IEICE TRANS. INF. & SYST., VOL.E98–D, NO.10 OCTOBER 2015

PAPER

Hide Association Rules with Fewer Side Effects

Peng CHENG[†,††a)], ***Student Member***, **Ivan LEE**[†††], **Jeng-Shyang PAN**[†], **Chun-Wei LIN**[†], ***and*** **John F. RODDICK**[††††], ***Nonmembers***

SUMMARY Association rule mining is a powerful data mining tool, and it can be used to discover unknown patterns from large volumes of data. However, people often have to face the risk of disclosing sensitive information when data is shared with different organizations. The association rule mining techniques may be improperly used to find sensitive patterns which the owner is unwilling to disclose. One of the great challenges in association rule mining is how to protect the confidentiality of sensitive patterns when data is released. Association rule hiding refers to sanitize a database so that certain sensitive association rules cannot be mined out in the released database. In this study, we proposed a new method which hides sensitive rules by removing some items in a database to reduce the support or confidence levels of sensitive rules below specified thresholds. Based on the information of positive border rules and negative border rules contained in transactions, the proposed method chooses suitable candidates for modification aimed at reducing the side effects and the data distortion degree. Comparative experiments on real datasets and synthetic datasets demonstrate that the proposed method can hide sensitive rules with much fewer side effects and database modifications.

key words: *Association rule hiding, side effects, border rules*

1. Introduction

It is a common practice to share data among different organizations during business collaboration. However, sharing data to the third party brings the risk of disclosing sensitive information contained in it. The data owner is unwilling to disclose the sensitive information due to the concern of individual privacy, business conflicts between competitors or other considerations. Therefore, there is a need to hide information that is considered sensitive before publishing or sharing the data. One way to address this issue is to sanitize the database by changing some values in it.

In this study, we focus on privacy preserving in association rule mining. This can be further divided into privacy protection for the data and privacy protection for the mining results, i.e., frequent patterns or association rules. Agrawal and Srikant [1] proposed an idea of distorting the original data to ensure the privacy of sensitive data while keeping the mining results as close to those of mining the original database as possible. In this paper, on the other hand, we mainly investigate privacy preserving for the mining results. The discovery of association rules from a large database can provide valuable information, such as customer purchasing patterns in supermarkets. At the same time, it also may pose the threat of disclosing sensitive relations to other parties if used inappropriately.

To illustrate the need of sharing data and protecting sensitive knowledge contained it, we present a motivation example borrowed from the work of Verikios et al. [2]. Suppose that we are purchasing directors of a large supermarket chain BigMart. The sales directors of Dedtrees Paper Company, are negotiating with us. They are willing to offer their products at a low price if we agree to share our customer sales data to them. We agree to the data sharing proposal. Then, Dedtrees performs association rule mining on the shared data and finds that people who buy skim milk also buy Green paper. Dedtrees replicates the strategy and decides to run a coupon campaign by providing a reduced price for every purchase of skim milk and Dedtrees paper together. The campaign greatly reduces the sales of Green paper and they raise the prices to us. When Dedtrees has gained the competitive advantage, they become unwilling to provide us their products at a reduced price.

In such circumstances, it is necessary to take some measures to ensure that sensitive patterns in the shared data are not revealed. This is usually accomplished by modifying the data so that the sensitive rules become uninteresting from a data mining point of view. The database may be modified by adding new items or removing existing items so that sensitive knowledge cannot be discovered at some specified interestingness thresholds.

However, the modification is often accompanied with side effects, such as missing non-sensitive rules or invalid spurious rules. The challenge is in protecting the sensitive rules while minimizing the changes to non-sensitive rules. The transformation process from the original database into a released one to hide sensitive rules is termed as association rule hiding or data sanitization.

An ideal sanitization process hides all sensitive rules and retains all non-sensitive rules from the original database, without introducing any ghost rules. However, in practice, it is difficult to achieve such an ideal goal without introducing any side effects. It strongly depends on the dataset and

Manuscript received October 15, 2014.
Manuscript revised April 23, 2015.
Manuscript publicized July 14, 2015.

[†]The authors are with Shenzhen Graduate School, Harbin Institute of Technology, P.R. China.
[††]The author is also with School of Computer and Information Science, Southwest University, P.R. China.
[†††]The author is with School of IT and Mathematical Sciences, University of South Australia, Australia.
[††††]The author is with School of Computer Science, Engineering and Mathematics, Flinders University, Australia.
a) E-mail: chengp.mail@gmail.com

DOI: 10.1587/transinf.2014EDP7345

sensitive rules specified by a user. Thus, the aim of a hiding approach is to conceal the sensitive rules with the minimal side effects and data distortion.

Some methods have been proposed to solve privacy issues in association rule mining [3], [4]. Atallah et al. [5] proposed a protection algorithm for data sanitization to avoid the inference of association rules, and proved that the underlying problems are NP-hard. Dasseni et al. [2], [6] suggested hiding sensitive rules by removing or inserting items to reduce their supports or confidences. The transaction length is used as a basis to select the candidates for modification. The concept behind this strategy is that modifying a shorter transaction may cause fewer side effects than modifying a longer one.

The information on the itemsets or rules contained in a transaction can be used to select candidates for modification. In this direction, Amiri et al. [7] proposed heuristic algorithms to hide itemsets by removing transactions or items, in terms of the sensitive and non-sensitive itemsets supported by each transaction. Verykios et al. [8] improved the existing work by introducing both a distortion-based hiding algorithm and a blocking-based hiding algorithm. The distortion algorithm assigns a weight to each rule according to the difference of its confidence level to the minimum confidence threshold. A priority value is computed for each transaction according to the accumulated weight values of the rules it supports. The transactions with lower priority values are firstly modified. The blocking algorithm follows a similar logic but it sanitizes the database with unknown values. The support and confidence of an association rule can be fuzzified by this way. The blocking based technique is considered less destructive to the original data. It is also discussed in [9]–[11]. Borrowing the idea of TF-IDF (Term Frequency-Inverse Document Frequency) in text mining, Hong et al. [12] devised a greedy-based sanitization approach called SIF-IDF for itemset hiding. Each transaction is evaluated by the correlation degree with the sensitive itemset. This method only utilizes the information on sensitive itemsets contained in a transaction, the non-sensitive ones are not considered although they are closely related to side effects.

Menon et al. [13] firstly formulated the problem of itemset hiding as a constraint satisfaction problem (CSP) and solved it with an integer program. The number of transactions that have to be modified so as to hide sensitive itemsets is minimized using the integer program. They further improved their methods in [14] which aims to minimize both the number of sanitized transactions and the number of missing non-sensitive itemsets. Sun et al. [15] proposed a border-based approach to hide itemsets, which focused on preserving the border of non-sensitive frequent itemsets rather than considering all during the sanitization process. Based on this concept, Divanis et al. [16], [17] developed an exact solution to the frequent itemset hiding problem.

Some approaches allow for partial sensitive itemsets or rules not to be hidden. Oliveira et al. [18] introduced multiple itemset hiding approaches. One major novelty with their approach is to begin to take into account the impact on hiding legitimate non-sensitive patterns. However, these approaches cannot ensure to hide all sensitive itemsets completely. Wu et al. [19] proposed a method aimed at avoiding all the side effects in the rule-hiding process instead of hiding all sensitive rules.

Though most existing approaches can completely conceal the specified sensitive itemsets or rules, the accompanied side effects are often significant. More efforts are needed to explore new ways to reduce the side effects along with the sanitization. In this paper, we propose a new rule-hiding approach. It aims at reducing side effects generated in the sanitization process, especially the missing non-sensitive rules while ensuring all sensitive rules are concealed. The proposed approach hides sensitive rules by removing items in the identified transactions which support them, so that sensitive rules can escape the mining in the modified database at some predefined thresholds. The concepts of positive border rule and negative border rule are defined to identify the rules which may be easily affected by database modifications to produce side effects. The supporting transactions are evaluated based on their relation with positive border rules and negative border rules. The weakly relevant ones will be selected preferentially for modification. The proposed approach is compared against the algorithm 2.a [2] and WSDA [8]. Our study shows that the proposed approach can achieve satisfactory results. All sensitive rules can be concealed with much fewer side effects and data distortion.

2. Definitions and Problem Description

In this section, we firstly introduce some basic notions about frequent itemsets and association rules mining. Let $I = \{I_1, I_2, \ldots, I_m\}$ be a set of available items. An itemset X is a subset of I. A transaction t is characterized by an ordered pair, denoted as $t = \langle ID, X \rangle$, where ID is a unique transaction identifier number and X is a list of items making up the transaction. A transactional database D is a relation consisting of a set of transactions. For instance, in market basket data, a transactional database is composed of business transactions. Each transaction consists of items purchased in a store.

Absolute support of the itemset X is the number of transactions in D that contain X, which is denoted as $supp(X)$. Likewise, the relative support of X is the fraction (or percentage) of the transactions in database which contain X, denoted as $supp'(X)$. $supp'(X) = supp(X)/|D|$.

An itemset X is called frequent if $supp'(X)$ is at least equal to a minimum relative support threshold (denoted as MST) specified by the user. The goal of frequent itemset mining is to find all itemsets which are frequent with the database D.

The notion of confidence is relevant to association rules. A rule has the form of $X \rightarrow Y$ (or $r_l \rightarrow r_r$). It means that the antecedent X infers to the consequent Y, where both X and Y are itemsets. $X \cap Y = \emptyset$. The confi-

dence of a rule is computed as $supp(X \cup Y)/\, supp(X)$, and denoted as $conf(X \rightarrow Y)$. It indicates a rule's reliability. Like MST, users can also specify a minimum confidence threshold called MCT.

In this work, a rule $X \rightarrow Y$ is considered strong if it satisfies the following conditions:

- $supp'(X \cup Y) \geq MST$ and
- $conf(X \rightarrow Y) \geq MCT$.

Association rule mining usually includes two phases:

- Firstly, frequent itemsets are mined with a given MST.
- Then, strong association rules are generated from the frequent itemsets based on a given MCT.

For the remainder of this paper, when the concept of association rules is used, we refer to strong rules. For a rule r, r_l denotes the left itemset (i.e., the antecedent of r) and r_r denotes the right itemset (i.e., the consequent of r).

The rule-hiding problem can be formulated as follows. Let R be the set of strong rules that can be mined from D with given MST and MCT. Let R_S denotes a set of sensitive rules that need to be hidden, and $R_S \subset R$. R_N is the set of non-sensitive rules. $R_N \cup R_S = R$. The hiding problem is to transform D into a sanitized database D' so that only the rules which belong to R_N can be mined out from D' with the same MST and MCT. Let R' denote the strong rules mined from the sanitized database D'.

A sensitive rule is considered to be concealed if its support drops below MST or its confidence drops below MCT. In other words, the sensitive rule cannot be mined out at the same or higher MST and MCT in the sanitized database as the ones used in the original database.

There are three possible side effects after transforming D into D':

- Some sensitive rules may still be exposed in the modified database D' with the same thresholds. This subset of sensitive rules is denoted as S-N-H (Sensitive rules Not Hidden). S-N-H = $\{r \in R_S | r \in R'\}$.
- Some non-sensitive strong rules in D may also cease to be strong when their supports drop below MST or their confidences drop below MCT in D'. These rules have been falsely hidden. We denote them as N-S-L (Non-Sensitive rules Lost). N-S-L = $\{r \in R_N | r \notin R'\}$.
- Some rules which are not strong originally in D but become strong in D'. These newly generated spurious/ghost rules are denoted as S-F-G (Spurious rules Falsely Generated). S-F-G = $\{r \in R' | r \notin R\}$.

Table 1 Side effects generated during the hiding process

Before Hiding	After Hiding	Side effects
$supp'(r) \geq MST \wedge conf(r) \geq MCT \wedge r \in R_S$	$supp'(r) \geq MST \vee conf(r) \geq MCT$	S-N-H
$supp'(r) \geq MST \wedge conf(r) \geq MCT \wedge r \in R_N$	$supp'(r) < MST \vee conf(r) < MCT$	N-S-L
$(supp'(r) < MST \vee conf(r) < MCT) \wedge r \notin R$	$supp'(r) \geq MST \wedge conf(r) \geq MCT$	S-F-G

Table 1 summarizes the side effects along with the sanitization process.

3. Proposed Solution

3.1 Hiding Process

The rule hiding process can be divided into two phases. In the first phase, the Apriori algorithm [20], [21] may be used to find the frequent itemsets and the association rules. In order to improve the efficiency, we adopted an improved version of Apriori algorithm – the trie-based Apriori implementation [22]–[24]. The output of the first phase is a set of frequent itemsets and association rules. Then a user needs to select and specify the sensitive rules from the generated association rules. The definition of sensitive rules is based on the user's personal preference, policy enforcement or business benefit conflict. In the second phase, the hiding algorithm is applied to the original database and it identifies some candidate transactions to modify by removing items. The aim of the second phase is to reduce the support or confidence of sensitive rules below the minimum thresholds while ensuring minimal impacts to the integrity of the original data and non-sensitive knowledge contained in it.

3.2 Hiding Strategy

The basic rule hiding strategy is to modify the database by removing some items so that the support or confidence of all sensitive rules drops below a user-specified threshold: MST or MCT. Based on this strategy, two underlying questions need to be answered before the item removal operation.

- Which transactions are identified for modification?
- Which item is selected for removal in an identified transaction?

For the first question, only the transactions which fully support the sensitive rule need to be considered, because the modification of other transactions does not influence the support of the generating itemset or confidence of the sensitive rule. For transactions that only support the antecedent part of a sensitive rule but not support the consequent part, if the removal operation is performed to reduce the support of the antecedent part, it will increase the confidence of the sensitive rule. This is opposite to our goal of hiding the rule. Otherwise, for the transactions that only support the consequent part of the sensitive rule, the removal operation on the consequent part has no any effect on the support and confidence of sensitive rules.

However, randomly selecting transactions which fully support any sensitive rule to modify is an inadequate strategy, despite the fact that sensitive rules can be hidden by this way, as choosing different subsets of supporting transactions may lead to different side effects. Thus we need to find some measure(s) to evaluate the relevance of different supporting transactions in order to select the ones that yield the fewest side effects.

For the second question, if we remove an item in the consequent part of the sensitive rule in an identified supporting transaction, both the support of the generating itemset and the confidence of the sensitive rule will be reduced, and the support of its antecedent remains the same as before. In contrast, if we remove an item corresponding to the antecedent part of the sensitive rule in a supporting transaction, both the union support of generating itemset and the support of the antecedent will be reduced. Although this can reduce the confidence of the sensitive rule, the confidence decreases slowly compared with the former method. Since some sensitive rules can be concealed with a fewer number of database modifications through reducing its confidence below MCT than through reducing its support below MST, we choose the item to remove that holds the highest support in the consequent part. Removing the item with the highest support might bring fewer side effects.

3.3 The Minimum Number of Transactions for Modification to Hide a Rule

In order to minimize the damage to the database, we can calculate in advance the minimum number of transactions that have to be modified in order to hide a given sensitive rule. A sensitive rule is concealed by reducing its support below MST or its confidence below MCT. Thus, the following properties can be derived.

Property 1. Let $\Sigma_{X\cup Y}$ be the set of all transactions that support sensitive rule $X \rightarrow Y$. In order to decrease the confidence of the rule below MCT, the minimal number of transactions requiring sanitization in $\Sigma_{X\cup Y}$ can be determined by:

$$NUM_1 = \lceil (supp(X\cup Y) - supp(X) * MCT) \rceil + 1 \quad (1)$$

Proof. Removing one item from a transaction in $\Sigma_{X\cup Y}$ which corresponds to the consequent part will decrease the support of the rule $X \rightarrow Y$ by 1. Assuming Θ is the minimum number of transactions that need to be modified in $\Sigma_{X\cup Y}$ to reduce the confidence of the rule below MCT, then we have:

$$\begin{aligned} & (supp(X\cup Y) - \Theta)/\, supp(X) \; < MCT \\ \rightarrow \;& supp(X\cup Y) - supp(X) * MCT < \Theta \end{aligned}$$

Because Θ is an integer and Θ is the minimum number which is greater than $supp(X \cup Y) - supp(X) * MCT$, we can get:

$$\begin{aligned} & \Theta > supp(X\cup Y) - supp(X) * MCT \\ \rightarrow \;& \Theta = \lceil (supp(X\cup Y) - supp(X) * MCT) \rceil + 1 \end{aligned}$$

□

Property 2. Let $\Sigma_{X\cup Y}$ be the set of all transactions which support sensitive rule $X \rightarrow Y$. In order to decrease the support of the generating itemset of $X \rightarrow Y$ below MST, the minimal number of transactions which have to be sanitized in $\Sigma_{X\cup Y}$ is:

$$NUM_2 = \lceil supp(X \cup Y) - MST * |D| \rceil + 1 \quad (2)$$

Proof. Removing one item in a transaction belonging to $\Sigma_{X\cup Y}$ will decrease the support of the rule $X \rightarrow Y$ by 1. Assume Θ is the minimum number of transactions that have to be sanitized in $\Sigma_{X\cup Y}$ in order to reduce the support of the rule below MST. Then we have:

$$\begin{aligned} & (supp(X \cup Y) - \Theta)/|D| < MST \\ \rightarrow\; & supp(X \cup Y) - |D| * MST < \Theta \end{aligned}$$

Because Θ is an integer and Θ is the minimum number which is greater than $supp(X\cup Y) - MST * |D|$, we can get:

$$\begin{aligned} & \Theta > supp(X \cup Y) - |D| * MCT \\ \rightarrow \;& \Theta = \lceil supp(X \cup Y) - MST * |D| \rceil + 1 \end{aligned}$$

□

Based on Property 1 and Property 2, we can infer the minimum number of transactions to be sanitized to hide the sensitive rule $X \rightarrow Y$ is:

$$\begin{aligned} & N_{Mod} = Min\{NUM_1, NUM_2\} = \\ & Min\{\lceil supp(X\cup Y) - supp(X) * MCT \rceil + 1, \\ & \quad \lceil supp(X\cup Y) - MST * |D| \rceil + 1\} \end{aligned} \quad (3)$$

3.4 Border Rules

Since the aim is to minimize side effects – missing non-sensitive rules and spurious/ghost rules, a better strategy is to investigate how these side effects can be generated. Missing rules refer to originally strong rules which become hidden because their supports drop below MST or their confidences drop below MCT in the sanitization process. Actually, only a part of strong rules is likely to fall below the thresholds. Since we remove the item in the consequent with the highest support, only the rules which contain the removed item are possibly affected by the removal operation. Assuming that the deleted item is i_{del} and the strong rules affected is represented as R_{common}, we can narrow down the affected strong rules, which could be hidden by mistake, as shown in Eq. (4).

$$R_{common}(i_{del}) = \{r | r \in R_N \wedge i_{del} \in r\} \quad (4)$$

Furthermore, even if a strong rule contains the removed item in the antecedent or in the consequent and is always affected by the removal operation, it still may not be concealed. The reason is that some affected rules's supports or confidences are so high that a limited number of removal operations cannot reduce their supports below MST or reduce their confidences below MCT. So these rules need not to be considered when evaluating a transaction, although their supports or confidences can be reduced in the sanitization process. The minimum number of removal operations to hide a sensitive rule, denoted as N_{mod}, has been derived in Eq. (3). Assume that r is a non-sensitive rule which contains the removed item i_{del} and r_l denotes the antecedent part of the rule r. r is likely to be concealed mistakenly if at least

one of the following conditions is satisfied. The proof is similar to the Eq. (1) and Eq. (2).

(i) $supp(r) - |D| * MST < N_{mod}$

(ii) $supp(r) - supp(r_l) * MCT < N_{mod}$

Thus, we only need to consider the affected rules as indicated in Eq. (5). We call them as positive border rules set ($PBRS$).

$$\begin{aligned} PBRS(i_{del}) = &\{r | r \in R_N \wedge i_{del} \in r \wedge \\ &(supp(r) - |D| * MST < N_{mod} \\ &\vee supp(r) - supp(r_l) * MCT < N_{mod})\} \end{aligned} \tag{5}$$

Positive Border Rule. If a rule belongs to $PBRS$ indicated in Eq. (5), we consider it as a positive border rule. A positive border rule can be easily concealed by mistake in the sanitization process. A missing non-sensitive rule (not strong in D') must be originally a positive border rule in D. It is impossible to reduce the support or confidence of a strong rule below MST or MCT if it is not in $PBRS$.

The removal operation may reduce the support of a rule's antecedent. If the union support of the rule remains unchanged, its confidence will rise. Thus it is possible a rule which is originally not strong become a strong rule after sanitization. This may take place when an item corresponding to the antecedent is removed in the transactions which support the antecedent but not support the consequent.

For instance, the rule $A \rightarrow B$ is not strong if $supp(A \cup B) \geq MST$ and $conf(A \rightarrow B) < MCT$. If a modified transaction only contains the itemset A but not contains the itemset B, the removal operation on A will decrease $supp(A)$, and $conf(A \rightarrow B)$ will increase. When $conf(A \rightarrow B)$ rises above MCT, it becomes a strong rule. So we need to consider the rules with such a feature, as indicated in Eq. (6). We call them as pre-strong rules or negative rules.

$$\begin{aligned} R_{pre-strong}(i_{del}) = &\{r | r \notin R \wedge i_{del} \in r_l \wedge \\ &supp'(r) \geq MST \wedge conf(r) < MCT\} \end{aligned} \tag{6}$$

Similarly, we can further narrow down the size of the pre-strong rules set. Even if a pre-strong rule contains the removed item in the antecedent and is always affected by the removal operation, it still may not become a strong rule. The reason is that the antecedent's support of some affected pre-strong rules are so high that a limited number of removal operations cannot increase the rule's confidences above MCT. So we only need to consider the affected pre-strong rules as indicated in Eq. (7).

$$\begin{aligned} NBRS(i_{del}) = &\{r | r \notin R \wedge i_{del} \in r_l \wedge \\ &supp'(r) \geq MST \wedge conf(r) < MCT \wedge \\ &supp(r_l) - supp(r)/MCT < N_{mod}\} \end{aligned} \tag{7}$$

Negative Border Rule. If a rule r belongs to $NBRS$ indicated in Eq. (7), we consider it as a negative border rule. A negative border rule can easily become a spurious rule. A newly generated spurious rule in D' must originally be a negative border rule in D.

The transactions can be evaluated according to positive border rules and negative border rules they support. The transactions that contain less ones are modified first. For each positive border rule in $PBRS$, we assign a weight to it, as showed in Eq. (8). Note that r_l and r_r denote the left part(i.e. the antecedent) and the right part (i.e., the consequent) of the rule r respectively.

$$\begin{aligned} &\text{if } i_{del} \in r_r, \text{then} Weight_1(r) = \\ &\quad Min\{1.0/(supp(r) - |D| * MST), \\ &\qquad 1.0/(supp(r) - supp(r_l) * MCT)\} \\ &\text{if } i_{del} \in r_l, \text{then } Weight_1(r) = \\ &\quad Min\{1.0/(supp(r) - |D| * MST), \\ &\qquad 1.0/(supp(r) - supp(r_l) * MCT)/(1 - MCT)\} \end{aligned} \tag{8}$$

Similarly, for each negative border rule in $NBRS$, the weight is calculated as Eq. (9).

$$Weight_2(r) = 1.0/(supp(r_l) - supp(r)/MCT) \tag{9}$$

The closer a positive border rule's support is above MST or the closer its confidence is above MCT, the more likely this rule is excluded from the strong rules set after the sanitization. So this rule is more *dangerous* and gets a higher weight value. In a similar manner, the closer a negative border rule's confidence is below MCT, the more likely it become a strong rule if some items corresponding to its antecedent are removed in its partially supporting transactions. So this negative border rule gets a higher weight value.

For a transaction t, we can calculate its relevance values according to Eq. (10). $Relevance_1(t)$ and $Relevance_2(t)$) can be calculated by summarizing the weight values of positive border rules supported by t and by summarizing the weight values of negative border rules partially supported by t respectively.

$$\begin{aligned} Relevance_1(t) &= \sum_{r \in PBRS \wedge r_l \subseteq t \wedge r_r \subseteq t} Weight_1(r) \\ Relevance_2(t) &= \sum_{r \in NBRS \wedge r_l \subseteq t \wedge r_r \not\subseteq t} Weight_2(r) \end{aligned} \tag{10}$$

3.5 BRDA Algorithm

Based on the discussion above, we devised the hiding algorithm as indicated in Algorithm 1. We named it as BRDA (Border Rule-based Distortion Algorithm).

A smaller MCT level is adopted when applying Apriori, in order to discover both strong rules and pre-strong rules (The concept of pre-strong rules is defined in Eq. (6). As showed in Algorithm 1, to hide the i^{th} sensitive rule r_i,

Algorithm 1 BRDA

Input: The source database D, MST and MCT
Output: The sanitized database D'

$R := Apriori(D)$
// Use Apriori to discover strong association rules
// and pre-strong rules (defined in Eq. (6)).
Specify the sensitive rules R_S from R. $R = R_S \cup R_N$.
for the i^{th} sensitive rule $r_i(X \rightarrow Y)$ in R_S **do**
  $i_{del} := Choose\_item(r_i)$
  // Choose the item to be removed, which corresponds to
  // the consequent of the rule r_i with the highest support.
  $N_{mod} := Min\{\lceil supp(X \cup Y) - supp(X) * MCT \rceil + 1,$
    $\lceil supp(X \cup Y) - MST * |D| \rceil + 1\}$
  // Calculate the minimum number of transactions that
  // need modifications to hide r_i, according to Eq. (3)
  $PBRS := Find\_PBRS(R, i_{del}, N_{mod})$
  $NBRS := Find\_NBRS(R, i_{del}, N_{mod})$
  // Find out the positive border rules set PBRS and
  // the negative border rules set NBRS related to r_i,
  // according to Eq. (5) and Eq. (7).
  Calculate each border rule's weight according to Eq. (8) and Eq. (9).
  $\Sigma_i := \{t \in D | t$ fully supports $r_i\}$
  // Filter out transactions which fully support r_i, denoted as Σ_i.
  Calculate_relevance(Σ_i , $PBRS$, $NBRS$)
  // Calculate the relevance value for each supporting
  // transaction in Σ_i according to Eq. (10).
  Sort(Σ_i)
  // Sort the transactions in Σ_i by relevance in ascending order,
  // firstly by $Relevance_1$ and secondly by $Relevance_2$.
  for j :=1 to N_{mod} **do**
    $t := \Sigma_i[1]$
    // Choose the transaction in Σ_i with the lowest relevance.
    Remove_item(t, i_{del})
    // Remove the item i_{del} from the transaction t.
    Update(t, i_{del}, R_{common})
    // Update supports and confidences of affected rules.
    Remove t from Σ_i
  end for
end for

the algorithm firstly finds positive and negative border rules related to r_i (the details of finding PBRS are given in Algorithm 2, and finding NBRS is similar), and assigns a weight to each of them. Then, the relevance value for each supporting transaction in Σ_i is calculated according to how weak it supports border rules (as detailed in Algorithm 3). The supporting transactions in Σ_i are sorted by relevance in ascending order, and the front N_{mod} transactions are selected for modification.

Algorithm 2 shows how to find the positive border rules set for a sensitive rule. The aim of this algorithm is to filter out the rules that are likely to become missing after the sanitization process. It traverses the rules in R one by one and judges whether the current one meets the following three conditions: (1) It is a strong rule; (2) It contains the removed item i_{del}; (3) It is possible that its support or its confidence can be reduced below the minimum threshold through N_{mod} times of removal operations. The number of removal operations, i.e., the input parameter N_{mod}, is related to the sensitive rule to be hidden and can be determined by Eq. (3). If the current rule satisfies these conditions, it will be added into the set $PBRS$. R_{common} is the set of rules that have common items with $\{i_{del}\}$. $PBRS \subseteq R_{common}$. R_{common} collects all rules that may be affected due to the removal of the item i_{del}. In contrast, $PBRS$ only contains the rules that may be hidden as the item i_{del} is removed.

The process of finding the negative border rules has a similar logic flow, but the traversing scope is limited to the pre-strong rules (as defined by Eq. (6)). The selection criteria is based on Eq. (7).

Algorithm 3 shows how to assign a relevance value to each supporting transaction. Σ_i denotes the supporting transactions set for the i^{th} sensitive rule. The relevance value of a transaction is calculated according to the weight of positive border rules it contains and the weight of negative border rules it partially contains.

Algorithm 2 Find_PBRS

Input: R, i_{del}, N_{mod}
Output: $PBRS, R_{common}$

$R_{common} := \emptyset, PBRS := \emptyset$
for each $r(X \rightarrow Y) \in R$ **do**
  if $i_{del} \in r$ **then**
    $R_{common} := R_{common} \cup \{r\}$
  end if
  if $i_{del} \in r \wedge supp'(r) \geq MST \wedge conf(r) \geq MCT$ **then**
    $Above\_MST := supp(r) - |D| * MST$
    if $i_{del} \in Y$ **then**
      $Above\_MCT := supp(r) - supp(X) * MCT$
      // The consequent of r contains the item i_{del} .
    else
      $Above\_MCT := (supp(r) - supp(X) * MCT)/(1 - MCT)$
      // The antecedent of r (i.e., X) contains the item i_{del}.
    end if
    if $Above\_MST < N_{mod}$ or $Above\_MCT < N_{mod}$ **then**
      $PBRS := PBRS \cup \{r\}$
    end if
  end if
end for

Algorithm 3 Calculate_relevance

Input: Σ_i , $PBRS$, $NBRS$
Output: The relevance of each transactions in Σ_i

for each $t \in \Sigma_i$ **do**
  for each $r \in PBRS$ **do**
    if t fully support r **then**
      $Relevance_1(t) := Relevance_1(t) + Weight_1(r)$
    end if
  end for
  for each $r \in NBRS$ **do**
    if t only support the antecedent of r **then**
      $Relevance_2(t) := Relevance_2(t) + Weight_2(r)$
    end if
  end for
end for

4. Performance Evaluation

The proposed algorithm was implemented in C++, running on a server with four Intel Xeon X5650 processors at

2.67GHz and 4GB of main memory. The experiment results were measured according to the three side effects, the distortion degree and CPU time described as follows. Lower values over these criteria reflect better sanitization results.

- α = |S-N-H| : the number of sensitive rules exposed in the sanitized database.
- β = |N-S-L| : the number of missing non-sensitive rules.
- γ = |S-F-G| : the number of spurious/ghost rules newly generated.
- # of modified items: the number of items modified during the sanitization process. This metric reflects the degree of data loss/distortion.
- CPU time: the efficiency is measured by the CPU time in seconds. It includes the time of sanitizing the database, and the time of calculating side effects.

4.1 Real Datasets

We examined the proposed approach using five well-known real datasets available on the FIMI repository[†]: mushroom, bms-webview-1, bms-webview-2, chess and retail. The mushroom and chess datasets, prepared by Roberto Bayardo [25], are from the Irvine machine learning database repository. The bms-1 and bms-2 datasets were used for the KDD Cup of 2000 [26] and contain click stream data from the website of a legwear and legcare retailer. The retail dataset contains the retail market basket data from an anonymous Belgian retail store. As shown in Table 2, these datasets exhibit varying characteristics with respect to the number of transactions and items that they contain, as well as with respect to the average transaction length.

The thresholds MST and MCT were set to ensure that sufficient frequent itemsets and association rules can be generated. Generally, a higher MST level is used for a denser dataset (such as mushroom and chess), and a lower MST level is used for a sparser dataset (such as bms-1, bms-2 and retail). The density of a database can be measured as the average transaction length divided by the number of different items. In a denser dataset, most itemsets hold very high supports. As indicated in [13], very dense data sets are not the representatives of transactional data in reality. Sparse data sets are usually observed more commonly in the real-world scenario. However, we may use denser datasets to evaluate the performance of a sanitization approach.

The proposed method, BRDA, is compared against WSDA [8] and Algo2.a [2]. The results are indicated in Table 3. For each dataset, 5, 10 and 20 sensitive rules were randomly selected from the strong rules set to perform the hiding task. To conceal more sensitive rules means that more transactions need to be sanitized so as to suppress the support or confidence of sensitive rules below the minimum thresholds. Thus more non-sensitive rules may be affected.

Table 2 The characteristics of datasets

Dataset	Count of Tran.	Count of Items	Avg. Tran. Len.	MST	MCT
Mushroom	8,124	119	23	0.05	0.5
Bms-1	59,602	497	2.5	0.001	0.2
Bms-2	77,512	3,340	5.0	0.002	0.2
Chess	3,197	75	40.2	0.5	0.5
Retail	88,162	16,469	10.3	0.001	0.5

[†]http://fimi.cs.helsinki.fi/

Table 3 shows that the three approaches can hide all sensitive rules completely. On the side effect of non-sensitive lost rules (β), the proposed approach outperforms the method WSDA and Algo2.a by a large margin in all comparative cases. This is expected since the proposed method evaluates the candidates in terms of border rules and chooses weakly correlated transactions for modification. On the side effect of spurious rules (γ), it also achieves competing or better results for the mushroom, chess and retail datasets. For the bms-1 and bms-2 dataset, the WSDA method gets a slightly better values on the side effect of spurious rules at cost of much greater side effect of missing non-sensitive rules. In particular, for the three test cases on retail, the proposed method achieves perfect results with no side effects. The retail dataset is from a real retailing store. It is a good representative of sparser dataset in the real world application scenario.

For the number of modified items, it is obvious in most comparative cases that the proposed method can hide all sensitive rules with fewer database modifications than WSDA and Algo2.a, resulting in a lower distortion degree. When the number of sensitive rules increases, the number of modified items is increased accordingly.

As illustrated in Table 3, the CPU time increases with the increasing number of sensitive rules. In contrast, Algo2.a is the most efficient, followed by BRDA. However, the time spent for BRDA is worthwhile since the quality of solutions is superior for most test cases (i.e., much fewer side effects and database modifications).

The comparative experimental results demonstrate that the proposed approach, BRDA, can effectively perform the hiding task with much fewer side effects and database modifications. The strategy of weighting each supporting transaction in terms of related border rules guides the algorithm to find proper transactions to sanitize with fewer side effects. The pre-computation of the minimum number of transactions for modification to conceal each sensitive rule make the algorithm can perform the sanitization with fewer database modifications. In addition, in step 8 of BRDA, the mechanism of dynamically updating supports and confidence of the affected rules in the sanitization process may reduce the support or confidence of the follow-up sensitive rules.

Algo2.a chooses transactions to sanitize based on their lengths. It assumes that the modification on a longer transaction may bring more side effects compared with modification on a shorter transaction, since a longer transaction may contain more rules on average. The information on sensitive rules or non-sensitive rules is not used as the guiding criteria to select candidate transactions. However, in some datasets

Table 3 Comparative results with increasing numbers of sensitive rules

Dataset	$\|R_S\|$	BRDA			WSDA			Algo2.a		
		(α, β, γ)	# of modified	time(sec)	(α, β, γ)	# of modified	time(sec)	(α, β, γ)	# of modified	time(sec)
Mushroom	5	(0, 11, 1)	1061	78.17	(0, 13, 1)	1547	184.23	(0, 54, 2)	1469	45.95
	10	(0, 21, 1)	2649	214.94	(0, 28, 4)	3390	424.52	(0, 123, 2)	3179	75.58
	20	(0, 72, 5)	7765	670.34	(0, 87, 6)	8506	1119.83	(0, 216, 5)	11905	435.13
Bms-1	5	(0, 3, 0)	188	12.19	(0, 7, 0)	344	12.49	(0, 3, 0)	188	8.80
	10	(0, 7, 3)	936	38.09	(0, 12, 2)	1095	38.94	(0, 19, 18)	1057	31.36
	20	(0, 17, 4)	3389	195.30	(0, 23, 3)	3552	201.05	(0, 53, 33)	4104	200.08
Bms-2	5	(0, 2, 0)	277	12.36	(0, 9, 0)	474	18.19	(0, 9, 0)	277	10.41
	10	(0, 14, 1)	1642	69.11	(0, 26, 0)	1908	92.70	(0, 48, 3)	1909	50.06
	20	(0, 19, 0)	2671	105.24	(0, 36, 0)	3084	139.67	(0, 73, 4)	3148	75.45
Chess	5	(0, 17, 0)	1514	105.05	(0, 131, 0)	4515	676.61	(0, 121, 0)	1514	37.47
	10	(0, 24, 0)	2305	154.22	(0, 256, 0)	8171	1187.69	(0, 178, 0)	2305	53.89
	20	(0, 268, 0)	8317	584.55	(0, 610, 0)	15943	2055.31	(0, 424, 0)	9041	264.84
Retail	5	(0, 0, 0)	85	40.53	(0, 3, 0)	170	126.64	(0, 6, 0)	117	19.84
	10	(0, 0, 0)	117	83.64	(0, 5, 0)	281	259.69	(0, 7, 0)	149	35.34
	20	(0, 0, 0)	202	178.31	(0, 13, 0)	526	503.31	(0, 9, 0)	234	66.64

most transactions possess the same length. The mushroom dataset is such a case. Although the WSDA method gives each transaction a priority value according to how weak it supports the related strong rules (the related strong rules are those which have at least one common items with the consequent part of the sensitive rule), it does not take the support range or confidence range of related rules into account. It is possible that the supports and confidences for some related rules are so high that they cannot be suppressed below thresholds in sanitization. In addition, WSDA only utilize the confidence information to hide sensitive rules without consideration on the supports of sensitive rules. This may also limit its performance.

According to our observation, the density of a dataset seems to play a significant role. With the same number of sensitive rules to be hidden, more side effects are generated and more items need to be removed on the denser datasets (mushroom and chess) than on the sparser datasets (bms-1, bms-2 and retail). In addition, the CPU time is also more for sanitizing denser datasets. The density of a dataset can be measured by the average transaction length divided by the number of items. In a denser dataset, a transaction may contain more association rules on average, and modification on it could affect more rules and bring more potential side effects. In addition, the supports of sensitive rules in a denser dataset are often higher than a sparser one and the number of transactions that need to be sanitized tends to be more. Therefore, on denser datasets, the side effects, the distortion degree and the solution time are increased significantly compared with sparser ones.

4.2 Synthetic Datasets

We conducted extended tests to evaluate the scalability of these algorithms. They are measured in terms of CPU time, side effects and distortion degree by increasing the database size. The IBM Synthetic Data Generator was utilized to generate artificial data sets. The number of items is 50 and the average transaction length is 5. Three algorithms were tested on the databases with increasing sizes: 20k, 40k, 60k, 80k and 100k. 10 rules are randomly selected as sensitive ones. These sensitive rules were used for all tests with increasing database sizes.

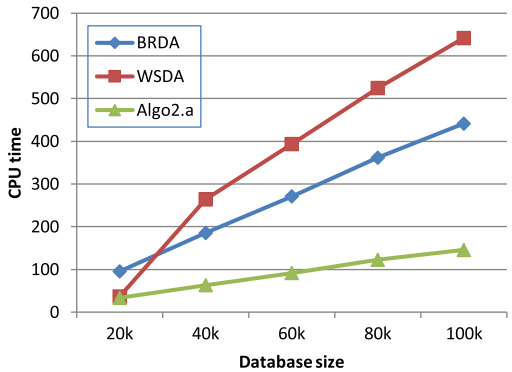

Fig. 1 CPU times of three algorithms when increasing database sizes

The experiments results are shown in Fig. 1, Fig. 2 and Fig. 3. The side effects are measured as the sum of |S-N-H|, |N-S-L| and |S-F-G|. Actually, the sensitive rules can be completely concealed on all test cases, i.e. |S-N-H| = 0. We observe that the CPU time grows linearly with increasing database sizes for three algorithms. Algo2.a proves to the most efficient again. However, it has the worst performance with respect to side effects and distortion degree. BRDA performs the hiding task on five test cases with no side effects and the fewest distortion degrees. The side effects for three methods have not risen with increasing database sizes, although their distortion degrees increase linearly.

4.3 Discussions on How to Hide Sensitive Rules

In most existing works on association rule hiding, a sensitive rule is assumed to be hidden if its support level or confidence level drops below a pre-defined threshold (i.e., MST or MCT). However, the measure of MST or MCT is defined by a data miner using the domain knowledge. An attacker may easily find sensitive rules by choosing smaller

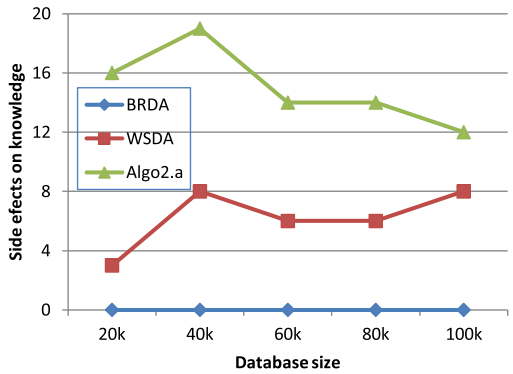

Fig. 2 Side effects of three algorithms when increasing database sizes

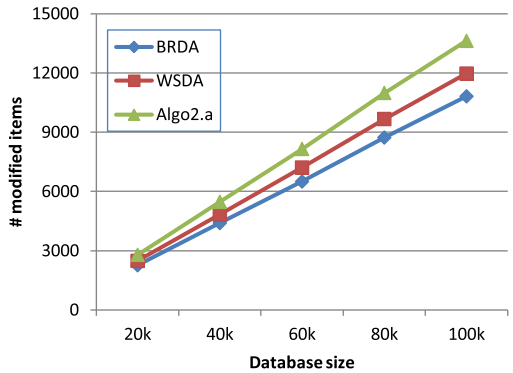

Fig. 3 The number of modified items of three algorithms when increasing database sizes

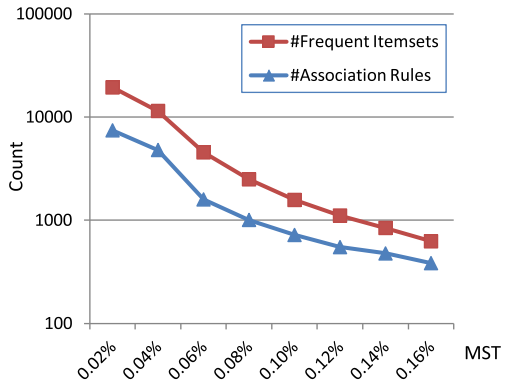

Fig. 4 The number of frequent itemsets and association rules at different MST levels on Bms-1. MCT is fixed at 20%

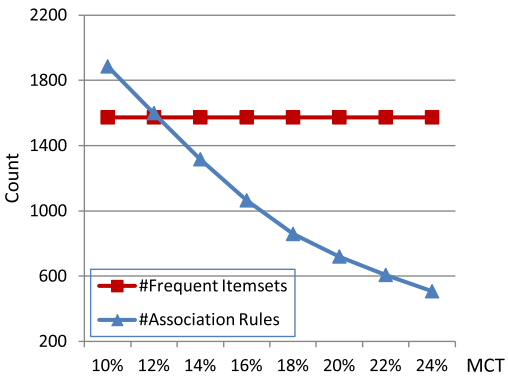

Fig. 5 The number of frequent itemsets and association rules at different MCT levels on Bms-1. MST is fixed at 0.1%

MST and MCT levels than the ones that the data owner expects. Thus, a question arises how sensitive rules can still be protected under such a situation.

To cope with this problem, the data owner may choose relatively low MST and MCT values in the sanitization so that a large number of trivial rules can be generated. In this way, even if an attacker attempts to expose sensitive rules by using smaller MST and MCT values, the sensitive rules have been hidden in such many trivial rules. It is not easy to infer which of the rules are sensitive and which are not.

To demonstrate the effectiveness of this strategy, we take bms-1 as an example to investigate how the distributions of the minimum supports and the minimum confidences may affect the count of frequent itemsets and the count of association rules. Figure 4 shows that the number of frequent itemsets and the number of association rules increase rapidly when the MST level decreases. Note that the vertical axis in Fig. 4 is on a logarithmic scale. The exponentially increasing number of frequent itemsets suggests we may get many trivial rules by using smaller MST values on bms-1. Similarly, we may control the number of rules by varying the MCT levels. Figure 5 shows that the number of rules increases as MCT decreases on bms-1. From Fig. 4 and Fig. 5, we may notice that the number of rules increases more rapidly by reducing MST than by reducing MCT. This is because many new frequent itemsets can be generated by using smaller MST values. The newly generated frequent itemsets may produce a large number of additional rules.

A user can specify the *Safety Margin* (SM) threshold to increase the privacy level. The safety margin is a value below MST or MCT. By using the concept of safety margin, the support of a sensitive rule is reduced below $MST - SM_1$ or the confidence of it is reduced below $MCT - SM_2$. SM_1 and SM_2 are safety margins used for the minimum support and the minimum confidence respectively. The use of safety margin allows the support or confidence levels of sensitive rules become much lower after sanitization than the minimum threshold. The attacking behavior through using smaller MST and MCT to uncover sensitive rules will lead to more trivial rules to be generated. However, the trade-off between privacy and side effects needs to be balanced. Larger safety margin values will create more side effects since more supporting transactions have to be sanitized.

5. Conclusions

Privacy preserving in association rule mining is an important research topic in database security. This paper has proposed an efficient method to solve the association rule hiding problem for data sharing. The concepts of positive border rules and negative border rules are introduced, which may provide an estimation for the possibility of a rule to develop

into a missing rule or a ghost rule, influenced by database modifications. Based on positive and negative border rules contained, each supporting transaction is assigned two relevance values to indicate how weakly it supports them. Then, the transactions with weaker relevance are selected for sanitization firstly. The proposed method can hide all sensitive rules in a single pass and has been shown to be very effective on reducing side effects. The experiment results show that the proposed method can hide sensitive rules with much fewer side effects and data loss compared with WSDA and 2.a. How to sort the supporting transactions in the proposed method in a better way may be studied further. Sometimes, there is a tradeoff between two relevance values. Some transactions hold a lower $relevance_1$ but a higher $relevance_2$, and some transactions are just the opposite. We are also interested in the blocking based rule hiding methods, which need not to distort the database but replacing some items with unknown values. This way can ensure a user to construct the model with the remaining data safely and deserve more attentions in the future.

References

[1] R. Agrawal and R. Srikant, "Privacy-preserving data mining," ACM SIGMOD Record, vol.29, no.2, pp.439–450, 2000.
[2] V.S. Verykios, A.K. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, "Association rule hiding," IEEE Trans. Knowl. Data Eng., vol.16, no.4, pp.434–447, 2004.
[3] A. Gkoulalas-Divanis and V.S. Verykios, Association Rule Hiding for Data Mining, Advances in Database Systems, vol.41, Kluwer, 2010.
[4] V.S. Verykios, "Association rule hiding methods," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol.3, no.1, pp.28–36, 2013.
[5] M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. Verykios, "Disclosure limitation of sensitive rules," Proc. Workshop on Knowledge and Data Engineering Exchange(KDEX), pp.45–52, IEEE, 1999.
[6] E. Dasseni, V.S. Verykios, A.K. Elmagarmid, and E. Bertino, "Hiding association rules by using confidence and support," Information Hiding, pp.369–383, Springer, 2001.
[7] A. Amiri, "Dare to share: Protecting sensitive knowledge with data sanitization," Decision Support Systems, vol.43, no.1, pp.181–191, 2007.
[8] V.S. Verykios, E.D. Pontikakis, Y. Theodoridis, and L. Chang, "Efficient algorithms for distortion and blocking techniques in association rule hiding," Distributed and Parallel Databases, vol.22, no.1, pp.85–104, 2007.
[9] E. Bertino, I.N. Fovino, and L.P. Provenza, "A framework for evaluating privacy preserving data mining algorithms," Data Mining and Knowledge Discovery, vol.11, no.2, pp.121–154, 2005.
[10] E.D. Pontikakis, A.A. Tsitsonis, and V.S. Verykios, "An experimental study of distortion-based techniques for association rule hiding," in Research Directions in Data and Applications Security XVIII, pp.325–339, Springer, 2004.
[11] Y. Saygin, V.S. Verykios, and C. Clifton, "Using unknowns to prevent discovery of association rules," ACM SIGMOD Record, vol.30, no.4, pp.45–54, 2001.
[12] T.P. Hong, C.W. Lin, K.T. Yang, and S.L. Wang, "Using TF-IDF to hide sensitive itemsets," Applied Intelligence, vol.38, no.4, pp.502–510, 2013.
[13] S. Menon, S. Sarkar, and S. Mukherjee, "Maximizing accuracy of shared databases when concealing sensitive patterns," Information Systems Research, vol.16, no.3, pp.256–270, 2005.
[14] S. Menon and S. Sarkar, "Minimizing information loss and preserving privacy," Management Science, vol.53, no.1, pp.101–116, 2007.
[15] X. Sun and P.S. Yu, "A border-based approach for hiding sensitive frequent itemsets," Proc. Fifth IEEE International Conference on Data Mining, pp.426–433, 2005.
[16] A. Gkoulalas-Divanis and V.S. Verykios, "Hiding sensitive knowledge without side effects," Knowledge and Information Systems, vol.20, no.3, pp.263–299, 2009.
[17] A. Gkoulalas-Divanis and V.S. Verykios, "An integer programming approach for frequent itemset hiding," Proc. 15th ACM international conference on Information and knowledge management, pp.748–757, ACM, 2006.
[18] S.R. Oliveira and O.R. Zaiane, "Privacy preserving frequent itemset mining," Proc. IEEE international conference on Privacy, security and data mining, pp.43–54, 2002.
[19] Y.H. Wu, C.M. Chiang, and A.L. Chen, "Hiding sensitive association rules with limited side effects," IEEE Trans. Knowl. Data Eng., vol.19, no.1, pp.29–42, 2007.
[20] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," Proc. 20th international conference on very large data bases, VLDB, pp.487–499, 1994.
[21] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," ACM SIGMOD Record, vol.22, no.2, pp.207–216, 1993.
[22] F. Bodon, "A survey on frequent itemset mining," Budapest University of Technology and Economics, Tech. Rep, 2006.
[23] F. Bodon, "A trie-based apriori implementation for mining frequent item sequences," Proc. 1st international workshop on open source data mining: frequent pattern mining implementations, pp.56–65, ACM, 2005.
[24] F. Bodon, "A fast apriori implementation," Proc. IEEE ICDM workshop on frequent itemset mining implementations (FIMI), vol.90, 2003.
[25] R.J. Bayardo Jr, "Efficiently mining long patterns from databases," ACM SIGMOD Record, vol.27, no.2, pp.85–93, 1998.
[26] R. Kohavi, C.E. Brodley, B. Frasca, L. Mason, and Z. Zheng, "Kdd-cup 2000 organizers' report: Peeling the onion," ACM SIGKDD Explorations Newsletter, vol.2, no.2, pp.86–93, 2000.

Peng Cheng received the M.S. degree in computer science from the Beijing University of Technology, Beijing, China. Then he worked as a lecturer at Southwest University at China. He is currently working towards the Ph.D. degree in computer science at Shenzhen Graduate School, Harbin Institute of Technology, China. His research interests include privacy preserving data mining, data mining, evolutionary multi-objective optimization, swarm intelligence.

Ivan Lee received BEng, MCom, MEng by Research, and PhD degrees from the University of Sydney, Australia. In the past, he was a development engineer at Cisco Systems, a software engineer at Remotek Corporation, and an assistant professor at Ryerson University. He is currently working at the University of South Australia as a senior lecturer. His research interests include multimedia systems, data mining, and signal processing.

Jeng-Shyang Pan received the B.S. degree in electronic engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan in 1986, the M.S. degree in communication engineering from the National Chiao Tung University, Hsinchu, Taiwan in 1988, and the Ph.D. degree in electrical engineering from the University of Edinburgh, Edinburgh, U.K. in 1996. He is currently a Professor in Harbin Institute of Technology Shenzhen Graduate School. He is also an adjunct professor in Flinders University, Australia. His current research interests include soft computing, information security and signal processing.

Chun-Wei Lin received Ph.D. degree in Dept. of Computer Science and Information Engineering in 2010 from National Cheng Kung University, Tainan, Taiwan. He is currently working as an assistant professor at School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China. He has published more than 100 research papers in referred journals and international conferences. In 2013, he has awarded High-end Professionals under the Peacock Plan of the Shenzhen Government (B-level). His interests include data mining, soft computing, privacy preserving data mining and security, social network and cloud computing.

John F. Roddick received the BSc degree from Imperial College, London, the MSc degree from Deakin University, and the PhD degree from La Trobe University. He is currently Dean of the School of Computer Science, Engineering and Mathematics at Flinders University, Australia. He joined Flinders in April 2000 after 15 years at the Universities of Tasmania and South Australia. This followed 10 years experience in the computing industry as (progressively) a programmer, analyst, project leader and consultant. His technical interests include data mining and knowledge discovery, schema versioning, and enterprise systems. He is a fellow of the Australian Computer Society and the Institution of Engineers, Australia.