_2, x_3, x_6, x_7\} = \{x_2, x_3\}$;

$CR_C(x_4) = \{x_2, x_3, x_4, x_5, x_6, x_7, x_8\} \cap \{x_2, x_3, x_4, x_5, x_6, x_7, x_8\} \cap \{x_1, x_2, x_3, x_4, x_6, x_7\} \cap \{x_4, x_5, x_6, x_7\} = \{x_4, x_6, x_7\}$;

$CR_C(x_5) = \{x_2, x_3, x_4, x_5\} \cap \{x_2, x_3, x_4, x_5, x_6, x_7, x_8\} \cap \{x_4, x_5, x_6, x_7\} = \{x_4, x_5\}$;

$CR_C(x_6) = CR_C(x_7) = \{x_4, x_6, x_7, x_8\} \cap \{x_2, x_3, x_4, x_5, x_6, x_7, x_8\} \cap \{x_1, x_2, x_3, x_4, x_6, x_7\} \cap \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\} = \{x_4, x_6, x_7\}$

$CR_C(x_8) = \{x_4, x_6, x_7, x_8\} \cap \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\} \cap \{x_8\} \cap \{x_6, x_7, x_8\} = \{x_8\}$.

Hence, $X_1^{CR} = \{x_1\}$, $X_2^{CR} = \{x_2, x_3\}$, $X_3^{CR} = \{x_4, x_6, x_7\}$, $X_4^{CR} = \{x_4, x_5\}$, $X_5^{CR} = \{x_4, x_6, x_7\}$, $X_6^{CR} = \{x_8\}$.

Then, we construct the 6×3 support matrix for $X_i \to D_j$ at time t as follows.

$$Supp^{(t)}(D|X) = \begin{pmatrix} 10 & 0 & 0 \\ 15 & 5 & 0 \\ 0 & 45 & 10 \\ 0 & 35 & 0 \\ 0 & 45 & 10 \\ 0 & 0 & 5 \end{pmatrix} \tag{19}$$

The accuracy matrix and the coverage matrix at time t are calculated respectively according to Definition 6 as follows.

$$Acc^{(t)}(D|X) = \begin{pmatrix} 1 & 0 & 0 \\ 0.75 & 0.25 & 0 \\ 0 & 0.818 & 0.182 \\ 0 & 1 & 0 \\ 0 & 0.818 & 0.182 \\ 0 & 0 & 1 \end{pmatrix} \tag{20}$$

$$Cov^{(t)}(D|X) = \begin{pmatrix} 0.4 & 0 & 0 \\ 0.6 & 0.039 & 0 \\ 0 & 0.346 & 0.4 \\ 0 & 0.269 & 0 \\ 0 & 0.346 & 0.4 \\ 0 & 0 & 0.2 \end{pmatrix} \tag{21}$$

Suppose some new data of patient's symptoms will enter and some original data will get out of the incomplete decision table at time $t+1$. We assume there are three cases happen: (I) 10 patients with the diagnosis of "$a_1 = 1$, $a_2 = 1$, $a_3 = 2$, $a_4 = 2$, $d = 2$" named x_9 will enter the decision table; (II) 20 patients with the diagnosis of "$a_1 = 0$, $a_2 = 1$, $a_3 = 1$, $a_4 = 0$, $d = 0$" named x_{10} will enter the decision table; (III) 5 patients with the diagnosis of "$a_1 = 0$, $a_2 = 1$, $a_3 = ?$, $a_4 = 1$, $d = 1$" in x_5 will get out of the decision table.

For the case (I), the symptom of "$a_1 = 1$, $a_2 = 1$, $a_3 = 2$, $a_4 = 2$, $d = 2$" generates a new class $X_7 = \{x_9\}$. Due to $CR_C(x_9) = \{x_4, x_6, x_7, x_8, x_9\} \cap \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\} \cap \{x_8, x_9\} \cap \{x_6, x_7, x_8, x_9\} = \{x_8, x_9\}$ and $\{x_8\} \subseteq X_6$, $X_7'^{CR} = \{x_9\} \cup X_6 = \{x_8, x_9\}$. On the other hand, $\forall x \in X_i$ $(i = 1, 2, 3, 4, 5)$, $(x, x_9) \notin CR$. So $X_i'^{CR} = X_i^{CR}$. Due to $\{x_8\} \subseteq X_6$, $(x_8, x_9) \in CR$, then $X_6'^{CR} = X_6^{CR} \cup \{x_9\}$. According to Section 3.2.1, we have:

$$X_i'^{CR} = \begin{cases} X_i^{CR} & i = 1, 2, 3, 4, 5 \\ X_i^{CR} \cup \{x_9\} & i = 6 \\ \{x_9\} \cup X_6 & i = 7 \end{cases}$$

$$D_j' = \begin{cases} D_j & j = 1, 2 \\ D_j \cup \{x_9\} & j = 3 \end{cases}$$

For the case (II), the symptom of "$a_1 = 0$, $a_2 = 1$, $a_3 = 1$, $a_4 = 0$, $d = 0$" has existed in Table 2, and $\{x_{10}\} \in X_2$, $X_2^{CR} \cap X_i^{CR} = \emptyset$ $(i \neq 2)$. According to Section 3.2.1, we have:

$$X_i'^{CR} = \begin{cases} X_i^{CR} \cup \{x_{10}\} & i = 2 \\ X_i^{CR} & i = 1, 3, 4, 5, 6 \end{cases}$$

$$D_j' = \begin{cases} D_j \cup \{x_{10}\} & j = 1 \\ D_j & j = 2, 3 \end{cases}$$

For the case (III), there are 5 medical records deleted from x_5. Due to $x_5 \in X_4^{CR}$ and $x_5 \in D_2$, we have $|X_4'^{CR}| = |X_4^{CR}| - 5$ and $|D_2'| = |D_2| - 5$ according to Section 3.2.2.

As stated above, the support matrix for $X_i' \to D_j'$ at time $t+1$ is calculated according to (16) when we combine the three cases.

$$Supp^{(t+1)}(D'|X') = \begin{pmatrix} 10 & 0 & 0 \\ 15+20 & 5 & 0 \\ 0 & 45 & 10 \\ 0 & 35-5 & 0 \\ 0 & 45 & 10 \\ 0 & 0 & 5+10 \\ 0 & 0 & 10+5 \end{pmatrix} = \begin{pmatrix} 10 & 0 & 0 \\ 35 & 5 & 0 \\ 0 & 45 & 10 \\ 0 & 30 & 0 \\ 0 & 45 & 10 \\ 0 & 0 & 15 \\ 0 & 0 & 15 \end{pmatrix} \tag{22}$$

Table 3
The knowledge with 2-dimensional value pairs at time t and $t+1$.

Time	Class	D_1 (D'_1)	D_2 (D'_2)	D_3 (D'_3)
t	X_1	$\underline{(1, 0.4)}$	$(0, 0)$	$(0, 0)$
$t+1$	X'_1	$(1, 0.222)$	$(0, 0)$	$(0, 0)$
t	X_2	$\underline{(0.75, 0.6)}$	$(0.25, 0.039)$	$(0, 0)$
$t+1$	X'_2	$\underline{(0.875, 0.778)}$	$(0.125, 0.04)$	$(0, 0)$
t	X_3	$(0, 0)$	$\underline{(0.818, 0.346)}$	$(0.182, 0.4)$
$t+1$	X'_3	$(0, 0)$	$\underline{(0.818, 0.36)}$	$(0.182, 0.2)$
t	X_4	$(0, 0)$	$(1, 0.269)$	$(0, 0)$
$t+1$	X'_4	$(0, 0)$	$(1, 0.24)$	$(0, 0)$
t	X_5	$(0, 0)$	$\underline{(0.818, 0.346)}$	$(0.182, 0.4)$
$t+1$	X'_5	$(0, 0)$	$\underline{(0.818, 0.36)}$	$(0.182, 0.2)$
t	X_6	$(0, 0)$	$(0, 0)$	$(1, 0.2)$
$t+1$	X'_6	$(0, 0)$	$(0, 0)$	$\underline{(1, 0.3)}$
t	X_7	–	–	–
$t+1$	X'_7	$(0, 0)$	$(0, 0)$	$\underline{(1, 0.3)}$

Note: The first place of the value pair (•,•) means the value of accuracy and the second place means the value of coverage. The sign "–" stands for the non-existence pair.

In the same way, the accuracy matrix and the coverage matrix at time $t+1$ is calculated by (22) according to Definition 6 as follows.

$$Acc^{(t+1)}(D'|X') = \begin{pmatrix} 1 & 0 & 0 \\ 0.875 & 0.125 & 0 \\ 0 & 0.818 & 0.182 \\ 0 & 1 & 0 \\ 0 & 0.818 & 0.182 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \tag{23}$$

$$Cov^{(t+1)}(D'|X') = \begin{pmatrix} 0.222 & 0 & 0 \\ 0.778 & 0.040 & 0 \\ 0 & 0.360 & 0.200 \\ 0 & 0.240 & 0 \\ 0 & 0.360 & 0.200 \\ 0 & 0 & 0.300 \\ 0 & 0 & 0.300 \end{pmatrix} \tag{24}$$

Followed by the concrete steps in Section 3.2.5, (20) and (21) are utilized to construct the decision rules $X_i \to D_j$ at time t; (23) and (24) are utilized to construct the decision rules $X'_i \to D'_j$ at time $t+1$. To illuminate our problem, we construct the 2-dimensional value pairs $(Acc^{(t)}(D_j|X_i), Cov^{(t)}(D_j|X_i))$ at time t and $(Acc^{(t+1)}(D'_j|X'_i), Cov^{(t+1)}(D'_j|X'_i))$ at time $t+1$, respectively. The knowledge with 2-dimensional value pairs at time t and $t+1$ is shown in Table 3.

Table 3 provides an intuitive picture for understanding the knowledge discovery procedure in the two stages. We can easily obtain the knowledge at time t and $t+1$ when we fix a reasonable value pair (α, β). For example, if we set $\alpha = 0.6$ and $\beta = 0.3$, the underlined value pairs in Table 3 denote the rules which satisfy $\alpha \geq 0.6$ and $\beta \geq 0.3$. The rules of $X_1 \to D_1$, $X_2 \to D_1$, $X_3 \to D_2$ and $X_5 \to D_2$ are important at time t by using the judging criteria in Definition 6. In the same way, the rules of $X'_2 \to D'_1$, $X'_3 \to D'_2$, $X'_5 \to D'_2$, $X'_6 \to D'_3$ and $X'_7 \to D'_3$ are important at time $t+1$. We find the rule $X'_1 \to D'_1$ is no longer important and the rules $X'_6 \to D'_3$ and $X'_7 \to D'_3$ become important because of the updating of database. Generally, decision makers can choose suitable α and β to generate the knowledge according to their own experience [43,44].

4.2. The experimental evaluations

4.2.1. The experimental performances with nine datasets

Experiments are performed on a computer with Intel(R) Core(TM)2 DUO CPU E7500 and 4 GB of memory, running Microsoft Windows 7. The algorithms for updating knowledge in our discussion and the non-incremental updating strategy are developed and programmed in VC++ 6.0. We choose nine datasets, from the UC Irvine Machine Learning Database Repository (http://archive.ics.uci.edu/ml/datasets.html), named "IRIS", "Cup", "Zoo", "Segment", "Bank", "Tic-tac-toe", "Soybean", "Car" and "Mushroom", as benchmarks for the performance tests, and all data in the nine datasets are marked with numerical or category types. Table 4 shows the basic statistical information of the nine datasets.

Note that, the attribute characteristics of the nine datasets in Table 4 consist of categorical, integer or real number. For the datasets with categorical or integer number, we can directly use them to our experiments. For the datasets with real

Table 4
The basic information of the nine datasets.

No.	Data set name	Samples	Features	The number of concepts	Miss values
1	Iris	150	4	3	No
2	Cup	209	6	3	No
3	Zoo	101	17	7	No
4	Segment	1500	18	7	No
5	Bank	600	10	2	No
6	Tic-tac-toe	958	9	2	No
7	Soybean	682	35	19	No
8	Car	1728	6	4	No
9	Mushroom	8124	22	2	Yes

values (*e.g.*, the Iris dataset), we discrete them by using the equal frequency binning algorithm in Rosetta[1] at first, then utilize the discretization version of datasets in our experiments.

According to the last column in Table 4, the first eight data sets (excluding the data set "Mushroom") are complete with no missing values. In order to execute our experiments, we randomly choose 1% data of each of the first eight data sets in Table 4 and utilize "*" to replace the true values, respectively. Meanwhile, we utilize "*" to define the missing values in the data set "Mushroom". For simplicity, we denote the nine treated data sets as the first group. However, the data sets in the first group are only suitable for the relations *TR*, *SR* and *LR*. With respect to the relation *CR*, followed by the data sets in the first group, we randomly choose a part of the incomplete data for each data set in first group and utilize "?" to replace "*", respectively. Similarly, we denote the nine further treated data sets as the second group.

Our experiment's process is divided into two scenarios. First, we consider the knowledge updating procedure when adding objects. Second, we consider the knowledge updating procedure when deleting objects.

The strategies of the experiments are based on the 10-cross validation method. The original data in each dataset are equally divided into ten parts. In the first scenario, we randomly choose nine of ten parts as the training data at time t, and the remaining part is considered as the immigrating objects which will enter into the system at time $t+1$. In the second scenario, we choose the ten parts data as the training data at time t, and we randomly choose one part from the ten parts at time t as the emigrating objects which will get out of the system at time $t+1$.

For simplicity, we denote the algorithm proposed in our paper as "incremental algorithm" (see Algorithm 1 and Algorithm 2). To compare the efficiency of our algorithm, we utilize the non-incremental algorithm as the reference algorithm. The non-incremental algorithm means we treat the updating data by variation of objects as absolutely new data without using any incremental strategy and directly compute the accuracy matrix and the coverage matrix at time $t+1$. In order to validate our algorithm, we utilize the "10-cross validation" method to estimate the elapsed times for the two different algorithms. We calculate the average elapsed times by repeating the computing process for 100 times when fixing a pair of threshold values α and β (here we set $\alpha=0.6$ and $\beta=0.1$). Considered by the binary relations *TR*, *SR*, *LR* and *CR* in IIS, we repeat our experiments four times for all nine databases by using "incremental algorithm" and "non-incremental algorithm" with *TR*, *SR*, *LR* and *CR*. The experimental results when adding objects are shown in Table 5, and the experimental results when deleting objects are shown in Table 6.

From Table 5 and Table 6, we see that the "incremental algorithm" is much faster than "non-incremental algorithm" on knowledge updating in all nine data sets with four binary relations *TR*, *SR*, *LR* and *CR*. We find the "incremental algorithm" is effective for inducing knowledge in dynamic information systems, especially for the complex and massive database.

Furthermore, the computational complexity analysis of the proposed algorithms are discussed as follows. As we stated in Section 3, we suppose an original IIS includes U objects, which form m conditional classes and n decision classes at time t. At time $t+1$, we suppose there are M new objects (denote by $U_{im}=\{U_{im}^1, U_{im}^2, \cdots, U_{im}^M\}$) enter the system which form l new conditional classes and r new decision classes. In addition, we suppose there are N objects (denote by $U_{em}=\{U_{em}^1, U_{em}^2, \cdots, U_{em}^N\}$) get out of system, accompanied by l' conditional classes and r' decision classes eliminate from the original IIS. For convenience, we set $m'=m+l$, $n'=n+r$; $m''=m-l'$, $n''=n-r'$.

(I) We consider the time complexity of the algorithms. There are five key steps to calculate the time complexity.

Step 1: The time complexity of calculating condition classes and decision classes;

Step 2: The time complexity of calculating X_i^R which generates from X_i with a relation R;

Step 3: The time complexity of calculating the support matrix $Supp(D|X)$;

Step 4: The time complexity of calculating the accuracy matrix $Acc(D|X)$ and coverage matrix $Cov(D|X)$;

Step 5: The time complexity of outputting rules according to the given α and β.

The time complexity of each step in non-incremental algorithm and incremental algorithm (Algorithm 1 and Algorithm 2) is briefly shown as Table 7.

[1] It can available from http://www.lcb.uu.se/tools/rosetta/.

Table 5
The average elapsed times when adding object sets (unit: seconds).

TR			*SR*		
Date set	Non-incremental algorithm	Incremental algorithm	Date set	Non-incremental algorithm	Incremental algorithm
Iris	1.0793×10^{-3}	4.3497×10^{-4}	Iris	9.6240×10^{-4}	3.7366×10^{-4}
Cup	3.7021×10^{-3}	1.0489×10^{-3}	Cup	3.5049×10^{-3}	1.0267×10^{-3}
Zoo	5.2082×10^{-3}	1.0582×10^{-3}	Zoo	5.1126×10^{-3}	1.0037×10^{-3}
Segment	6.6553×10^{-2}	2.5933×10^{-2}	Segment	6.4010×10^{-2}	2.3756×10^{-2}
Bank	9.0071×10^{-2}	1.6303×10^{-2}	Bank	9.3200×10^{-2}	1.6035×10^{-2}
Tic-tac-toe	1.0890×10^{-1}	2.3375×10^{-2}	Tic-tac-toe	1.1023×10^{-1}	2.3376×10^{-2}
Soybean	1.5158×10^{-1}	2.4977×10^{-2}	Soybean	1.5454×10^{-1}	2.4982×10^{-2}
Car	2.8670×10^{-1}	6.9996×10^{-2}	Car	2.8126×10^{-1}	6.9510×10^{-2}
Mushroom	57.9235	21.8362	Mushroom	58.6076	21.0952
LR			*CR*		
Date Set	Non-incremental algorithm	Incremental algorithm	Date set	Non-incremental algorithm	Incremental algorithm
Iris	1.0966×10^{-3}	4.0019×10^{-4}	Iris	1.1004×10^{-3}	4.1793×10^{-4}
Cup	3.7265×10^{-3}	6.7008×10^{-4}	Cup	3.4642×10^{-3}	8.5461×10^{-4}
Zoo	5.2795×10^{-3}	1.0678×10^{-3}	Zoo	4.8596×10^{-3}	1.0370×10^{-3}
Segment	6.6980×10^{-2}	2.5500×10^{-2}	Segment	6.4614×10^{-2}	2.5794×10^{-2}
Bank	9.1006×10^{-2}	1.6298×10^{-2}	Bank	8.3274×10^{-2}	1.5546×10^{-2}
Tic-tac-toe	1.0983×10^{-1}	2.3543×10^{-2}	Tic-tac-toe	9.9035×10^{-2}	2.3126×10^{-2}
Soybean	1.5052×10^{-1}	2.5000×10^{-2}	Soybean	1.6587×10^{-1}	2.7245×10^{-2}
Car	2.8910×10^{-1}	7.0113×10^{-2}	Car	2.6820×10^{-1}	7.0184×10^{-2}
Mushroom	59.5748	21.4022	Mushroom	57.5214	21.2461

Table 6
The average elapsed times when deleting object sets (unit: seconds).

TR			*SR*		
Date set	Non-incremental algorithm	Incremental algorithm	Date set	Non-incremental algorithm	Incremental algorithm
Iris	9.8610×10^{-4}	3.5161×10^{-4}	Iris	8.4803×10^{-4}	3.0678×10^{-4}
Cup	3.3127×10^{-3}	8.2259×10^{-4}	Cup	3.1691×10^{-3}	7.5396×10^{-4}
Zoo	4.7771×10^{-3}	7.4004×10^{-4}	Zoo	4.5968×10^{-3}	6.9419×10^{-4}
Segment	6.9700×10^{-2}	2.1834×10^{-2}	Segment	5.9836×10^{-2}	2.0113×10^{-2}
Bank	8.0622×10^{-2}	9.9092×10^{-3}	Bank	8.3037×10^{-2}	9.5841×10^{-3}
Tic-tac-toe	9.1863×10^{-2}	1.6177×10^{-2}	Tic-tac-toe	9.1582×10^{-2}	1.6380×10^{-2}
Soybean	1.3564×10^{-1}	1.5765×10^{-2}	Soybean	1.3865×10^{-1}	1.5527×10^{-2}
Car	2.5564×10^{-1}	5.0304×10^{-2}	Car	2.5182×10^{-1}	5.0848×10^{-2}
Mushroom	38.4795	10.9179	Mushroom	38.1016	10.9485
LR			*CR*		
Date set	Non-incremental algorithm	Incremental algorithm	Date set	Non-incremental algorithm	Incremental algorithm
Iris	9.6314×10^{-4}	3.5693×10^{-4}	Iris	9.8424×10^{-4}	3.5661×10^{-4}
Cup	3.3614×10^{-3}	8.4121×10^{-4}	Cup	3.0655×10^{-3}	8.4709×10^{-4}
Zoo	5.2032×10^{-3}	7.5169×10^{-4}	Zoo	4.2482×10^{-3}	7.5344×10^{-4}
Segment	7.5065×10^{-2}	2.1876×10^{-2}	Segment	6.0999×10^{-2}	2.2290×10^{-2}
Bank	8.6029×10^{-2}	9.7884×10^{-3}	Bank	7.3181×10^{-2}	9.8418×10^{-3}
Tic-tac-toe	9.1636×10^{-2}	1.5963×10^{-2}	Tic-tac-toe	8.8071×10^{-2}	1.6218×10^{-2}
Soybean	1.3478×10^{-1}	3.2826×10^{-2}	Soybean	1.5123×10^{-1}	1.8143×10^{-2}
Car	2.5700×10^{-1}	5.0534×10^{-2}	Car	2.3688×10^{-1}	6.4228×10^{-2}
Mushroom	38.2455	10.9902	Mushroom	38.0469	10.9208

In Table 7, Step 1 and Step 2 are the key steps in the algorithms. In Step 1, the time complexity of Algorithm 1 is $O(|C| \cdot |U_{im}|) + O(|D| \cdot |U_{im}|)$, and the time complexity of Algorithm 2 is $O(|C| \cdot |U_{em}|) + O(|D| \cdot |U_{em}|)$. They are both much lower than that of non-incremental algorithm. Analogously, we can get the similar conclusions when comparing with our incremental algorithms and non-incremental algorithm in Step 2. Hence, one can draw a conclusion that the incremental algorithms proposed in this paper may significantly reduce the computational time for updating knowledge from IIS.

(II) We consider the space complexity of the algorithms. There are four key steps to calculate the space complexity.

Step 1: Storing the condition classes;

Step 2: Storing the decision classes;

Table 7
The time complexities description.

Adding object sets		
Steps	Non-incremental algorithm	Incremental algorithm (Algorithm 1)
Step 1	$O(\lvert C\rvert \cdot \lvert U\cup U_{im}\rvert)+O(\lvert D\rvert \cdot \lvert U\cup U_{im}\rvert)$	$O(\lvert C\rvert \cdot \lvert U_{im}\rvert)+O(\lvert D\rvert \cdot \lvert U_{im}\rvert)$
Step 2	$O(m'^2\lvert C\rvert)$	$O(m' \cdot l \cdot \lvert C\rvert)$
Step 3	$O(\lvert U\cup U_{im}\rvert + \sum_{i=1}^{m'} \lvert X_i^R\rvert)$	$O(\lvert U\cup U_{im}\rvert + \sum_{i=1}^{m'} \lvert X_i^R\rvert)$
Step 4	$O(2\cdot m' \cdot n')$	$O(2\cdot m' \cdot n')$
Step 5	$O(m' \cdot n')$	$O(m' \cdot n')$
Deleting object sets		
Steps	Non-incremental algorithm	Incremental algorithm (Algorithm 2)
Step 1	$O(\lvert C\rvert \cdot \lvert U-U_{em}\rvert)+O(\lvert D\rvert \cdot \lvert U-U_{em}\rvert)$	$O(\lvert C\rvert \cdot \lvert U_{em}\rvert)+O(\lvert D\rvert \cdot \lvert U_{em}\rvert)$
Step 2	$O(m''^2\lvert C\rvert)$	$O(m'' \cdot l' \cdot \lvert C\rvert)$
Step 3	$O(\lvert U-U_{em}\rvert + \sum_{i=1}^{m''} \lvert X_i^R\rvert)$	$O(\lvert U-U_{em}\rvert + \sum_{i=1}^{m''} \lvert X_i^R\rvert)$
Step 4	$O(2\cdot m'' \cdot n'')$	$O(2\cdot m'' \cdot n'')$
Step 5	$O(m'' \cdot n'')$	$O(m'' \cdot n'')$

Table 8
The space complexities description.

Adding object sets		
Steps	Non-incremental algorithm	Incremental algorithm (Algorithm 1)
Step 1	$O(m' \cdot \lvert C\rvert)$	$O(l \cdot \lvert C\rvert)$
Step 2	$O(n' \cdot \lvert D\rvert)$	$O(r \cdot \lvert D\rvert)$
Step 3	$O(m' \cdot \lvert C\rvert + \sum_{i=1}^{m'} \lvert X_i^R\rvert)$	$O(l \cdot \lvert C\rvert + \sum_{i=m+1}^{m'} \lvert X_i^R\rvert)$
Step 4	$O(3\cdot m' \cdot n')$	$O(3\cdot m' \cdot n')$
Deleting object sets		
Steps	Non-incremental algorithm	Incremental algorithm (Algorithm 2)
Step 1	$O(m'' \cdot \lvert C\rvert)$	–
Step 2	$O(n'' \cdot \lvert D\rvert)$	–
Step 3	$O(m'' \cdot \lvert C\rvert + \sum_{i=1}^{m''} \lvert X_i^R\rvert)$	–
Step 4	$O(3\cdot m'' \cdot n'')$	$O(3\cdot m'' \cdot n'')$

Note: "–" represents it needs no extra space.

Step 3: Storing the classes X_i^R which generates from X_i with a relation R;
Step 4: Storing the three matrices of $Supp(D|X)$, $Acc(D|X)$ and $Cov(D|X)$.

The space complexity of each step in non-incremental algorithm and incremental algorithm (Algorithm 1 and Algorithm 2) is briefly shown in Table 8.

In Table 8, the space complexity of Algorithm 1 is significantly reduced comparing with the non-incremental algorithm, which is because it is only considered on the new generated classes in Step 1–Step 3. Meanwhile, the space complexity of Algorithm 2 is significantly reduced comparing with the non-incremental algorithm. It doesn't require extra space for original classes in Step 1–Step 3 since the IIS does not generate new classes. Hence, it is concluded that the incremental algorithms proposed in this paper may significantly reduce the computational space for updating knowledge from IIS.

4.2.2. The experimental performances under "mushroom" with increasing the size of data

Furthermore, we utilize the "mushroom" to compare the two algorithms when gradually increasing the size of data. To achieve our goal, we divide the 8124 samples into ten parts of equal size (approximate 800 samples for each size). The first part is regarded as the 1st data set, the combination of the first part and the second part is viewed as the 2nd data set, the combination of the 2nd data set and the third part is regarded as the 3rd data set, $\cdots$, the combination of all ten parts is viewed as the 10th data set as shown in [65]. Then, we choose a small part of data (approximate 1.5% data) in each data set as the changing objects, respectively. When we consider the scenario on adding objects, we use the remaining part for each data set as the original data, and the changing objects are considered as the immigrating objects which will enter into the system at time $t+1$. Similarly, when we consider the scenario on deleting objects, we choose all the data in each data set as the original data, and the changing objects are considered as the emigrating objects which will get out of the system at time $t+1$. The experimental strategies on adding and deleting objects are outlined in Table 9. In addition, with the increase of data size in "mushroom", the experimental results of two algorithms when adding objects under four binary relations are shown in Table 10 and Fig. 3. The experimental results of two algorithms when deleting objects under four binary relations are shown in Table 11 and Fig. 4.

In Figs. 3 and 4, the x-coordinate pertains to the size of the data set (the 10 data sets starting from the smallest one), while the y-coordinate concerns the computational time. Tables 10 and 11 show the comparisons of computational time between the "non-incremental algorithm" and "incremental algorithm", while Figs. 3 and 4 illustrate the trend of each of

Table 9
The experimental strategies on adding and deleting objects in mushroom data set.

Data set name	Ratio of size	Adding process		Deleting process	
		Original data	Immigrating data	Original data	Emigrating data
1st data set	10%	800	12	812	12
2nd data set	20%	1600	24	1624	24
3rd data set	30%	2400	36	2436	36
4th data set	40%	3200	48	3248	48
5th data set	50%	4000	60	4060	60
6th data set	60%	4800	72	4872	72
7th data set	70%	5600	84	5684	84
8th data set	80%	6400	96	6496	96
9th data set	90%	7200	108	7308	108
10th data set	100%	8000	124	8124	124

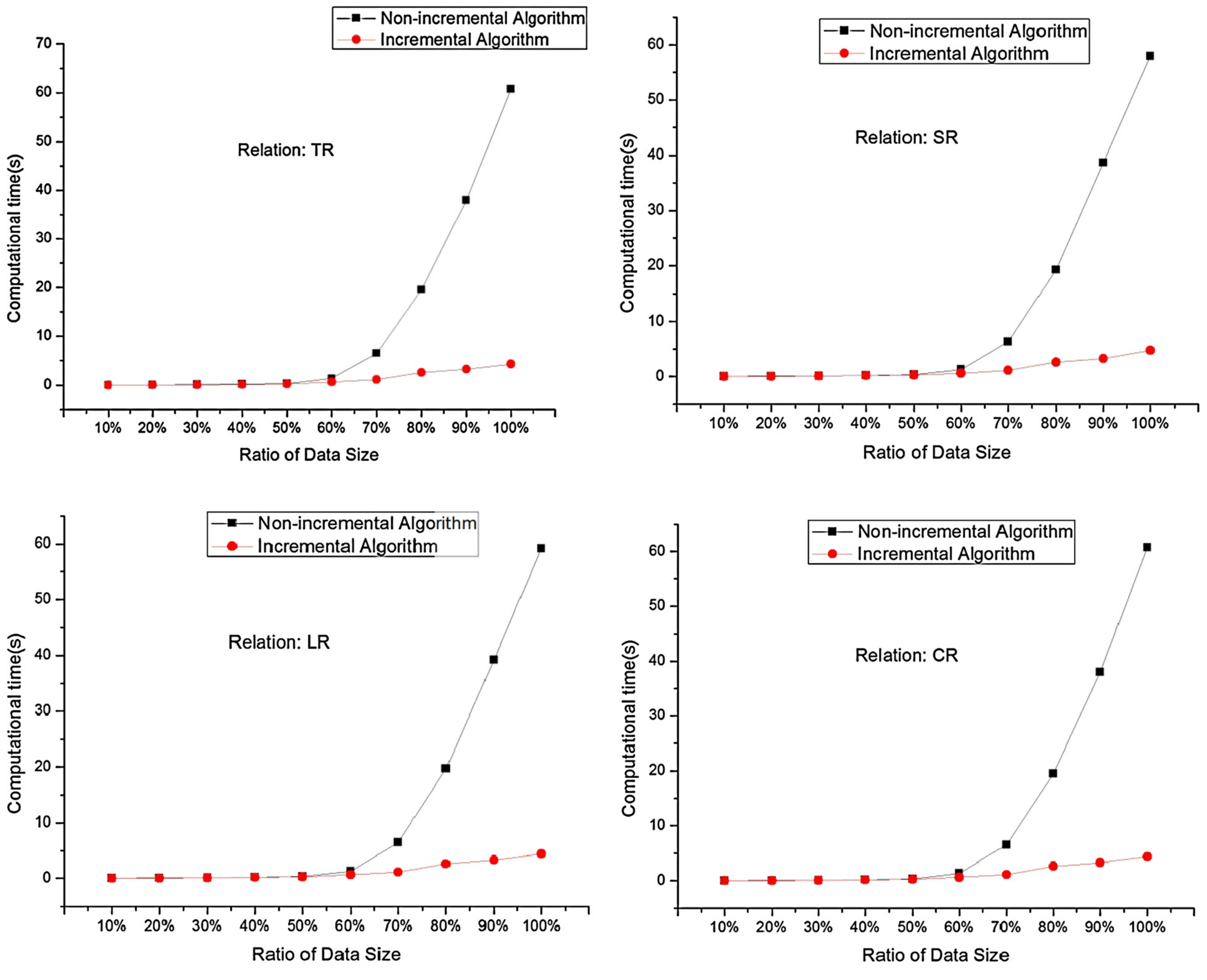

Fig. 3. The elapsed time of two algorithms with adding objects under increasing the size of mushroom data set.

the two algorithms with the increasing size of the data set. According to Tables 10, 11 and Figs. 3, 4, we find “incremental algorithm” is effective and performs much better than the “non-incremental algorithm” under all four relations *TR*, *SR*, *LR* and *CR*, especially for the increasing of the ratio of data size in “Mushroom”.

4.2.3. The experimental performances under a large data set

For the following evaluations, we utilize a large data set KDDCup-99 from the machine learning data repository, University of California at Irvine. The data set KDDCup-99 consists of approximately five million records. Each records consists of 1 decision attribute and 41 condition attributes, where 6 are categorical and 35 are numeric. In our experiments, we choose

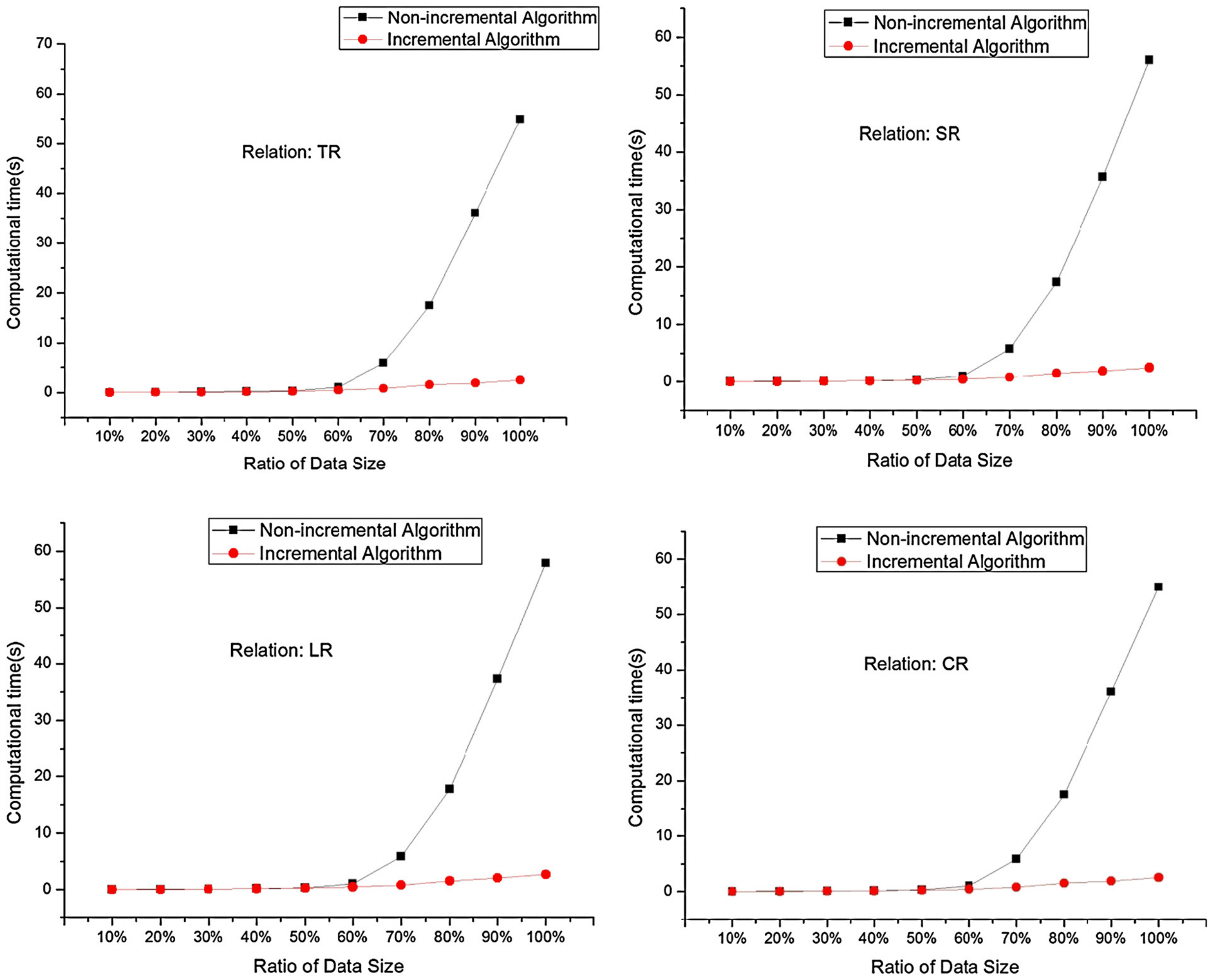

Fig. 4. The elapsed time of two algorithms with deleting objects under increasing the size of mushroom data set.

400 000 records (about 10% of KDDCup-99) as the benchmark data set. We set $\alpha = 0.6$, $\beta = 0.3$ and the missing rate of the data set is 5% ($missing\ rate = \frac{missing\ number\ of\ data}{total\ number\ of\ data}$. Here, we suppose the missing value for each object in KDDCup-99 is no more than one). When consider the scenario on adding objects, we choose the 400 000 records as the original data, and randomly choose some other records in KDDCup-99 as the immigrating objects. When consider the scenario on deleting objects, we also choose the 400 000 records as the original data, and randomly choose a part of data in 400 000 records as the emigrating objects. Then, we do 5 groups of experiments, namely, randomly add or delete 1000, 2000, 3000, 4000 and 5000 records, respectively. Considered by the four binary relations, we calculate the average elapsed times by repeating the computing process for 10 times. The experimental results with two algorithms are shown in Table 12.

In Table 12, we find "incremental algorithm" performs much better than the "non-incremental algorithm" under all four relations, which indicates our proposed algorithms are also effective for big data analysis.

Furthermore, we consider the classification accuracy when adding and deleting objects under the four relations. In [54], Mesterharm et al. used J48, SVM and their proposed active learning algorithms to calculate the error rates of KDDCup-99. Followed the strategies in their experiments, we randomly choose 400 000 records of KDDCup-99 as the benchmark data set and set $\alpha = 0.6$, $\beta = 0.3$. In order to investigate the relation between percentage of missing data and classification accuracy, we choose three groups of missing rates, namely, 5%, 10% and 20%. For each certain missing rate, we randomly add (delete) 1000, 2000, 3000, 4000, 5000 records to (from) the benchmark data set, respectively. In each experiment, we repeat the adding (deleting) process 100 times. We utilize the average classification accuracy to evaluate the performance of our method. Table 13 outlines the performances for the average classification accuracy with different missing rates.

In Table 13, we find the average classification accuracy under the four relations all keep steady and robust when the missing rate changes.

Table 10
The elapsed time of two algorithms with adding objects under increasing the size of mushroom data set (unit: seconds).

TR			*SR*		
Ratio	Non-incremental algorithm	Incremental algorithm	Ratio	Non-incremental algorithm	Incremental algorithm
10%	2.4484×10^{-2}	1.0763×10^{-2}	10%	2.5565×10^{-2}	1.0888×10^{-2}
20%	7.1055×10^{-2}	4.0999×10^{-2}	20%	7.0530×10^{-2}	4.0778×10^{-2}
30%	1.3383×10^{-1}	8.9646×10^{-2}	30%	1.3378×10^{-1}	8.9966×10^{-2}
40%	2.2005×10^{-1}	1.5666×10^{-1}	40%	2.1969×10^{-1}	1.5634×10^{-1}
50%	3.7580×10^{-1}	2.7782×10^{-1}	50%	3.5588×10^{-1}	2.6654×10^{-1}
60%	1.3011	6.0260×10^{-1}	60%	1.2888	6.0990×10^{-1}
70%	6.4371	1.1523	70%	6.3311	1.1478
80%	20.5697	2.7650	80%	19.2620	2.5854
90%	38.7326	3.3390	90%	38.6806	3.2509
100%	61.3899	4.6287	100%	57.8930	4.7673
LR			*CR*		
Ratio	Non-incremental algorithm	Incremental algorithm	Ratio	Non-incremental algorithm	Incremental algorithm
10%	2.4713×10^{-2}	1.1837×10^{-2}	10%	2.4987×10^{-2}	1.0938×10^{-2}
20%	6.9879×10^{-2}	4.0656×10^{-2}	20%	7.1093×10^{-2}	4.0717×10^{-2}
30%	1.3376×10^{-1}	8.9359×10^{-2}	30%	1.3325×10^{-1}	8.9216×10^{-2}
40%	2.1802×10^{-1}	1.5643×10^{-1}	40%	2.1860×10^{-1}	1.5655×10^{-1}
50%	3.5139×10^{-1}	2.5572×10^{-1}	50%	3.5588×10^{-1}	2.6654×10^{-1}
60%	1.3145	6.3054×10^{-1}	60%	1.3711	6.2049×10^{-1}
70%	6.5012	1.1553	70%	6.5015	1.1566
80%	19.6829	2.6128	80%	19.5109	2.5949
90%	39.1922	3.3157	90%	38.011	3.2531
100%	59.1536	4.4193	100%	60.7224	4.3107

Table 11
The elapsed time of two algorithms with deleting objects under increasing the size of mushroom data set (unit: seconds).

TR			*SR*		
Ratio	Non-incremental algorithm	Incremental algorithm	Ratio	Non-incremental algorithm	Incremental algorithm
10%	2.4783×10^{-2}	1.0389×10^{-2}	10%	2.485×10^{-2}	1.0439×10^{-2}
20%	6.9738×10^{-2}	3.9387×10^{-2}	20%	6.8550×10^{-2}	3.8823×10^{-2}
30%	1.3762×10^{-1}	8.668×10^{-2}	30%	1.3016×10^{-1}	8.6806×10^{-2}
40%	2.1240×10^{-1}	1.5122×10^{-1}	40%	2.1494×10^{-1}	1.5158×10^{-1}
50%	3.2635×10^{-1}	2.4209×10^{-1}	50%	3.2580×10^{-1}	2.4052×10^{-1}
60%	1.0517	4.7659×10^{-1}	60%	1.0132	4.6000×10^{-1}
70%	5.7405	7.8643×10^{-1}	70%	5.7901	7.8958×10^{-1}
80%	17.6425	1.5535	80%	17.3138	1.5383
90%	36.9641	1.9871	90%	35.6518	1.9224
100%	58.3145	2.54774	100%	56.0114	2.50427
LR			*CR*		
Ratio	Non-incremental algorithm	Incremental algorithm	Ratio	Non-incremental algorithm	Incremental algorithm
10%	2.4184×10^{-2}	1.0500×10^{-2}	10%	2.4668×10^{-2}	1.0380×10^{-2}
20%	6.8321×10^{-2}	3.8839×10^{-2}	20%	7.0454×10^{-2}	3.9253×10^{-2}
30%	1.2973×10^{-1}	8.6416×10^{-2}	30%	1.2983×10^{-1}	8.7128×10^{-2}
40%	2.1163×10^{-1}	1.5170×10^{-1}	40%	2.1306×10^{-1}	1.5382×10^{-1}
50%	3.2447×10^{-1}	2.3983×10^{-1}	50%	3.2420×10^{-1}	2.4010×10^{-1}
60%	1.0250	4.6240×10^{-1}	60%	1.0877	4.5731×10^{-1}
70%	5.8273	7.9832×10^{-1}	70%	5.8746	7.9012×10^{-1}
80%	17.7246	1.55403	80%	17.5231	1.5507
90%	37.3236	2.04492	90%	36.0923	1.9272
100%	57.8806	2.6732	100%	54.9112	2.5199

5. Conclusion

In this paper, the rough set based incremental approaches were proposed to deal with the knowledge discovery problem in dynamic IIS. The knowledge was defined based on both the accuracy and the coverage. A matrix based incremental approach with support, accuracy and coverage was constructed to illustrate the incremental process by the binary relations *TR*, *SR*, *LR* and *CR* in IIS. Compared with the knowledge updating strategies in complete information systems, the incremental model and algorithm to obtain knowledge in IIS was discussed. An example of medical diagnosis, the experimental

Table 12
The elapsed time of two algorithms with adding and deleting objects under KDDCup-99 (unit: seconds).

TR			*TR*		
Adding records	Non-incremental algorithm	Incremental algorithm	Deleting records	Non-incremental algorithm	Incremental algorithm
1000	1385.250	686.360	1000	1290.190	605.081
2000	1363.770	706.870	2000	1339.970	643.744
3000	1450.710	739.175	3000	1314.270	676.145
4000	1407.000	870.353	4000	1222.730	749.951
5000	1543.600	984.792	5000	1263.330	781.481
SR			*SR*		
Adding records	Non-incremental algorithm	Incremental algorithm	Deleting records	Non-incremental algorithm	Incremental algorithm
1000	1271.060	550.653	1000	1116.230	485.184
2000	1213.680	670.036	2000	1103.860	502.287
3000	1230.720	736.067	3000	1098.180	515.918
4000	1251.290	808.973	4000	1131.670	565.999
5000	1317.310	842.584	5000	1176.340	593.308
LR			*LR*		
Adding records	Non-incremental algorithm	Incremental algorithm	Deleting records	Non-incremental algorithm	Incremental algorithm
1000	1468.470	550.653	1000	1524.360	660.826
2000	1334.470	692.546	2000	1474.020	707.185
3000	1363.470	715.128	3000	1301.520	779.969
4000	1440.340	815.546	4000	1230.910	832.831
5000	1430.100	860.691	5000	1209.080	867.68
CR			*CR*		
Adding records	Non-incremental algorithm	Incremental algorithm	Deleting records	Non-incremental algorithm	Incremental algorithm
1000	1390.200	657.804	1000	1290.990	664.677
2000	1456.970	747.326	2000	1371.300	708.019
3000	1450.710	876.608	3000	1294.060	798.694
4000	1499.140	883.457	4000	1286.660	798.602
5000	1509.770	981.371	5000	1281.300	877.692

Table 13
The performances for the average classification accuracy with different missing rates.

Missing rate: 5%									
Adding records	*TR*	*SR*	*LR*	*CR*	Deleting records	*TR*	*SR*	*LR*	*CR*
1000	0.998265	0.998269	0.998252	0.998264	1000	0.998286	0.998296	0.998304	0.998311
2000	0.998272	0.998263	0.998276	0.998268	2000	0.998289	0.998300	0.998304	0.998315
3000	0.998275	0.998267	0.998284	0.998272	3000	0.998312	0.998336	0.998322	0.998322
4000	0.998294	0.998281	0.998296	0.998288	4000	0.999336	0.999342	0.999340	0.999328
5000	0.998308	0.998292	0.998313	0.998295	5000	0.999994	0.999997	0.999992	0.999988
Missing rate: 10%									
Adding records	*TR*	*SR*	*LR*	*CR*	Deleting records	*TR*	*SR*	*LR*	*CR*
1000	0.998242	0.998258	0.998262	0.998237	1000	0.998260	0.998273	0.998282	0.998230
2000	0.998306	0.998264	0.998265	0.998242	2000	0.998276	0.998269	0.998276	0.998223
3000	0.998310	0.998271	0.998274	0.998247	3000	0.998265	0.998261	0.998266	0.998218
4000	0.998314	0.998286	0.998280	0.998253	4000	0.999290	0.999151	0.999124	0.999064
5000	0.998319	0.998292	0.998288	0.998258	5000	0.999992	0.999998	0.999995	0.999986
Missing rate: 20%									
Adding records	*TR*	*SR*	*LR*	*CR*	Deleting records	*TR*	*SR*	*LR*	*CR*
1000	0.998235	0.998246	0.998237	0.998216	1000	0.998231	0.998257	0.998246	0.998216
2000	0.998244	0.998254	0.998244	0.998227	2000	0.998229	0.998268	0.998228	0.998211
3000	0.998249	0.998272	0.998251	0.998238	3000	0.998224	0.998285	0.998223	0.998205
4000	0.998252	0.998278	0.998258	0.998245	4000	0.999283	0.999302	0.999277	0.999268
5000	0.998259	0.998296	0.998262	0.998250	5000	0.999990	0.999997	0.999987	0.999985

evaluations with nine UCI datasets and a big dataset with millions of records were provided to validate the rationality and efficiency of our method. The incremental algorithm proposed in our paper can significantly reduce the average computational time under all four binary relations *TR*, *SR*, *LR* and *CR* in IIS. Our future work will focus on extensions of the current approach to neighborhood information systems, fuzzy information systems and other generalized rough sets and develop approaches to find the minimal patterns of updating knowledge with rough set theory.

Acknowledgements

This work is supported by the National Science Foundation of China (Nos. 71201133, 61175047, 71090402, 71110107027), the Youth Social Science Foundation of the Chinese Education Commission (11YJC630127), the Research Fund for the Doctoral Program of Higher Education of China (No. 20120184120028), the China Postdoctoral Science Foundation (Nos. 2012M520310, 2013T60132) and the Fundamental Research Funds for the Central Universities of China (No. SWJTU12CX117).

References

[1] W.C. Bang, Z. Bien, New incremental learning algorithm in the framework of rough set theory, Int. J. Fuzzy Syst. 1 (1999) 25–36.
[2] J. Bazan, H.S. Nguyen, S.H. Nguyen, P. Synak, J. Wroblewski, Rough set algorithms in classification problems, in: Polkowski, et al. (Eds.), Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems, in: Studies in Fuzziness and Soft Computing, 2000, pp. 49–88.
[3] J. Blaszczynski, R. Slowinski, Incremental induction of decision rules from dominance-based rough approximations, Electron. Notes Theor. Comput. Sci. 82 (2003) 40–51.
[4] G. Cattaneo, D. Ciucci, Investigation about time monotonicity of similarity and preclusive rough approximations in incomplete information systems, in: Proceedings of 4th International Conference on Rough Sets and Current Trends in Computing, in: LNAI, vol. 3066, 2004, pp. 38–48.
[5] C.C. Chan, Incremental learning of production rules from examples under uncertainty: a rough set approach, Int. J. Softw. Eng. Knowl. Eng. 1 (1991) 439–461.
[6] C.C. Chan, A rough set approach to attribute generalization in data mining, Inf. Sci. 107 (1998) 177–194.
[7] H.M. Chen, T.R. Li, S.J. Qiao, D. Ruan, A rough set based dynamic maintenance approach for approximations in coarsening and refining attribute values, Int. J. Intell. Syst. 25 (2010) 1005–1026.
[8] H.M. Chen, T.R. Li, D. Ruan, J.H. Lin, C.X. Hu, A rough-set based incremental approach for updating approximations under dynamic maintenance environments, IEEE Trans. Knowl. Data Eng. 25 (2013) 274–284.
[9] Y. Cheng, The incremental method for fast computing the rough fuzzy approximations, Data Knowl. Eng. 70 (2011) 84–100.
[10] D. Ciucci, Classification of dynamics in rough sets, in: Proceedings of 7th International Conference on Rough Sets and Current Trends in Computing, in: LNAI, vol. 6086, 2010, pp. 257–266.
[11] D. Ciucci, Temporal dynamics in rough sets based on covering, in: Proceedings of 5th International Conference on Rough Sets and Knowledge Technology, in: LNAI, vol. 6401, 2010, pp. 126–133.
[12] D. Ciucci, Attribute dynamics in rough sets, in: Proceedings of ISMIS11, in: LNCS, vol. 6804, 2011, pp. 43–51.
[13] D. Ciucci, Temporal dynamics in information tables, Fundam. Inform. 115 (2012) 57–74.
[14] Y. Fan, T. Tseng, C. Chern, C. Huang, Rule induction based on an incremental rough set, Expert Syst. Appl. 36 (2009) 11439–11450.
[15] J.W. Grzymala-Busse, Knowledge acquisition under uncertainty – a rough set approach, J. Intell. Robot. Syst. 1 (1988) 3–16.
[16] J.W. Grzymala-Busse, A new version of the rule induction system LERS, Fundam. Inform. 31 (1997) 27–39.
[17] J.W. Grzymala-Busse, S. Siddhaye, Rough set approaches to rule induction from incomplete data, in: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, 2004, pp. 923–930.
[18] J.W. Grzymala-Busse, Data with missing attribute values: generalization of indiscernibility relation and rule induction, Trans. Rough Sets I (2004) 78–95.
[19] J.W. Grzymala-Busse, Characteristic relations for incomplete data: a generalization of the indiscernibility relation, Trans. Rough Sets IV (2005) 58–68.
[20] J.W. Grzymala-Busse, Three approaches to missing attribute values: a rough set perspective, Stud. Comput. Intell. 118 (2008) 139–152.
[21] J.W. Grzymala-Busse, Inducing better rule sets by adding missing attribute values, in: Proceedings of 6th International Conference on Rough Sets and Current Trends in Computing, in: LNAI, vol. 5306, 2008, pp. 160–169.
[22] J.W. Grzymala-Busse, W.J. Grzymala-Busse, Increasing incompleteness of data sets – a strategy for inducing better rule sets, Stud. Comput. Intell. 262 (2010) 345–365.
[23] S. Guo, Z.Y. Wang, Z.C. Wu, H.P. Yam, A novel dynamic incremental rule extraction algorithm based on rough set theory, in: Proceedings of the 4th International Conference on Machine Learning and Cybernetics, IEEE Press, 2005, pp. 1902–1907.
[24] F. Hu, G.Y. Wang, H. Huang, Y. Wu, Incremental attribute reduction based on elementary sets, in: RSFDGrC2005, in: LNAI, vol. 3641, 2005, pp. 185–193.
[25] Y.C. Hu, G.H. Tzeng, C.M. Chen, Deriving two-stage learning sequences from knowledge in fuzzy sequential pattern mining, Inf. Sci. 159 (2004) 69–86.
[26] J. Han, M. Kamber, Data Mining: Concepts and Techniques, second ed., Morgan Kaufman, 2006.
[27] P.Y. Hsu, Y.L. Chen, C.C. Ling, Algorithms for mining association rules in bag databases, Inf. Sci. 166 (2004) 31–47.
[28] J.G. Ibrahim, M.H. Chen, S.R. Lipsitz, A.H. Herring, Missing data methods for generalized linear models: a comparative review, J. Am. Stat. Assoc. 100 (2005) 332–346.
[29] M. Khan, M. Banerjee, Formal reasoning with rough sets in multiple-source approximation systems, Int. J. Approx. Reason. 49 (2008) 466–477.
[30] M. Kryszkiewicz, Rough set approach to incomplete information system, Inf. Sci. 112 (1998) 39–49.
[31] M. Kryszkiewicz, Rules in incomplete information systems, Inf. Sci. 113 (1999) 271–292.
[32] K. Lagus, S. Kaski, T. Kohonen, Mining massive document collections by the WEBSOM method, Inf. Sci. 163 (2004) 135–156.
[33] Y. Leung, W.Z. Wu, W.X. Zhang, Knowledge acquisition in incomplete information systems: a rough set approach, Eur. J. Oper. Res. 168 (2006) 164–180.
[34] H.X. Li, M.H. Wang, X.Z. Zhou, J.B. Zhao, An interval set model for learning rules from incomplete information table, Int. J. Approx. Reason. 53 (2012) 24–37.
[35] J.H. Li, C.L. Mei, Y.J. Lv, Incomplete decision contexts: approximate concept construction, rule acquisition and knowledge reduction, Int. J. Approx. Reason. 54 (2013) 149–165.
[36] S.Y. Li, T.R. Li, D. Liu, Dynamic maintenance of approximations in dominance-based rough set approach under the variation of the object set, Int. J. Intell. Syst. 28 (2013) 729–751.
[37] S.Y. Li, T.R. Li, D. Liu, Incremental updating approximations in dominance-based rough sets approach under the variation of the attribute set, Knowl.-Based Syst. 40 (2013) 17–26.

[38] T.R. Li, N. Yang, Y. Xu, J. Ma, An incremental algorithm for mining classification rules in incomplete information systems, in: Proceedings of NAFIPS, IEEE Press, 2004, pp. 446–449.
[39] T.R. Li, D. Ruan, W. Geert, J. Song, Y. Xu, A rough sets based characteristic relation approach for dynamic attribute generalization in data mining, Knowl.-Based Syst. 20 (2007) 485–494.
[40] Y. Li, S. Zhu, X.S. Wang, S. Jajodia, Looking into the seeds of time: discovering temporal patterns in large transaction sets, Inf. Sci. 176 (2006) 1003–1031.
[41] W.J. Lipski, On semantic issues connected with incomplete information databases, ACM Trans. Database Syst. 4 (1979) 262–296.
[42] D. Liu, P. Hu, C.Z. Jiang, Incremental learning methodology of VPRS based on complete information system, in: Proceedings of 2010 International Conference on Rough Sets and Knowledge Technology, 2008, pp. 276–283.
[43] D. Liu, T.R. Li, D. Ruan, W.L. Zou, An incremental approach for inducing knowledge from dynamic information systems, Fundam. Inform. 94 (2009) 245–260.
[44] D. Liu, T.R. Li, D. Ruan, J.B. Zhang, Incremental learning optimization on knowledge discovery in dynamic business intelligent systems, J. Glob. Optim. 51 (2011) 325–344.
[45] D. Liu, T.R. Li, H.M. Chen, X.L. Ji, Approaches to knowledge incremental learning based on the changes of attribute values, in: Proceedings of the 4th International Conference on Intelligent Systems and Knowledge Engineering, 2009, pp. 94–99.
[46] D. Liu, T.R. Li, G.R. Liu, P. Hu, An approach for inducing interesting incremental knowledge based on the change of attribute values, in: Proceedings of 2009 IEEE International Conference on Granular Computing, 2009, pp. 415–418.
[47] D. Liu, J.B. Zhang, T.R. Li, A probabilistic rough set approach for incremental learning knowledge on the change of attribute, in: Proceedings of 2010 International Conference on Foundations and Applications of Computational Intelligence, 2010, pp. 722–727.
[48] D. Liu, T.R. Li, J.B. Zhang, An incremental approach for rule induction under coarsening and refining of attribute values in E-business systems, in: Proceedings of 2010 International Conference on Electronic-Business Intelligence, 2010, pp. 541–547.
[49] D. Liu, T.R. Li, D. Ruan, Probabilistic model criteria with decision-theoretic rough sets, Inf. Sci. 181 (2011) 3709–3722.
[50] D. Liu, D.C. Liang, Incremental learning researches on rough set theory: status and future, Int. J. Rough Sets. Data Anal. 1 (2014) 99–112.
[51] D. Liu, T.R. Li, D.C. Liang, Incorporating logistic regression to decision-theoretic rough sets for classifications, Int. J. Approx. Reason. 55 (2014) 197–210.
[52] C. Luo, T.R. Li, H.M. Chen, D. Liu, Incremental approaches for updating approximations in set-valued ordered information systems, Knowl.-Based Syst. 50 (2013) 218–233.
[53] C. Luo, T.R. Li, J.B. Zhang, Dynamic maintenance of approximations in set-valued ordered decision systems under the attribute generalization, Inf. Sci. 257 (2014) 210–228.
[54] C. Mesterharm, M.J. Pazzani, Active learning using on-line algorithms, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 850–858.
[55] R.S. Michalski, I. Mozetic, J. Hong, N. Lavrac, The multi-purpose incremental learning system AQ15 and its testing application on three medical domains, in: Proceedings of AAAI-86, 1986, pp. 1041–1045.
[56] E. Orlowska, Dynamic information systems, Fundam. Inform. 5 (1982) 101–118.
[57] P. Pagliani, Pretopologies and dynamic spaces, Fundam. Inform. 59 (2004) 221–239.
[58] Z. Pawlak, Rough sets, Int. J. Comput. Inf. Sci. 11 (1982) 341–356.
[59] Z. Pawlak, Rough Sets, Theoretical Aspects of Reasoning about Data, Kluwer Academic Publishers Press, Dordrecht, 1991.
[60] Z. Pawlak, Rough sets, decision algorithm and Bayes' theorem, Eur. J. Oper. Res. 136 (2002) 181–189.
[61] Z. Pawlak, A. Skowron, Rudiments of rough sets, Inf. Sci. 177 (2007) 3–27.
[62] Z. Pawlak, A. Skowron, Rough sets: some extensions, Inf. Sci. 177 (2007) 28–40.
[63] Y.H. Qian, J.Y. Liang, Y.Y. Yao, C.Y. Dang, MGRS: a mulit-granulation rough set, Inf. Sci. 180 (2010) 949–970.
[64] Y.H. Qian, J.Y. Liang, C.Y. Dang, Incomplete multigranulation rough set, IEEE Trans. Syst. Man Cybern., Part A 40 (2010) 420–431.
[65] Y.H. Qian, J.Y. Liang, P. Witold, C.Y. Dang, Positive approximation: an accelerator for attribute reduction in rough set theory, Artif. Intell. 174 (2010) 597–618.
[66] C. Rauszer, Rough logic for multi-agent systems, in: Logic at Work 1992, in: LNCS, vol. 808, 1994, pp. 161–181.
[67] M. Saar-Tsechansky, F. Provost, Handling missing values when applying classification models, J. Mach. Learn. Res. 8 (2007) 1623–1657.
[68] L. Shan, W. Ziarko, Data-based acquisition and incremental modification of classification rules, Comput. Intell. 11 (1995) 357–370.
[69] J. Stefanowski, A. Tsoukias, On the extension of rough sets under incomplete information, in: Proceedings of the 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, 1999, pp. 73–81.
[70] J. Stefanowski, A. Tsoukias, Incomplete information tables and rough classification, Comput. Intell. 17 (2001) 545–566.
[71] L.Y. Tong, L.P. An, An Incremental learning of decision rules based on rough set theory, in: Proceedings of the World Congress on Intelligent Control and Automation, 2002, pp. 420–425.
[72] S. Tsumoto, Extraction of experts' decision process from clinical databases using rough set model, in: Proceedings of PKDD 1997, 1997, pp. 58–67.
[73] S. Tsumoto, Accuracy and coverage in rough set rule induction, in: Proceedings of 3rd International Conference on Rough Sets and Current Trends in Computing, in: LNAI, vol. 2475, 2002, pp. 373–380.
[74] G.Y. Wang, Extension of rough set under incomplete information systems, in: Proceedings of the 2002 IEEE International Conference on Fuzzy System, 2002, pp. 1098–1103.
[75] G.Y. Wang, L.H. Guan, F. Hu, Rough set extensions in incomplete information systems, Front. Electr. Electron. Eng. China 3 (2008) 399–405.
[76] S.K.M. Wong, W. Ziarko, Z. Pawlak, Algorithm for inductive learning, Bull. Pol. Acad. Sci., Tech. Sci. 34 (1986) 271–276.
[77] Y.T. Xu, L.S. Wang, R.Y. Zhang, A dynamic attribute reduction algorithm based on 0–1 integer programming, Knowl.-Based Syst. 24 (2011) 1341–1347.
[78] Y.Y. Yao, Two views of the theory of rough sets in finite universes, Int. J. Approx. Reason. 15 (1996) 291–317.
[79] Y.Y. Yao, Constructive and algebraic methods of the theory of rough sets, Inf. Sci. 109 (1998) 21–47.
[80] J.B. Zhang, T.R. Li, D. Ruan, D. Liu, Neighborhood rough sets for dynamic data mining, Int. J. Comput. Intell. Syst. 27 (2012) 317–342.
[81] J.B. Zhang, T.R. Li, D. Ruan, D. Liu, Rough sets based matrix approaches with dynamic attribute variation in set-valued information systems, Int. J. Approx. Reason. 53 (2012) 620–635.
[82] J.B. Zhang, T.R. Li, D. Ruan, Z.Z. Gao, C.B. Zhao, A parallel method for computing rough set approximations, Inf. Sci. 194 (2012) 209–223.
[83] J.B. Zhang, T.R. Li, H.M. Chen, Composite rough sets for dynamic data mining, Inf. Sci. 257 (2014) 81–100.
[84] J.B. Zhang, J.S. Wong, T.R. Li, Y. Pan, A comparison of parallel large-scale knowledge acquisition using rough set theory on different MapReduce runtime systems, Int. J. Approx. Reason. 55 (2014) 896–907.
[85] Z. Zheng, G.Y. Wang, RRIA: A rough set and rule tree based incremental knowledge acquisition algorithm, Fundam. Inform. 59 (2004) 299–313.