Hiding Sensitive Association Rules with Limited Side Effects

Yi-Hung Wu, Chia-Ming Chiang, and Arbee L.P. Chen, *Senior Member*, *IEEE Computer Society*

Abstract—Data mining techniques have been widely used in various applications. However, the misuse of these techniques may lead to the disclosure of sensitive information. Researchers have recently made efforts at hiding sensitive association rules. Nevertheless, undesired side effects, e.g., *nonsensitive rules falsely hidden* and *spurious rules falsely generated*, may be produced in the rule hiding process. In this paper, we present a novel approach that strategically modifies a few transactions in the transaction database to decrease the supports or confidences of sensitive rules without producing the side effects. Since the correlation among rules can make it impossible to achieve this goal, in this paper, we propose heuristic methods for increasing the number of hidden sensitive rules and reducing the number of modified entries. The experimental results show the effectiveness of our approach, i.e., undesired side effects are avoided in the rule hiding process. The results also report that in most cases, all the sensitive rules are hidden without spurious rules falsely generated. Moreover, the good scalability of our approach in terms of database size and the influence of the correlation among rules on rule hiding are observed.

Index Terms—Association rules, data mining, mining methods and algorithms, rule hiding.

1 Introduction

A variety of data mining problems have been studied to help people get an insight into the huge amount of data. One of them is *association rule mining*, which was first introduced by Agrawal et al. [2]. Agrawal and Srikant [4] extend and define the problem as follows: An *itemset* is a set of products (items) and a *transaction* keeps a set of items bought at the same time. The *support* of an itemset I (denoted as Sup_I) in a transaction database is the percentage of transactions that contain I in the entire database. An itemset is *frequent* if its support is not lower than a *minimum support threshold* (denoted as *MST*). For two itemsets X and Y where $X \cap Y = \emptyset$, the *confidence* of an association rule $X \rightarrow Y$ (denoted as $Conf_{X \rightarrow Y}$) is the probability that Y occurs given that X occurs, and is equal to $Sup_{X \cup Y}$ divided by Sup_X. We say that $X \rightarrow Y$ *holds* in the database if $X \cup Y$ is frequent and its confidence is not lower than a *minimum confidence threshold* (denoted as *MCT*). Such a rule is called the *strong association rule* (*strong rule* for short). For the convenience of presentation, in the rest of this paper, we call the association rules that do not hold in the database the *spurious rules*. Association rule mining is to discover all the strong rules in the database. Several methods have been proposed [3], [13], [17], [26], [27]. However, the misuse of them may bring undesired effects to people. An example mentioned in [9] is as follows:

Example 1. Consider a supermarket and two beer suppliers A and B. If the transaction database of the supermarket is released, A (or B) can mine the association rules related to his/her beers and apply the rules to the sales promotion and the goods supply. As a result, a supplier is willing to exchange a lower price of goods for the database with the supermarket. From this aspect, it is good for the supermarket to release the database. However, the conclusion can be opposite if a supplier uses the mining methods in a different way. For instance, if A finds the association rules related to B's beers, saying that most customers who buy diapers also buy B's beers, he/she can run a coupon that gives a 10 percent discount when buying A's beers together with diapers. Gradually, the amount of sales on B's beers is down and B cannot give a low price to the supermarket as before. Finally, A monopolizes the beer market and is unwilling to give a low price to the supermarket as before. From this aspect, releasing the database is bad for the supermarket. Therefore, for the supermarket, an effective way to release the database with sensitive rules hidden is required. This leads to the research of *sensitive rule hiding*.

The work in [10], [24] presents algorithms that insert or delete items to/from transactions for hiding sensitive rules. It makes a strong assumption—all the items in a sensitive rule do not appear in any other sensitive rule. With this assumption, hiding a sensitive rule will not affect any other sensitive rule and, therefore, hiding them one at a time or all together will not make any difference. Thus, their algorithms hide one rule at a time and decrease the supports or confidences one unit at a time. Since this work aims at hiding all sensitive rules, it cannot avoid the undesired side effects, i.e., *lost rules* (nonsensitive strong rules falsely hidden) and *false rules* (spurious rules falsely generated). In this paper, we remove the assumption and allow the user to select sensitive rules from all strong rules. The problem of sensitive rule hiding is described as follows:

- *Y.-H. Wu is with the Department of Information and Computer Engineering, Chung Yuan Christian University, Chungli 32023, Taiwan, ROC. E-mail: yhwu@ice.cycu.edu.tw.*
- *C.-M. Chiang is with VIA Technologies, Inc., Chung-Cheng Road, Hsien-Tien, Taipei 231, Taiwan, ROC. E-mail: g904349@alumni.nthu.edu.*
- *A.L.P. Chen is with the Department of Computer Science, National Chengchi University, Taipei 11605, Taiwan, ROC. E-mail: alpchen@cs.nccu.edu.tw.*

Manuscript received 12 May 2005; revised 2 Dec. 2005; accepted 24 Aug. 2006; published online 20 Nov. 2006.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0197-0505.

Given a transaction database, MST, MCT, and a set of sensitive rules, how can we modify the database such that using the same MST and MCT, the set of strong rules in the modified database satisfies all the constraints: 1) no sensitive rule, 2) no lost rule, and 3) no false rule?

Since violating the last two constraints means the production of side effects, focusing on the first constraint like the work in [24] can be inadequate for certain applications. For example, a supplier cannot enhance his/her goods supply if the corresponding rule is falsely hidden. In medical applications, a misleading rule falsely generated will threaten human lives. Furthermore, in some cases, the correlation among rules can make it impossible to hide sensitive rules without violating any constraint. Therefore, in this paper, we aim at avoiding the side effects (satisfying the last two constraints) in the rule hiding process instead of hiding all sensitive rules.

The common idea to modify the database for rule hiding is as follows: For a sensitive rule $r : X \rightarrow Y$, deleting item $i \in X \cup Y$ from transactions that contain $X \cup Y$ will decrease both $Sup_{X \cup Y}$ and $Conf_r$. Moreover, inserting item $i \in X$ into transactions that contain X but $\{i\}$ and do not contain Y will decrease $Confr$. It can be seen that a modification consists of three elements, including the modification scheme (either deletion or insertion), the item, and the transactions to be modified. The effects of a modification can be different if one of its elements is changed. For example, inserting item $i \in X$ into transactions that contain $X \cup Y$ but $\{i\}$ will increase both $Sup_{X \cup Y}$ and $Conf_r$. A modification is said to be *valid* if it will not produce any side effect in the modified database. Since a number of valid modifications can be applied to hide a sensitive rule, the goal of this paper is to select a proper subset of valid modifications to hide the sensitive rules. Specifically, the modification that affects the least number of transactions and helps to hide the most number of sensitive rules has priority over the others. The problem we are solving is defined as follows:

Given a transaction database, MST, MCT, a set of sensitive rules, and the user-specified constraint 2) no lost rule, 3) no false rule, or both), how can we modify the database such that the user-specified constraint is satisfied while the sensitive rules are hidden as many as possible?

To solve this problem, we propose a novel approach that strategically modifies the database to decrease the supports or confidences of the sensitive rules. Our approach classifies the valid modifications for hiding sensitive rules and represents each class of the modifications by three attributes. The first attribute records the modification scheme. In the case of deletion, the second attribute keeps the set of items that must be contained in the transactions to be modified. Among these items, the third attribute designates one as the item to be deleted. In the case of insertion, the second attribute uses two sets of items to describe the transactions to be modified. One is the set of items that must be contained in the transactions, while the other is the set of items that must not appear in the transactions. From the items in the second set, the third attribute specifies one as the item to be inserted. There can be two classes that are the same in the first two attributes, but different in the third attribute. As a result, for each class, a unique way of modifying the associated set of transactions for hiding some sensitive rules is determined.

Given a class of modifications C, using the idea mentioned above, the rules whose supports or confidences will be affected by C can be found. Our approach first finds all the sensitive rules whose supports or confidences will be decreased by C and computes the minimum number of transactions that should be modified in C to hide all the sensitive rules. After that, our approach finds all the nonsensitive rules whose supports or confidences will be decreased by C and then computes the maximum number of transactions that can be modified in C without incurring any lost rule. Finally, our approach examines the spurious rules whose supports or confidences will be increased by C to estimate the maximum number of transactions that can be modified in C without generating any false rule. With these numbers, the adequate number of transactions to be modified in C can be decided. Based on this number and the number of sensitive rules affected by C, our approach selects the classes of modifications in a one-by-one fashion and produces a sequence of classes as a result.

Before the rule hiding process, the strong rules are mined from the database based on the given *MST* and *MCT*. The user then selects some as sensitive rules and specifies the constraint, such as no lost rule, no false rule, or both. After that, we proceed to the rule hiding process that needs only two database scans. First, the transactions are retrieved from the database and stored as a compact structure in main memory, which also keeps the rule information. After the classes of modifications are produced in sequence as described above, we apply them to the database. The experimental results show that the undesired side effects are avoided by using our approach. The results also report that in most cases, all the sensitive rules are hidden without generating false rules. Moreover, the good scalability of our approach in terms of database size and the influence of the correlation among rules on rule hiding are observed.

The remainder of this paper is organized as follows: In Section 2, the problem definition, the modification schemes, the system framework, and the basic properties for rule hiding are presented. In Section 3, we introduce the concept of a template, index construction and template generation, and the avoidance of side effects in rule hiding. Section 4 shows the experiments and results. Other related works are included in Section 5. Finally, we conclude the paper in Section 6. More properties for rule hiding and the associated proofs are listed in the Appendix.

2 Preliminaries

2.1 Problem Formulation

Let $\Gamma = \{i_1, i_2 \ldots i_n\}$ and $D = \{t_1, t_2 \ldots t_m\}$, where every t_j is a subset of Γ, be the set of all distinct items and the transaction database, respectively. Each transaction t_j is associated with a unique identifier called *TID* and can be represented as a bit-vector $b_1 b_2 \ldots b_n$, where $b_k = 1$ if $i_k \in t_j$.

Definition 2.1: Association rule mining. *The* count *of itemset I (denoted as C_I) is the number of transactions containing I in D, and the* database size *(denoted as $|D|$) is the number of transactions in D. For two itemsets X and Y, where $X \cap Y = \emptyset$, $X \rightarrow Y$ holds in D (strong rule) if both the following conditions hold, where X and Y are called the* precedent *and the* consequent*, respectively.*

1. $Sup_{X \cup Y} (= C_{X \cup Y}/|D|) \geq MST$ *and*
2. $Conf_{X \rightarrow Y} (= C_{X \cup Y}/C_X) \geq MCT$.

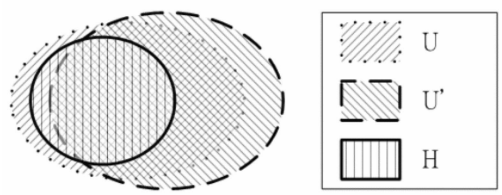


Fig. 1. The relationships among the rule sets U, U', and H.

Definition 2.2: A Class of modifications. *Given two transaction sets Σ_1 and Σ_2,* a class of modifications *is a function $\psi : (\Sigma_1, I, O) \rightarrow \Sigma_2$ that transforms Σ_1 to Σ_2, where I is the item(s) to be modified and O is the modification scheme.*

Definition 2.3: Association rule hiding. *Let D' be the database after applying a sequence of modifications to D. A strong rule $X \rightarrow Y$ in D will be* hidden *in D' if one of the following conditions holds in D':*

1. $Sup_{X \cup Y} < MST$ *and*
2. $Conf_{X \rightarrow Y} < MCT$.

Definition 2.4: Sensitive rule hiding under constraints. *With the fixed MST and MCT, U and U' denote the sets of strong rules in D and D', respectively. The rules in a subset H of U are selected as sensitive ones. We say that H is* safely hidden *in D' if all the following constraints are satisfied:*

1. $H \cap U' = \emptyset$ *(denoted as the constraint, F-T-H, i.e.,* Fail To be Hidden*),*
2. $(U - H) - U' = \emptyset$ *(denoted as the constraint, N-T-H, i.e.,* Not To be Hidden*), and*
3. $U' - U = \emptyset$ *(denoted as the constraint, N-T-G, i.e.,* Not To be Generated*).*

Fig. 1 shows the relationships among the rule sets, U, U', and H, where U-H is the set of nonsensitive rules. The three constraints ensure that three kinds of rules, i.e., sensitive rules that fail to be hidden, nonsensitive rules that are falsely hidden (*lost rules*), and spurious rules that are false generated (*false rules*), will not occur. The constraints N-T-H and N-T-G guard the database against the undesired side effects in the rule hiding process. Since the correlation among rules can make it impossible to hide rules without violating any constraint, in this paper, we relax the constraint F-T-H and allow the user to specify one or both of the constraints N-T-H and N-T-G. As a result, our work aims at hiding sensitive rules as many as possible without producing the side effects specified by the user. Moreover, the modification with fewer entries modified will be preferred in the proposed approach.

2.2 Modification Schemes for Rule Hiding

In the following, five modification schemes for rule hiding are introduced, respectively. For the ease of presentation, a transaction is sometimes viewed as a bit-vector as mentioned before.

Scheme 1: ***Modify entries from 1s to 0s.*** As mentioned in [6], if an item in $X \cup Y$ is deleted from a transaction containing $X \cup Y$, $Sup_{X \cup Y}$ and $Conf_{X \rightarrow Y}$ will be decreased. $X \rightarrow Y$ is hidden if we repeat this operation until one of the conditions in Definition 2.3 holds.

Scheme 2: ***Modify entries from 0s to 1s.*** As mentioned in [10], $Conf_{X \rightarrow Y}$ will be decreased if we insert an item $i \in X$ into a transaction that contains X but $\{i\}$ and does not contain Y. $X \rightarrow Y$ is hidden if we repeat this operation until condition 2 in Definition 2.3 holds.

Scheme 3: ***Modify entries from 1s to 0s or from 0s to 1s.*** Scheme 1 can guarantee to satisfy the constraint F-T-H, but Scheme 2 cannot. Both schemes may violate the other two constraints. Scheme 3 alternately uses them to decrease the supports and confidences of sensitive rules. Similarly, this scheme guarantees to satisfy the constraint F-T-H but may violate the others.

Scheme 4: ***Change 0s and 1s to ?s.*** As proposed in [21], for a transaction containing $X \cup Y$, if an item in $X \cup Y$ is replaced with an unknown, the minimum support of $X \cup Y$ and the minimum confidence of $X \rightarrow Y$ will be decreased. In addition, for a transaction that contains X but $\{i\}$ and does not contain Y, if the bit *0* denoting item i is replaced with an unknown, the minimum confidence of $X \rightarrow Y$ will be decreased. This scheme can guarantee to satisfy the constraint F-T-H but may violate the others.

Scheme 5: ***Swap 0 and 1 between two transactions.*** This is a special case of Scheme 3. For each item, the number of entries modified from *1* to *0* must be equal to the number of entries modified from *0* to *1*. In this way, the support of each item is unchanged after the rule hiding process. This characteristic can be useful for some applications such as the stock replenishment. However, due to its restriction, this scheme cannot satisfy the three constraints in most cases.

In this paper, we adopt Scheme 3 to modify the database, and use the constraints N-T-H and N-T-G as the guides to avoid the side effects. To consider both the numbers of hidden sensitive rules and modified entries, we include the correlation among rules in our modification scheme. The following lists the three considerations in hiding a sensitive rule, say $a \rightarrow b$:

1. Correlation with the other sensitive rule, say $c \rightarrow b$: From the transactions that contain abc, deleting b is better than deleting a since the former can affect both rules.
2. Correlation with a nonsensitive rule, say $b \rightarrow a$: Inserting a into the transactions that do not contain b is better than deleting a or b from the transactions containing ab because the latter may also hide $b \rightarrow a$.
3. Correlation with a spurious rule, say $ab \rightarrow c$: Inserting a into the transactions that do not contain b is better than deleting a or b from the transactions containing ab because the latter may also generate $ab \rightarrow c$.

2.3 System Framework

Fig. 2 shows the framework of our approach that consists of six components. Initially, the original database is converted into the *transaction table*, where each distinct item is encoded as a unique prime number. This conversion helps the efficiency and is detailed in Section 3.2. The *sensitive rule table* and the *nonsensitive rule table* are built to record the rule information. The *transaction-rule index* is also constructed using the concept of *inverted lists* [16] to correlate the tables for efficient retrieval.

The main challenge of rule hiding is how to select the items and transactions to modify. We propose to represent

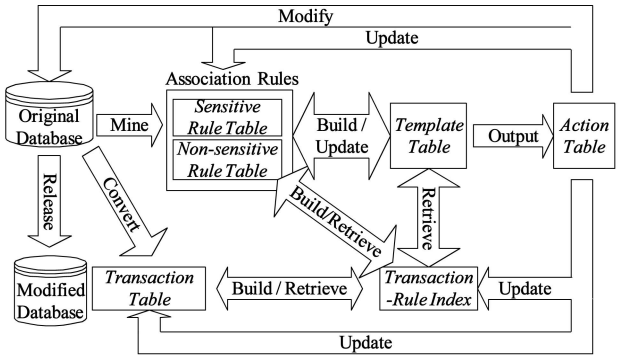


Fig. 2. The framework of our approach.

each class of modifications as a *template* and then select templates in a one-by-one fashion. The templates are kept in the *template table* and the selected templates are put into the *action table*. The six components are updated each time a template is selected. When all the sensitive rules are hidden or the template table is empty, the templates in the action table are applied to modify the original database. If some sensitive rules are not hidden, the user can release as it is, release nothing, or relax the constraint to hide more sensitive rules.

2.4 Properties for Rule Hiding

In the following, we present the properties that will be used to identify the valid modifications. Since Scheme 3 is a combination of Schemes 1 and 2, the properties are divided into two groups. Using Scheme 1 to hide a rule, the following properties indicate the minimal number of transactions that should be modified.

Property 1. *Let $\Sigma_{X\cup Y}$ be the set of all transactions containing $X \cup Y$. To hide $X \rightarrow Y$ by removing items in $X \cup Y$ from the transactions in $\Sigma_{X\cup Y}$, the minimal number of transactions that should be modified, called the minus support count, is computed as:*

$$MSC_{X\rightarrow Y} = C_{X\cup Y} - \left\lceil |D| \times MST \right\rceil + 1. \quad (1)$$

Proof. Removing items in $X \cup Y$ from the transactions in $\Sigma_{X\cup Y}$ will decrease $Sup_{X\cup Y}$. Let θ be the number of modified transactions when $X \rightarrow Y$ is hidden. By Definitions 2.1 and 2.3, we have:

$$(C_{X\cup Y} - \theta)/|D| < MST \Rightarrow C_{X\cup Y} - |D| \times MST < \theta.$$

$$\because \theta \text{ is an integer} \Rightarrow C_{X\cup Y} - \lceil |D| \times MST \rceil < \theta.$$
$$\therefore MSC_{X\rightarrow Y} = C_{X\cup Y} - \lceil |D| \times MST \rceil + 1. \quad \square$$

Property 2. *To hide $X \rightarrow Y$ by removing items in Y from the transactions in $\Sigma_{X\cup Y}$, the minimal number of transactions that should be modified, called the minus consequent confidence count, is computed as:*

$$MCCC_{X\rightarrow Y} = C_{X\cup Y} - \left\lceil C_X \times MCT \right\rceil + 1. \quad (2)$$

Proof. Removing items in Y from the transactions in $\Sigma_{X\cup Y}$ will decrease $Sup_{X\cup Y}$ but cannot decrease Sup_X. From Definition 2.1, $Conf_{X\rightarrow Y}$ will be decreased. Let θ be the number of modified transactions when $X \rightarrow Y$ is hidden. By Definition 2.3, we have:

$$(C_{X\cup Y} - \theta)/C_X < MCT \Rightarrow C_{X\cup Y} - C_X \times MCT < \theta.$$

$$\because \theta \text{ is an integer} \Rightarrow C_{X\cup Y} - \lceil C_X \times MCT \rceil < \theta.$$
$$\therefore MCCC_{X\rightarrow Y} = C_{X\cup Y} - \lceil C_X \times MCT \rceil + 1. \quad \square$$

Property 3. *To hide $X \rightarrow Y$ by removing items in X from the transactions in $\Sigma_{X\cup Y}$, the minimal number of transactions that should be modified, called the minus precedent confidence count, is computed as:*

$$MPCC_{X\rightarrow Y} = \left\lfloor (C_{X\cup Y} - C_X \times MCT)/(1 - MCT) \right\rfloor + 1. \quad (3)$$

Proof. Removing items in X from the transactions in $\Sigma_{X\cup Y}$ will decrease both $Sup_{X\cup Y}$ and Sup_X. By Definition 2.1, $Conf_{X\rightarrow Y}$, which is a proper fraction, will be decreased. Let θ be the number of modified transactions when $X \rightarrow Y$ is hidden. By Definition 2.3, we have:

$$(C_{X\cup Y} - \theta)/(C_X - \theta) < MCT \Rightarrow$$
$$C_{X\cup Y} - \theta < C_X \times MCT - \theta \times MCT \Rightarrow$$
$$\theta > (C_{X\cup Y} - C_X \times MCT)/(1 - MCT).$$

$$\because \theta \text{ is an integer}$$
$$\Rightarrow \theta > \lfloor (C_{X\cup Y} - C_X \times MCT)/(1 - MCT) \rfloor.$$

$$\therefore MPCC_{X\rightarrow Y} = \lfloor (C_{X\cup Y} - C_X \times MCT)/(1 - MCT) \rfloor + 1. \quad \square$$

Definition 2.5: Minus count. *The minimum of $MCCC_{X\rightarrow Y}$ and $MPCC_{X\rightarrow Y}$ is called the* minus confidence count *of $X \rightarrow Y$ ($MCC_{X\rightarrow Y}$). The minimum of $MSC_{X\rightarrow Y}$ and $MCC_{X\rightarrow Y}$ is called the* minus count *of $X \rightarrow Y (MC_{X\rightarrow Y})$, indicating the minimal number of transactions to be modified by Scheme 1 for hiding $X \rightarrow Y$.*

With Scheme 1, a rule can be eventually hidden if the side effects are ignored. Using Scheme 2 to hide a sensitive rule, we also have the following property to estimate the minimal number of transactions that should be modified.

Property 4. *Let Σ_{X-Y} be the set of all transactions that contain X but do not contain Y. To hide $X \rightarrow Y$ by adding items in X to introduce new transactions in Σ_{X-Y}, the minimal number of transactions that should be modified, called the plus precedent confidence count, is computed as:*

$$PPCC_{X\rightarrow Y} = \left\lfloor C_{X\cup Y}/MCT \right\rfloor - C_X + 1. \quad (4)$$

Proof. Raising the number of transactions in Σ_{X-Y} will increase Sup_X but cannot increase $Sup_{X\cup Y}$. By Definition 2.1, $Conf_{X\rightarrow Y}$ will be decreased. Let θ be the number of modified transactions when $X \rightarrow Y$ is hidden. By Definition 2.3, we have:

$$C_{X\cup Y}/(C_X+\theta) < MCT \Rightarrow C_{X\cup Y} - C_X \times MCT < \theta \times MCT$$
$$\Rightarrow \theta > C_{X\cup Y}/MCT - C_X.$$

$$\because \theta \text{ is an integer} \Rightarrow \theta > \lfloor C_{X\cup Y}/MCT \rfloor - C_X.$$
$$\therefore PPCC_{X\to Y} = \lfloor C_{X\cup Y}/MCT \rfloor - C_X + 1. \quad \square$$

Definition 2.6: Plus count. *Compared with Scheme 1, Scheme 2 only uses the support of the precedent in a rule to decrease the confidence. Therefore, we consider $PPCC_{X\to Y}$ as the minimal number of transactions to be modified by Scheme 2 for hiding $X \to Y$, and call it the* plus count *of $X \to Y (PC_{X\to Y})$.*

For each sensitive rule, based on the minus count and the plus count, our approach can determine the minimal number of transactions that should be modified. In a similar way, the maximal number of transactions that can be modified without hiding a nonsensitive rule can be estimated. Note that, in the cases for nonsensitive rules, the "plus one" operation at each of the four equations (1)-(4) must be omitted. On the other hand, to avoid generating a spurious rule, we have the following properties to estimate the maximal number of transactions that can be modified by Schemes 1 and 2, respectively.

Property 5. *Let Σ_{Y-Z} be the set of all transactions that contain Y but do not contain Z. To hide sensitive rules without generating $Y \to Z$, the maximal number of transactions in Σ_{Y-Z} that can be modified by removing items in Y, called the removable precedent confidence count, is computed as:*

$$RPCC_{Y\to Z} = C_Y - \left\lfloor C_{Y\cup Z}/MCT \right\rfloor - 1. \tag{5}$$

Proof. Removing items in Y from the transactions in Σ_{Y-Z} will decrease Sup_Y but cannot decrease $Sup_{Y\cup Z}$. From Definition 2.1, $Conf_{Y\to Z}$ will be increased. Let θ be the number of modified transactions when $Y \to Z$ is generated. By Definition 2.1, we have:

$$C_{Y\cup Z}/(C_Y-\theta) \geq MCT \Rightarrow C_{Y\cup Z} \geq C_Y \times MCT - \theta \times MCT$$
$$\Rightarrow \theta \geq C_Y - C_{Y\cup Z}/MCT.$$

$$\because \theta \text{ is an integer} \Rightarrow \theta \geq C_Y - \lfloor C_{Y\cup Z}/MCT \rfloor.$$
$$\therefore RPCC_{Y\to Z} = C_Y - \lfloor C_{Y\cup Z}/MCT \rfloor - 1. \quad \square$$

Property 6. *To hide rules by adding items in $Y \cup Z$ to some transactions without generating $Y \to Z$, we have the following cases to consider:*

1. *When $Sup_{Y\cup Z}$ is increased, the maximal number of transactions that can be modified, called the addible support count, can be computed as:*

$$ASC_{Y\to Z} = \left\lceil |D| \times MST \right\rceil - C_{Y\cup Z} - 1. \tag{6}$$

2. *When $Conf_{Y\to Z}$ is increased by adding items in Z, the maximal number of transactions that can be modified, called addible consequent confidence count, is:*

$$ACCC_{Y\to Z} = \left\lceil C_Y \times MCT \right\rceil - C_{Y\cup Z} - 1. \tag{7}$$

3. *When $Conf_{Y\to Z}$ is increased by adding items in Y, the maximal number of transactions that can be modified, called the addible precedent confidence count, can be computed as:*

$$APCC_{Y\to Z} = \left\lceil (C_Y \times MCT - C_{Y\cup Z})/(1-MCT) \right\rceil - 1. \tag{8}$$

Proof. Let θ be the number of modified transactions when $Y \to Z$ is generated. By Definition 2.1, we have:

1. $Sup_{Y\cup Z}$ is increased

$$\Rightarrow (C_{Y\cup Z}+\theta)/|D| \geq MST$$
$$\Rightarrow \theta \geq |D| \times MST - C_{Y\cup Z.}$$

$$\because \theta \text{ is an integer} \Rightarrow \theta \geq \lceil |D| \times MST \rceil - C_{Y\cup Z}.$$
$$\therefore ASC_{Y\to Z} = \lceil |D| \times MST \rceil - C_{Y\cup Z} - 1.$$

2. $Conf_{Y\to Z}$ is increased by adding items in Z to form transactions in $\Sigma_{Y\cup Z}$

$$\Rightarrow (C_{Y\cup Z}+\theta)/C_Y \geq MCT \Rightarrow$$
$$\theta \geq C_Y \times MCT - C_{Y\cup Z}.$$

$$\because \theta \text{ is an integer} \Rightarrow \theta \geq \lceil C_Y \times MCT \rceil - C_{Y\cup Z}.$$
$$\therefore ACCC_{Y\to Z} = \lceil C_Y \times MCT \rceil - C_{Y\cup Z} - 1.$$

3. $Conf_{Y\to Z}$ is increased by adding items in Y to form transactions in $\Sigma_{Y\cup Z}$

$$\Rightarrow (C_{Y\cup Z}+\theta)/(C_Y+\theta) \geq MCT \Rightarrow$$
$$C_{Y\cup Z} + \theta \geq C_Y \times MCT + \theta \times MCT \Rightarrow$$
$$\theta \geq (C_Y \times MCT - C_{Y\cup Z})/(1-MCT).$$

$$\because \theta \text{ is an integer}$$
$$\Rightarrow \theta \geq \lceil (C_Y \times MCT - C_{Y\cup Z})/(1-MCT) \rceil.$$

$$\therefore \; APCC_{Y\to Z} = \lceil (C_Y \times MCT - C_{Y\cup Z})/(1-MCT) \rceil - 1. \quad \square$$

Definition 2.7: Removable count and addible count. *The maximum of $ACCC_{Y\to Z}$ and $APCC_{Y\to Z}$ is called the* addible confidence count *of $Y \to Z (ACC_{Y\to Z})$. The maximum of $ASC_{Y\to Z}$ and $ACC_{Y\to Z}$ is called the* addible count *of $Y \to Z (AC_{Y\to Z})$, indicating the maximal number of transactions to be modified by Scheme 2 without generating $Y \to Z$. By contrast, $RPCC_{Y\to Z}$ stands for the maximal number of transactions to be modified by Scheme 1 without generating $Y \to Z$ and is called the* removable count *$(RC_{Y\to Z})$.*

Given a sensitive rule r, MC_r and PC_r act as the necessary conditions to determine whether a class of modifications can be used to hide r. Let $R_{N\text{-}T\text{-}H}$ and $R_{N\text{-}T\text{-}G}$, respectively, denote the sets of nonsensitive rules and spurious rules that have common items in r. For any $i \in R_{N\text{-}T\text{-}H}$ and $j \in R_{N\text{-}T\text{-}G}$, MC_i, PC_i, AC_j, and RC_j act as the constraints, deciding whether a class of modifications can safely hide r. In some cases, a sensitive rule cannot be safely hidden due to the correlation among rules. More properties for such cases are listed in the Appendix.

3 Our Approach for Rule Hiding

3.1 Classification of Modifications—Templates

As described in Definition 2.2, the rule hiding process can be viewed as a series of modifications. A class of modifications can be expressed in a logical form if we regard each transaction as a bit vector. For example, deleting items in Y from the transactions containing $X \cup Y$ can be expressed as: $X \wedge Y \Rightarrow X \wedge \neg Y$, where an itemset $\{x_1, x_2 \ldots x_n\}$ is expressed in the form of $x_1 \wedge x_2 \ldots x_n$.

Definition 3.1: Logical form. *Given itemsets T, $F_1 \ldots F_m$, and two transaction sets Σ_1 and Σ_2, where $\forall t \in \Sigma_1 \cup \Sigma_2$, $T \subseteq t$, $F_1 \not\subset t \ldots F_m \not\subset t$, a class of modifications $\psi : (\Sigma_1, \{i\}, O) \rightarrow \Sigma_2$ can be expressed in one of the logical forms:*

1. $i \wedge T \wedge \neg F_1 \wedge \ldots \neg F_m \Rightarrow \neg i \wedge T \wedge \neg F_1 \wedge \ldots \neg F_m$ if $O = Scheme\ 1$ and $\forall t \in \Sigma_1$, $i \in t$.
2. $\neg i \wedge T \wedge \neg F_1 \wedge \ldots \neg F_m \Rightarrow i \wedge T \wedge \neg F_1 \wedge \ldots \neg F_m$, if $O = Scheme\ 2$ and $\forall t \in \Sigma_1$, $i \notin t$.

The logical form for a class of modifications includes the modification scheme (O), the item to be modified (i), and the predicates (itemsets) describing the transactions to be modified $(T, F_1 \ldots F_m)$. The modification scheme is either $Scheme\ 1$ (denoted as Del) or $Scheme\ 2$ (denoted as Ins). The item to be modified is called the *modified item* (*MI*). The predicate specifying the items that must be contained in the transactions is called the *positive part* (*PP*). On the contrary, the predicates specifying the itemsets that must not appear in the transactions form the *negative part* (*NP*). Based on the logical forms, we propose representing a class of modifications as follows:

Definition 3.2: Template. *A class of modifications $\psi : (\Sigma_1, \{i\}, O) \rightarrow \Sigma_2$ is represented as a* template

$$\tau =< MI, O, PP, NP, CV, DF >,$$

where the last two components, CV and DF, will be defined later.

For example, deleting item c from the transactions containing $abcd$ to hide $ab \rightarrow cd$ can be represented as the template $< c, Del, abd, \{\}, *, * >$, where symbol $*$ denotes an undermined value. Another way to hide this rule is to insert item b into the transactions that contain a but do not contain b and cd. Its template is $< b, Ins, a, \{cd\}, *, * >$, where cd (the NP) means that the transactions associated with the template cannot have both c and d, but may contain one of them. Complex predicates may also be used in a template. For instance, $< b, Ins, a, \{cd, ef\}, *, * >$ hides both rules $ab \rightarrow cd$ and $ab \rightarrow ef$, while $< b, Del, acd, \{\}, *, * >$ hides both rules $abc \rightarrow d$ and $ab \rightarrow cd$.

Definition 3.3: Transaction containment. *A transaction t is* contained *in a template τ if both the following conditions hold, where the set of all transactions contained by τ is denoted as T_τ:*

1. $PP_\tau \subseteq t$ *and* $\forall F \in NP_\tau$, $F \not\subset t$.
2. $MI_\tau \in t$ *if* $O_\tau = Del$; $MI_\tau \notin t$ *if* $O_\tau = Ins$.

Definition 3.4: Sensitive rule coverage. *A sensitive rule $r : X \rightarrow Y$ is* covered *by a template τ if $Sup_{X \cup Y}$ or $Conf_r$ will be decreased as we modify MI_τ in transaction t by $O_\tau \forall t \in T_\tau$. The set of all the sensitive rules covered by τ is denoted as SR_τ and the number of rules in SR_τ is called the* coverage *of $\tau(|SR_\tau|)$.*

The coverage of a template is the fifth component (CV) in its template representation. From the above, a template τ correlates a set of sensitive rules SR_τ with a set of transactions T_τ in such a way that modifying the transactions in T_τ can help to hide the sensitive rules in SR_τ. To reduce the number of modified entries, we estimate the minimal number of transactions in T_τ that should be modified to hide all the sensitive rules in SR_τ as follows:

TABLE 1
A Sample Set of Sensitive Rules

Sensitive Rule	MSC	MPCC	MCCC	PPCC
a→b	4	4	3	5
ab→d	3	3	2	2
a→cd	2	4	3	3

Definition 3.5: Minus count of a template. *Given template τ and a rule $r : X \rightarrow Y \in SR_\tau$,* if *$O_\tau$ is* Del, the minus count of *$\tau(MC_{r,\tau})$ is as follows:*

1. $MC_{r,\tau} = min\{MSC_r, MPCC_r\}$ if $MI_\tau \in X$.
2. $MC_{r,\tau} = min\{MSC_r, MCCC_r\}$ if $MI_\tau \in Y$.

Definition 3.6: Difference of a template. *The* difference of a template *τ is defined as the maximum of $MC_{r,\tau} \forall r \in SR_\tau$, if O_τ is Del; or the maximum of $PC_r \forall r \in SR_\tau$, if O_τ is Ins. The difference is the sixth component (DF) of a template.*

Example 2. Table 1 shows three sensitive rules and the corresponding values of MSC, $MPCC$, $MCCC$, and $PPCC$. The templates in Table 2 are generated from it, where $\tau 1 \sim \tau 3$, $\tau 4 \sim \tau 8$, and $\tau 9 \sim \tau 12$ are derived from $a \rightarrow b$, $ab \rightarrow d$, and $a \rightarrow cd$, respectively. The CV field is computed from Definition 3.4, while the DF field is computed according to Definitions 3.5 and 3.6.

Considering $\tau 3$ as an example, by Definition 3.4, only one sensitive rule $(a \rightarrow b)$ is covered by it. Since O is Del and MI is the consequent b, by Definition 3.6 the DF of $\tau 3$ is equal to $MC_{a \rightarrow b, \tau 3}$, which is the minimum of $MSC_{a \rightarrow b}$ and $MCCC_{a \rightarrow b}$ by Definition 3.5. Observing the logical forms of $\tau 5$ and $\tau 7$ in Table 2, we find that the modification in the form of $\neg a \wedge \neg b \wedge \neg d \Rightarrow a \wedge b \wedge \neg d$ has the same impact on hiding $ab \rightarrow d$ because it inserts items a and b to the transactions that do not contain a, b, and d. We call it the *auxiliary template*. To keep the number of modified entries small, we generate the auxiliary templates only when the templates currently generated are not sufficient to hide all the sensitive rules.

Given a number of templates, we adopt a heuristic method to select the one with the lowest DF as the *pivot*. If several templates have the lowest DF, we select the one with the largest CV. Moreover, if several templates have the largest CV, we randomly select one of them because they have the same impacts on the number of hidden sensitive rules and the number of modified entries. After executing the modifications of the pivot, the set of sensitive rules will

TABLE 2
The Templates Generated from Table 1

	MI	O	PP	NP	CV	DF	Logical Form
τ1	*a*	*Del*	*b*	{}	*1*	*4*	$a \wedge b \Rightarrow \neg a \wedge b$
τ2	*a*	*Ins*	∅	{*b*}	*1*	*5*	$\neg a \wedge \neg b \Rightarrow a \wedge \neg b$
τ3	*b*	*Del*	*a*	{}	*1*	*3*	$a \wedge b \Rightarrow a \wedge \neg b$
τ4	*a*	*Del*	*bd*	{}	*2*	*4*	$a \wedge b \wedge d \Rightarrow \neg a \wedge b \wedge d$
τ5	*a*	*Ins*	*b*	{*d*}	*2*	*3*	$\neg a \wedge b \wedge \neg d \Rightarrow a \wedge b \wedge \neg d$
τ6	*b*	*Del*	*ad*	{}	*2*	*3*	$a \wedge b \wedge d \Rightarrow a \wedge \neg b \wedge d$
τ7	*b*	*Ins*	*a*	{*d*}	*1*	*2*	$a \wedge \neg b \wedge \neg d \Rightarrow a \wedge b \wedge \neg d$
τ8	*d*	*Del*	*ab*	{}	*1*	*2*	$a \wedge b \wedge d \Rightarrow a \wedge b \wedge \neg d$
τ9	*a*	*Del*	*cd*	{}	*1*	*2*	$a \wedge c \wedge d \Rightarrow \neg a \wedge c \wedge d$
τ10	*a*	*Ins*	∅	{*cd*}	*1*	*3*	$\neg a \wedge \neg(c \wedge d) \Rightarrow a \wedge \neg(c \wedge d)$
τ11	*c*	*Del*	*ad*	{}	*1*	*2*	$a \wedge c \wedge d \Rightarrow a \wedge \neg c \wedge d$
τ12	*d*	*Del*	*ac*	{}	*1*	*2*	$a \wedge c \wedge d \Rightarrow a \wedge c \wedge \neg d$

shrink. For each of the remaining templates, the values of CV and DF are recomputed for the next run of *pivot selection*. Since the coverage of a template can be too small to hide many sensitive rules at a time, we introduce a way to integrate multiple templates into one to cover more sensitive rules.

Definition 3.7: Joint template. *Two templates* $\tau 1$ *and* $\tau 2$ *are* joinable *if* $MI_{\tau 1} = MI_{\tau 2}$, $O_{\tau 1} = O_{\tau 2}$, $\forall F \in NP_{\tau 1}$, $F \cap PP_{\tau 2} = \emptyset$, *and* $\forall F' \in NP_{\tau 2}$, $F' \cap PP_{\tau 1} = \emptyset$. *The result is named the* joint template, *which is denoted as*

$$\tau 1 \oplus \tau 2 = < MI_{\tau 1}, O_{\tau 1}, PP_{\tau 1} \cup PP_{\tau 2}, NP_{\tau 1} \cup NP_{\tau 2}, \\ |SR_{\tau 1} \cup SR_{\tau 2}|, max\{DF_{\tau 1}, DF_{\tau 2}\} > .$$

Example 3. In Table 2, $\tau 4$ and $\tau 9$ are joinable. The joint template $\tau 4 \oplus \tau 9$ is $< a, Del, bcd, \{\}, 3, 4 >$ with the logical form $a \wedge b \wedge c \wedge d \Rightarrow \neg a \wedge b \wedge c \wedge d$. Obviously, this template is better than $\tau 4$ since it has the same DF but covers all the rules in Table 1. Similarly, the joint template $\tau 5 \oplus \tau 10$ is $< a, Ins, b, \{d, cd\}, 2, 3 >$ with the logical form $\neg a \wedge b \wedge \neg d \wedge \neg(c \wedge d) \Rightarrow a \wedge b \wedge \neg d \wedge \neg(c \wedge d)$. Note that the transactions that do not contain d will not contain cd, either. Therefore, the term $\neg(c \wedge d)$ in the logical form can be discarded and result in a simpler one, which happens to be $\tau 5$. In this way, we can reduce the joint template to its simplest form and avoid generating redundant templates.

Definition 3.8: Template reduction. *We say that a template* τ *is* reducible *if* $\exists F_i, F_j \in NP_\tau$, *where* $i \neq j$, $F_i \subseteq F_j$. *To reduce* τ, *we simply remove all* $F_j \in NP_\tau$ *if* $\exists F_i \in NP_\tau$, $F_i \subseteq F_j$.

TABLE 3
A Transaction Database in the Form of Bit-Vectors

TID	
1	*abe*
2	*ce*
3	*ace*
4	*abde*
5	*b*
6	*ab*

⇒

TID	*a*	*b*	*c*	*d*	*e*
1	*1*	*1*	*0*	*0*	*1*
2	*0*	*0*	*1*	*0*	*1*
3	*1*	*0*	*1*	*0*	*1*
4	*1*	*1*	*0*	*1*	*1*
5	*0*	*1*	*0*	*0*	*0*
6	*1*	*1*	*0*	*0*	*0*

TABLE 4
The Frequent Itemsets and the Strong Rules

Frequent itemsets	Strong rules
a(4), *b*(4), *c*(2), *e*(5), *ab*(3), *ae*(3), *be*(2), *ce*(2), *abe*(2)	*a*→*b*(3/4), *b*→*a*(3/4), *a*→*e*(3/4), *e*→*a*(3/5), *c*→*e*(2/2), *ab*→*e*(2/3), *ae*→*b*(2/3), *be*→*a*(2/2),

3.2 Index Construction and Template Generation

Suppose that a database at the left-hand side of Table 3 is given and represented in the form of bit-vectors at the right-hand side. We apply to it an algorithm for association rule mining with *MST 20%* and *MCT 60%* and obtain the frequent itemsets and the strong rules in Table 4, where the values in parentheses stand for counts and confidences, respectively.

We encode each item as a unique prime number such that all the transactions and rules are converted into products of prime numbers. For instance, using the encoding table at the left-hand side of Table 5, the database is converted to the one at the right-hand side, where nonfrequent item d is omitted during the conversion. Note that the frequent items are sorted by their counts and mapped to the prime numbers in reverse order (e.g., $e \rightarrow 5$). In this way, the product of prime numbers stored in the index and tables will not be too large. After that, the transactions are stored as the transaction table in Table 6, where each bucket keeps a distinct product and a count to indicate the number of transactions with this product.

Not only the transactions but also the rules are represented in the form of prime numbers or products of prime numbers. As depicted in Fig. 2, the strong rules are distributed into two tables. Assuming that from Table 4 the user selects $a \rightarrow b$, $b \rightarrow a$, and $a \rightarrow e$ as sensitive rules, the two kinds of rules are separately stored in Table 7 and Table 8. Note that the column *Itemset* of a rule $X \rightarrow Y$ stands for the union of the precedent X and the consequent Y and, therefore, it keeps the product of two prime numbers that, respectively, come from X and Y.

Since the number of sensitive rules is often much less than the number of nonsensitive rules, we do not keep the sensitive rules in the transaction-rule index. Fig. 3 shows an example indexing the transactions in Table 6 and the nonsensitive rules in Table 8. With this index, the transactions and nonsensitive rules satisfying a logical form can be quickly identified. For instance, the transactions that contain itemset 6 but do not contain itemset 35

TABLE 5
The Encoding Table and the Converted Database

Encoding Table
a→*2*
b→*3*
e→*5*
c→*7*

⇒

TID	*2*	*3*	*5*	*7*	Product
1	*1*	*1*	*1*	*0*	*30=2×3×5*
2	*0*	*0*	*1*	*1*	*35=5×7*
3	*1*	*0*	*1*	*1*	*70=2×5×7*
4	*1*	*1*	*1*	*0*	*30=2×3×5*
5	*0*	*1*	*0*	*0*	*3*
6	*1*	*1*	*0*	*0*	*6=2×3*

TABLE 6
The Transaction Table

Bucket Number	Product	Count
1	30	2
2	35	1
3	70	1
4	3	1
5	6	1

TABLE 7
The Sensitive Rule Table

Bucket Number	Precedent	Consequent	Itemset	Support	Confidence
1	2	3	6	3/6	3/4
2	3	2	6	3/6	3/4
3	2	5	10	3/6	3/4

are in the first and the fifth buckets of Table 6 since $(10101) \wedge (10011) \wedge \neg((11100) \wedge (01100)) = (10001)$.

After the index and tables are built, the templates are generated as described in Section 3.1. For example, the templates $\tau 1 \sim \tau 7$ in Table 9 come from the rules in Table 7. Note that we store value *1* if PP is empty. By Definition 3.7, we join $\tau 1$ and $\tau 5$ to generate $\tau 8$, where $PP_{\tau 8}$ is a product of $PP_{\tau 1}$ and $PP_{\tau 5}$. Similarly, we join $\tau 3$ and $\tau 7$ to generate $\tau 9$. Finally, by Definition 3.8, each template in Table 9 is not reducible.

Fig. 4 shows the algorithm for computing the values of CV and DF. For each template, we retrieve the sensitive rules covered by it according to Definition 3.4. The criteria to identify the coverage are shown in line 4 and line 9, and they can be computed via simple divisions. Take $\tau 8$ in Table 9 as an example, where the rule $a \rightarrow b$ is covered by it because $MI_{\tau 8} \in \{a, b\}$ and $\{a, b\} \subseteq \{MI_{\tau 8}\} \cup PP_{\tau}$. Since $\tau 8$ covers all the rules in Table 7 and $O_{\tau 8}$ is *Del*, lines $4 \sim 7$ are executed for each sensitive rule. From the sensitive rules, the coverage and the difference of the template are computed according to Definitions 3.5 and 3.6. For instance, in line 6, $MC_{a \rightarrow b, \tau 8}$ is computed as the minimum of $MSC_{a \rightarrow b} (= 2)$ and $MPCC_{a \rightarrow b} (= 2)$. For the other sensitive rules covered by $\tau 8$, $MC_{b \rightarrow a, \tau 8}$ and $MC_{a \rightarrow e, \tau 8}$ are *1* and *2*, respectively. As a result, $CV_{\tau 8}$ and $DF_{\tau 8}$ are *3* and *2*, respectively. Note that a template is removed if its coverage or difference is zero.

The efficiency of template generation depends on both the numbers of sensitive rules and items in the individual sensitive rules. If there are common items among the sensitive rules, the costs in this stage can be reduced. Besides, in our application scenario, the number of sensitive rules specified by the user will not be too large. Moreover, the index also helps the efficiency.

TABLE 8
The Nonsensitive Rule Table

Bucket Number	Precedent	Consequent	Itemset	Support	Confidence
1	5	2	10	3/6	3/5
2	7	5	35	2/6	2/2
3	6	5	30	2/6	2/3
4	10	3	30	2/6	2/3
5	15	2	30	2/6	2/2

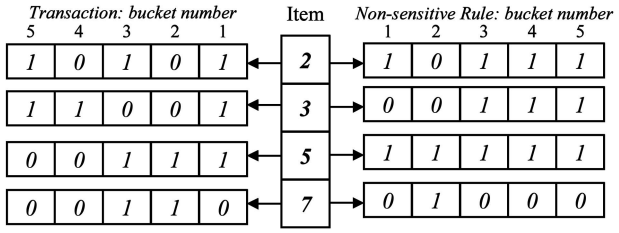


Fig. 3. The transaction-rule index.

TABLE 9
The Template Table

	MI	O	PP	NP	CV	DF	Logical Form
τ1	2	Del	3	{}	2	2	$a \wedge b \Rightarrow \neg a \wedge b$
τ2	3	Del	2	{}	2	2	$a \wedge b \Rightarrow a \wedge \neg b$
τ3	2	Ins	1	{3}	1	2	$\neg a \wedge \neg b \Rightarrow a \wedge \neg b$
τ4	3	Ins	1	{2}	1	2	$\neg a \wedge \neg b \Rightarrow \neg a \wedge b$
τ5	2	Del	5	{}	1	2	$a \wedge e \Rightarrow \neg a \wedge e$
τ6	5	Del	2	{}	1	2	$a \wedge e \Rightarrow a \wedge \neg e$
τ7	2	Ins	1	{5}	1	2	$\neg a \wedge \neg e \Rightarrow a \wedge \neg e$
τ8	2	Del	15	{}	3	2	$a \wedge b \wedge e \Rightarrow \neg a \wedge b \wedge e$
τ9	2	Ins	1	{3, 5}	2	2	$\neg a \wedge \neg b \wedge \neg e \Rightarrow a \wedge \neg b \wedge \neg e$

3.3 Avoidance of Side Effects

Definition 3.9: Transaction group. *For a template τ, the transactions in T_τ can be divided into groups such that the transactions in a group are identical. A* transaction group π *is represented as $< I\pi, N\pi >$, where I_π denotes the set of distinct items in π and N_π keeps the total number of transactions in π.*

Definition 3.10: Sensitive rule conflict. *Given a template τ, we say that a transaction group $\pi \subseteq T_\tau$* conflicts *with a sensitive rule $r : X \rightarrow Y$ if $Sup_{X \cup Y}$ or $Conf_r$ will be increased when we modify MI_τ in t by $O_\tau \forall t \in \pi$.*

Consider $\tau 5$ in Example 2 and a transaction group $\pi \subseteq T_{\tau 5}$. Modifying π can hide $ab \rightarrow d$ and $a \rightarrow cd$, but it also increases the support of ab and makes it difficult to hide $a \rightarrow b$ in the next run. Therefore, among the transaction

```
Algorithm CV-DF Computation
INPUT: the template table T
OUTPUT: the template table T with coverage
CVτ and difference DFτ ∀τ∈T
1. ∀τ∈T
2.   CVτ = DFτ = 0
3.   If (Oτ = Del)
4.   ∀r:X→Y∈SRτ //MIτ∈X∪Y and X∪Y⊆{MIτ}∪PPτ
5.     CVτ++
6.     Compute MCr,τ        //By Definition 3.5
7.     DFτ= max{DFτ, MCr,τ}//By Definition 3.6
8.   Else If (Oτ = Ins)
9.   ∀r:X→Y∈SRτ //MIτ∈X, X−{MIτ}⊆PPτ, and
                 ∃I∈NPτ, Y∩I≠∅
10.    CVτ++
11.    Compute PCr          //By Definition 3.5
12.    DFτ= max{DFτ, PCr}//By Definition 3.6
13.    If ((CVτ==0) or (DFτ==0))   T= T − {τ}
14. End
```

Fig. 4. The computation of CV and DF.

groups, we filter out those who conflict with sensitive rules. This operation is executed when a template is generated. After that, for each transaction group, we estimate the maximal number of transactions that can be modified without producing side effects. A rule that can be affected by the insertion or deletion of an item must have that item in its precedent or consequent. Therefore, we use MI_τ and I_π to identify all the nonsensitive rules and spurious rules that can be affected by modifying the transactions in π. Finally, two upper bounds on the number of transactions in π to be modified are computed.

Definition 3.11: Nonsensitive rule coverage. *Given τ and π, a nonsensitive rule $r : X \rightarrow Y$ is* covered *by π if $Sup_{X \cup Y}$ or $Conf_r$ will be decreased as we modify MI_τ in transaction t by $O_\tau \forall t \in \pi$. The set of all nonsensitive rules covered by π is denoted as LR_π and the number of rules in LR_π is called the* coverage *of $\pi(|LR_\pi|)$.*

Definition 3.12: Minus count of a transaction group. *Given τ, π, and $r : X \rightarrow Y \in LR_\pi$, if O_τ is* Del, *we define the* minus count of $\pi(MC_{r,\pi})$ *as follows:*

1. $MC_{r,\pi} = min\{MSC_r, MPCC_r\}$ if $MI_\tau \in X$.
2. $MC_{r,\pi} = min\{MSC_r, MCCC_r\}$ if $MI_\tau \in Y$.

Definition 3.13: N-T-H bound of a transaction group. *Given τ and π, the N -T-H* bound of *π (denoted as $B_{N\text{-}T\text{-}H}$) is defined as the minimum of $MC_{r,\pi} \forall r \in LR_\pi$, if O_τ is* Del *or the minimum of $PC_r \forall r \in LR_{\pi\tau}$, if O_τ is* Ins.

Like the approach to template selection, among the transaction groups of the pivot, we use a heuristic method to select the one with the highest $B_{N\text{-}T\text{-}H}$ as the candidate. If several transaction groups have the highest $B_{N\text{-}T\text{-}H}$, we select the one with the smallest coverage. Moreover, if several transaction groups have the smallest coverage, we randomly select one of them because they give the same upper bound on the number of transactions to be modified and the same number of affected nonsensitive rules. As a candidate π is selected, we further check the spurious rules that can be affected by modifying the transactions in π and compute another upper bound. Since there can be a huge number of spurious rules, we use the association graph proposed in [26] to reduce the search space. The association graph, where each node represents a frequent item and each link corresponds to a frequent itemset, is built in association rule mining. By the antimonotone property [2], we can prune some spurious rules as follows:

Property 7. *If O_τ is Del, $X \rightarrow Y$ can be pruned if one of the following conditions holds:*

1. $\exists W \subseteq X \cup Y, Sup_W < MST$.
2. $\exists X \rightarrow Z, Z \subseteq Y$ *and* $Conf_{X \rightarrow Z} < MCT$.

Proof.

1. $W \subseteq X \cup Y \Rightarrow C_{X \cup Y} \leq C_W \Rightarrow Sup_{X \cup Y} \leq Sup_W < MST$.
   $\therefore X \rightarrow Y$ can be pruned.
2. $Z \subseteq Y \Rightarrow X \cup Z \subseteq X \cup Y \Rightarrow C_{X \cup Y} \leq C_{X \cup Z} \Rightarrow Conf_{X \rightarrow Y} \leq Conf_{X \rightarrow Z} < MCT$.
   $\therefore X \rightarrow Y$ can be pruned. □

Definition 3.14: Spurious rule coverage. *Given τ and π, a spurious rule $r : X \rightarrow Y$ is* covered *by π if both the following conditions hold, where the set of all spurious rules covered by π is denoted as FR_π.*

1. *The rule r does not satisfy any condition in Property 7.*
2. *As we modify MI_τ in transaction t by $O_\tau \forall t \in \pi$, $Sup_{X \cup Y}$ or $Conf_r$ will be increased.*

Definition 3.15: Addible count of a transaction group. *Given τ, π, and $r : X \rightarrow Y \in FR_\pi$, if O_τ is* Ins, *we define the* addible count of $\pi(AC_{r,\pi})$ *as follows:*

1. $AC_{r,\pi} = max\{ASC_r, APCC_r\}$ if $MI_\tau \in X$.
2. $AC_{r,\pi} = max\{ASC_r, ACCC_r\}$ if $MI_\tau \in Y$.

Definition 3.16: N-T-G bound of a transaction group. *Given τ and π, the N-T -G* bound of *π (denoted as $B_{N\text{-}T\text{-}G}$) is defined as the minimum of $AC_{r,\pi}$ $\forall r \in FR_\pi$, if O_τ is* Ins*; or, the minimum of $RC_r \forall r \in FR_\pi$, if O_τ is* Del.

Recall that the user can specify one or both of the constraints N-T-H and N-T-G before the rule hiding process. If no constraint is specified, we use N_π as the upper bound, indicating that all the transactions in π can be modified. Let δ be the minimum of $B_{N\text{-}T\text{-}H}$ and $B_{N\text{-}T\text{-}G}$, indicating that at most δ transactions in π can be modified without producing side effects. If δ is nonzero, we output a record $< \tau, \pi, min\{\delta, DF_\tau\} >$ to the action table, which means modifying MI_τ in the transactions of π by O_τ and the number of transactions to be modified is set to $min\{\delta, DF_\tau\}$. We simulate these modifications on the transaction table and also update the sensitive rule table and the transaction-rule index. After that, we decrease DF_τ by δ and select the next candidate from the remaining transaction groups if DF_τ is nonzero. Until DF_τ is zero or no candidate can be found, we return to select the next pivot from the remaining templates. A complete flowchart of our approach is shown in Fig. 5.

4 Performance Evaluation

We adopt the *IBM data generator* [28] to synthesize the databases for the experiments. The average transaction length and the database size are set to *10* and *5K*, respectively. We first generate a database with size *5K*, and then duplicate it to generate the databases with size *10K*, *15K*, *20K*, and *25K*. Both the number of frequent itemsets and the number of distinct items are set to *1K*. Using the fixed thresholds *MST 1%* and *MCT 60%*, there are *1,564* strong rules. We randomly select five of them as sensitive rules and make experiments on Pentium IV *2.8GHz* PC with *256MB* RAM running Windows XP. Our approach is programmed in C# with four variations, including no constraint (*None*), no loss rule (N-T-H), no false rule (N-T-G), and both constraints (*Both*). Each measurement in the experimental results is an average computed from *10* trials.

To evaluate the efficiency, we measure the CPU time of the entire process including the preprocessing stages. On the other hand, we use four measures to evaluate the

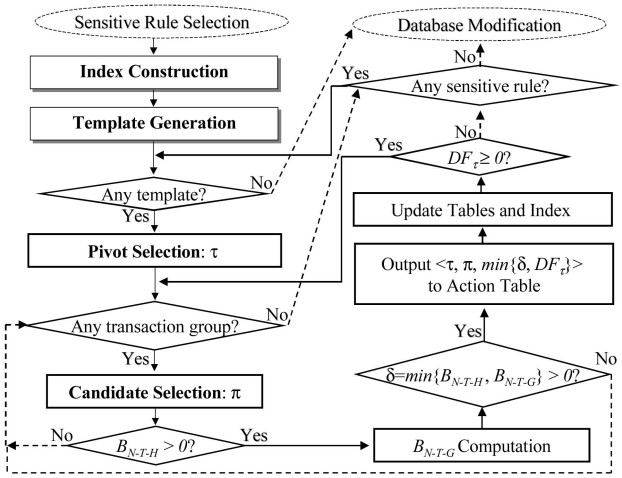


Fig. 5. The flowchart of our approach.

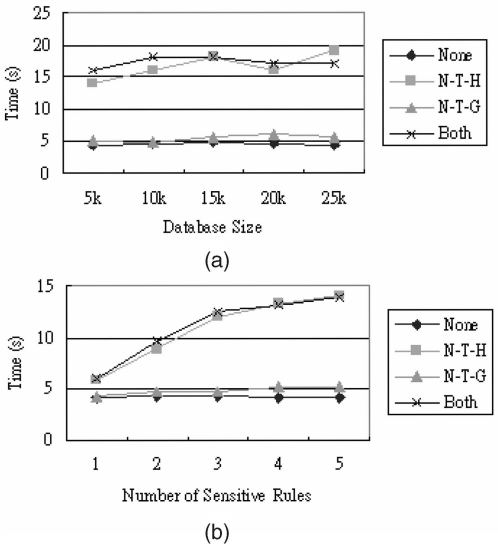


Fig. 6. (a) Scalability on the database size and (b) the number of sensitive rules.

effectiveness. One is the number of modified entries, indicating how much the content of the original database is preserved. The other measures are defined as follows: Let H and U be the sets of all sensitive rules and all strong rules in the original database, respectively. After rule hiding, let the set of all strong rules in the modified database be denoted as U'. Moreover, let SR, LR, and FR, respectively, denote the sets of all the sensitive rules that fail to be hidden, all the loss rules, and all the false rules. The number of rules in any notation R is denoted as $|R|$. By Definition 2.4, we adopt three measures to evaluate the performance of our approach on sensitive rule hiding under constraints:

- $FTH\ ratio = |SR|/|H|$: the percentage of the sensitive rules that fail to be hidden.
- $NTH\ ratio = |LR|/|U - H|$: the percentage of the nonsensitive rules falsely hidden.
- $NTG\ ratio = |FR|/|U'|$: the percentage of new strong rules that were originally spurious.

For all the ratios, the lower they are, the better our approach performs. We first evaluate the scalability of our approach in terms of the database size, the number of sensitive rules, and the number of strong rules, respectively. After that, the sensitive rules are selected in such a way that all of them have at least one item in common to evaluate the effectiveness of our approach on the four measures. Finally, we define the overlapping degree of a rule and make experiments to observe how the correlation among rules influences the performance.

4.1 Scalability of our Approach

The processing time reported includes the CPU time consumed in the preprocessing steps (after sensitive rules have been selected), the template generation, and the complete process for hiding sensitive rules. We exclude the I/O time spent on the index construction and the database modification in order to highlight the impact of the database scale on our indexing mechanisms and the proposed method for rule hiding.

The results of the CPU time under varied database sizes are plotted in Fig. 6a. Each of the variations is scalable in terms of the database size. In our implementation, the table keeping the template information can be fully loaded into the main memory. The indexing mechanisms and the prime-number representation are the major reasons for the good scalability of our approach. The former supports fast data access with hashing techniques. Moreover, the frequent items are mapped to the prime numbers in a reverse order of their frequencies. In this way, the product of prime numbers for representing a frequent itemset will not be too large. In addition, the results plotted in Fig. 6b indicate that all the variations are scalable in terms of the number of sensitive rules.

To observe the impact of the number of nonsensitive rules, the results on the CPU time of our approach under varied MCT are plotted in Fig. 7a. Since more nonsensitive rules lead to more costs on checking the constraint $N\text{-}T\text{-}H$, the two variations $N\text{-}T\text{-}H$ and *Both* perform worse than the others. In addition, since the possibility to generate spurious rules grows as the decrease of MCT, the difference between $N\text{-}T\text{-}H$ and *Both* is more significant at a lower MCT. In Fig. 7b, we also show how the number of strong rules varies as the growth of MCT.

4.2 Sensitive Rules with Common Items

In this experiment, we divide the strong rules into clusters by the common item in them. For example, given four rules $a \rightarrow b$, $b \rightarrow a$, $c \rightarrow d$, and $c \rightarrow e$, there can be the five clusters in Table 10. We randomly select one cluster with exactly five rules and then alternately select one, two, three, four, and five rules from it as sensitive rules. In this way, there must be at least one solution to hide all the sensitive rules because they have at least one item in common. The figure of the CPU time is omitted since there is no obvious trend among the results of the four variations.

Fig. 8a shows the number of modified entries for the four variations with respect to the number of sensitive rules. It can be seen that each curve climbs up and then down from one extreme case (only one sensitive rule) to the other (the entire cluster). It is reasonable because very few modifications are required if all the sensitive rules have a common item. In Fig. 8b, the variations $N\text{-}T\text{-}G$ and *None* always hide

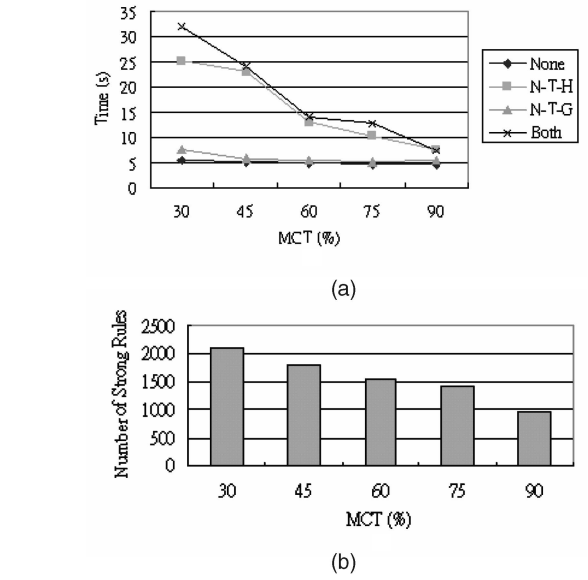


Fig. 7. Scalability on the number of strong rules.

all the sensitive rules. Moreover, the variation *Both* can hide all the sensitive rules only when the entire cluster of rules is regarded as sensitive. In Fig. 8c and Fig. 8d, all the variations achieve very low NTH ratio and NTG ratio. Moreover, the curves of *Both* in both figures also verify that our approach can avoid producing side effects.

4.3 Sensitive Rules with High Overlap

Definition 4.1: Overlapping degree. *The* occurrence *of an item means the number of strong rules that have it in the precedent or the consequent. We define the* overlapping degree *of a rule r as follows:*

$$\log_{\omega}\left(\sum_{\text{item}\in r}\text{occurence}_{\text{item}}\right).$$

Take the rule $a \rightarrow b$ in Table 10 as an example. Its overlapping degree is equal to two if we set ω as 2. It is more difficult to hide the rule with a higher overlapping degree because there can be more rules affected in the rule hiding process. In this experiment, we set ω to 5 and alternately select the rules whose overlapping degrees are one, two, three, four, and five as sensitive ones. In this way, we can observe how the correlation among sensitive rules influences the performance. The figure of the number of modified entries is omitted since there is no obvious trend among the results of the four variations. Fig. 9 shows the results on the other measures.

Fig. 9a and Fig. 9b both show that the curves of *N-T-H* and Both have the upward trend as the growth of overlapping degree. In Fig. 9b, the points where some sensitive rules are not hidden mean that the correlation among the rules has a great impact on the FTH ratio, especially for the constraint *N-T-H*. In Fig. 9c and Fig. 9d, it can be observed that the side effects become worse as the overlapping degree increases. The curves of Both in both figures verify again that our approach avoids producing the side effects. Moreover, from the curves of *N-T-G* in Fig. 8b and Fig. 9b, we find that in most cases, all the sensitive rules can be hidden without generating any false rule.

TABLE 10
An Example of Rule Clustering

Common Item	*a*	*b*	*c*	*d*	*e*
Rule Cluster	*a→b*, *b→a*	*a→b*, *b→a*	*c→d*, *c→e*	*c→d*	*c→e*

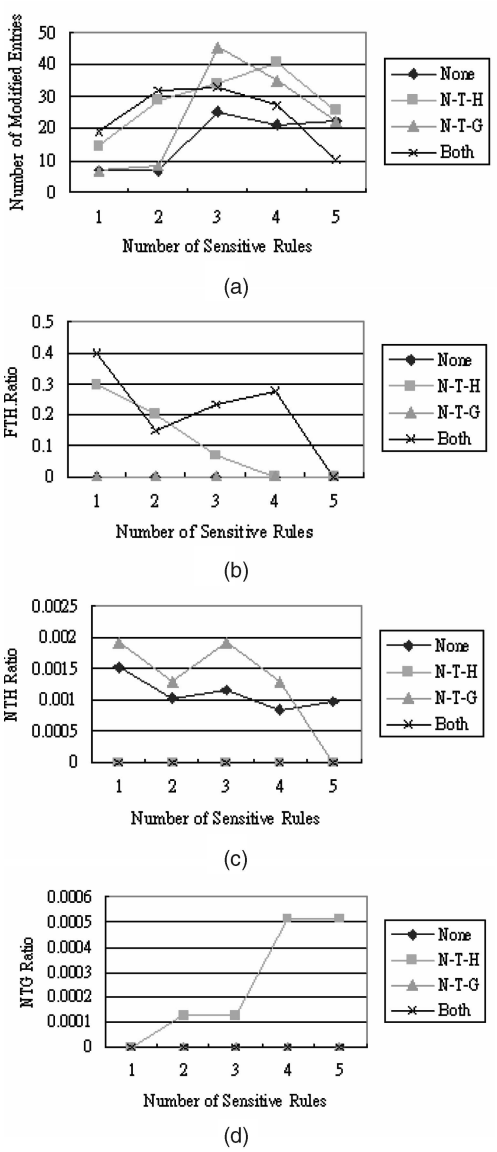


Fig. 8. Effectiveness on the number of sensitive rules.

5 Related Work

Sensitive rule hiding is a subfield of *privacy preserving data mining*, which can be divided into two categories. One is the preserving of *data privacy*, which considers all or parts of the data to be sensitive [1], [5], [11], [14], [20]. Its goal is to blur the sensitive data but keep the summary information unchanged. The other is the preserving of *information privacy*, assuming that only the summary information is sensitive. Its goal is to hide the sensitive information but

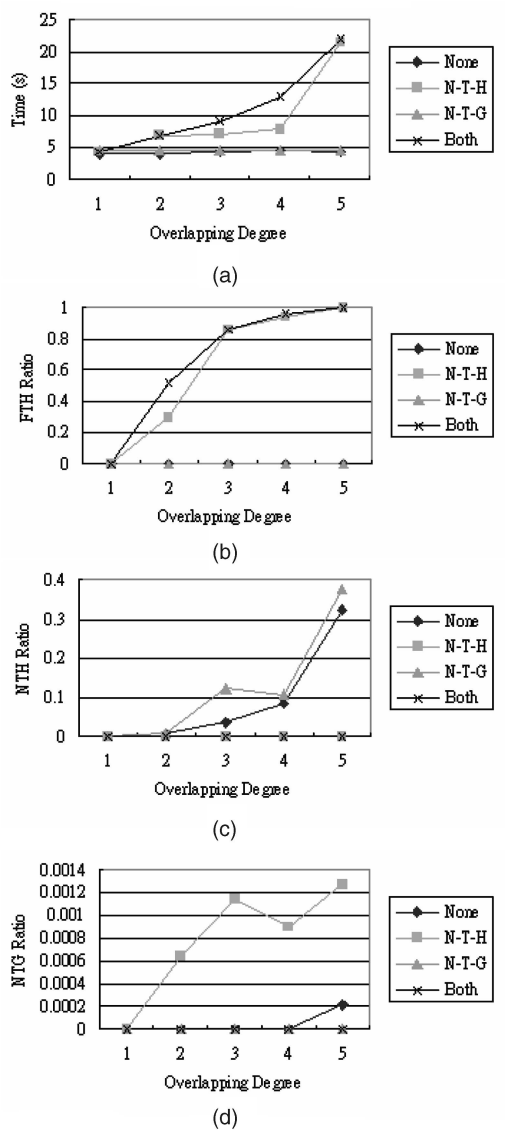


Fig. 9. Performance on the overlapping degree.

retain most of the original data. Sensitive rule hiding belongs to the second category.

For data privacy, Evfimievski et al. [12] introduce a way to measure the degree of privacy when the database contents are blurred. They also present randomization operators yielding a high degree of data privacy and formulae estimating the itemset supports on the randomized database. Rizvi and Haritsa [19] randomize the database by the *Bernoulli* function such that the itemset supports in the original database can be recovered with a high degree of accuracy. Vaidya and Clifton [23] describe a particular scenario, where the items of a database are distributed over different sites. They design an algorithm to find all the frequent itemsets without revealing the contents of the individual transactions. Kantarcioglu and Clifton [15] study another scenario, where the transactions are divided into partitions. They employ the cryptographic techniques to minimize the amount of shared information for mining frequent itemsets in a distributed environment. All these works do not take into account the case where the mining results themselves are confidential.

For information privacy, Clifton and Marks [9] give the example mentioned in Section 1 and discuss the related topics, including the sensitive rule hiding. Most of the previous works adopt either *sampling* or *modifying* approaches. A sampling approach [8] makes a sample database such that the sensitive rules cannot be uncovered, while a modifying approach alters a few parts of the database to decrease the supports or confidences of the sensitive rules. In this paper, we adopt the modifying approach since it can retain the original data as much as possible. More discussions on privacy preserving data mining can be found in a survey paper [25]. In the following, we focus the discussions on sensitive rule hiding.

Atallah et al. [6] refer to sensitive rule hiding as *data sanitization*, which aims at hiding a set of sensitive itemsets, and prove that optimal sanitization is NP-Hard. Moreover, a heuristic approach is proposed using an *itemset graph* to hide sensitive itemsets in a one-by-one fashion. Oliveira and Zaïne [18] further address the efficiency and effectiveness issues. They *plug* a transaction retrieval engine to the hiding process and achieve a linear scalability in terms of database size. On effectiveness, they introduce three measures for *hiding failures* (sensitive patterns that are not hidden), *missing costs* (nonsensitive patterns falsely hidden), and *artifactual patterns* (spurious patterns falsely generated). Since these approaches only consider the decrease of supports, they may fail to hide a rule if the rule can be hidden only by decreasing the confidence.

Saygin et al. [21], [22] argue that both the insertion and deletion of items will introduce false information and make it hard to determine whether the rules derived from the modified database can be trusted. Therefore, they propose a modification scheme to replace entries with *unknowns*. With this scheme, the itemset support is represented as a range from the *minimum support* (ignore unknowns) to the *maximum support* (regard unknowns as normal entries). Similarly, the confidence of a rule, say $X \rightarrow Y$, is also represented as a range from the *minimum confidence* (the minimum support of $X \cup Y$ divided by the maximum support of X) to the *maximum confidence* (the maximum support of $X \cup Y$ divided by the minimum support of X). By definition, a rule $X \rightarrow Y$ is hidden if its minimum confidence is below MCT or the minimum support of $X \cup Y$ is below MST. From our view, the uncertainty due to the ranges on supports and confidences makes it hard to determine whether the derived rules can be trusted. Moreover, the side effects will be out of control since they do not consider the correlation among rules in their modification scheme.

The work proposed by Verykios et al. in [24] should be the most relevant to our work. In that work, the authors propose five algorithms for rule hiding, which are also based on the decrease of supports and confidences. Unlike us, they make a strong assumption—only the rules that are supported by disjoint frequent itemsets can be selected as sensitive. With this assumption, their algorithms can hide one rule at a time and decrease supports or confidences one unit at a time. Moreover, since the authors aim at hiding all sensitive rules instead of avoiding side effects, they do not consider the correlation among rules in their algorithms. Finally, to hide a rule, there can be a large number of "candidate" entries to modify. The "minimum impact" criterion used in the five algorithms only focuses on minimizing the number of modified entries. By contrast,

in this paper, we drop the assumption and decide the modification schemes and the entries to modify based on the correlation among three kinds of rules (sensitive, nonsensitive, and spurious).

6 Conclusion

In this paper, we present a novel approach that modifies the database to hide sensitive rules with limited side effects. We propose to classify all the valid modifications such that every class of modifications is related with the sensitive rules, nonsensitive rules, and spurious rules that can be affected after the modifications. With the methods proposed in this paper, we can modify the transactions in an order so that both the numbers of hidden sensitive rules and modified entries are considered. The experimental results show that our approach is scalable in terms of database size. Moreover, our approach to the avoidance of undesired side effects in rule hiding is effective in two well-designed experiments. In most cases, all the sensitive rules are hidden without false rules generated. In addition, it is observed that the common items and the overlapping degrees among sensitive rules have a great impact on the performance of rule hiding.

It can be interesting to discover the full set of rules that will be falsely hidden or generated as the side effects after rule hiding. Efficient mechanisms are required to speed up the rule hiding process for large databases. Another issue is the fast recognition of sensitive rules that cannot be hidden according to the user-specified constraint. An ideal goal is to build a system that can aid the database administrator to find the sensitive rules for hiding. The other issue is to remove the threshold assumption. A rule hiding approach should be robust no matter how the adversary looks into the modified database, e.g., using a lower MCT to reveal the hidden sensitive rules. The challenge is to take into account both the above attacks and the undesired side effects.

Appendix

In some cases, a sensitive rule cannot be safely hidden due to the correlation among rules. The following are some properties for such cases.

Property 8. *Given a sensitive rule $X \rightarrow Y$ and a nonsensitive rule $Y \rightarrow X$, $X \rightarrow Y$ cannot be hidden without hiding $Y \rightarrow X$ by applying Scheme 1 to $\Sigma_{X \cup Y}$ if one of the conditions holds:*

1. $MC_{X \rightarrow Y} = MSC_{X \rightarrow Y}$ *and* $MSC_{X \rightarrow Y} = MSC_{Y \rightarrow X}$.
2. $MC_{X \rightarrow Y} = MCCC_{X \rightarrow Y}$ *and*
$$MCCC_{X \rightarrow Y} \geq MPCC_{Y \rightarrow X}.$$
3. $MC_{X \rightarrow Y} = MPCC_{X \rightarrow Y}$ *and*
$$MPCC_{X \rightarrow Y} \geq MCCC_{Y \rightarrow X}.$$

Proof. From Definition 2.5, we have

$$MC_{X \rightarrow Y} = min\{MSC_{X \rightarrow Y}, MCCC_{X \rightarrow Y}, MPCC_{X \rightarrow Y}\}.$$

1. If $X \rightarrow Y$ is hidden, since $MC_{X \rightarrow Y} = MSC_{X \rightarrow Y} \Rightarrow$ there are $MSC_{X \rightarrow Y}$ transactions in $\Sigma_{X \cup Y}$ in which an item in $X \cup Y$ is deleted $\Rightarrow Y \rightarrow X$ is also hidden because $MSC_{Y \rightarrow X} = MSC_{X \rightarrow Y}$.
2. If $X \rightarrow Y$ is hidden, since
$$MC_{X \rightarrow Y} = MCCC_{X \rightarrow Y} \Rightarrow,$$
there are $MCCC_{X \rightarrow Y}$ transactions in $\Sigma_{X \cup Y}$ in which an item in Y is deleted $\Rightarrow Y \rightarrow X$ is also hidden because $MPCC_{Y \rightarrow X} \leq MCCC_{X \rightarrow Y}$.
3. Similarly, if $X \rightarrow Y$ is hidden, since $MC_{X \rightarrow Y} = MPCC_{X \rightarrow Y} \Rightarrow$ there are $MPCC_{X \rightarrow Y}$ transactions in $\Sigma_{X \cup Y}$ in which an item in X is deleted $\Rightarrow Y \rightarrow X$ is also hidden because
$$MCCC_{Y \rightarrow X} \leq MPCC_{X \rightarrow Y}.$$

□

Property 9. *Given a sensitive rule $X \rightarrow Y$ and a nonsensitive rule $X \rightarrow Z$,* let $\Sigma_{X-(Y \cup Z)}$ *be the set of all transactions that contain X* but do not contain *$Y \cup Z$. $X \rightarrow Y$ cannot be hidden without hiding $X \rightarrow Z$ by applying Scheme 2 to introduce new transactions in $\Sigma_{X-(Y \cup Z)}$ if $PC_{X \rightarrow Y} \geq PC_{X \rightarrow Z}$.*

Proof. If $X \rightarrow Y$ is hidden, by Definition 2.6, there are $PC_{X \rightarrow Y}$ transactions added into $\Sigma_{X-(Y \cup Z)} \Rightarrow X \rightarrow Z$ and it is also hidden because $PC_{X \rightarrow Z} \leq PC_{X \rightarrow Y}$. □

Property 10. *Given a sensitive rule $X \rightarrow Y$ and a spurious rule $Y \rightarrow Z$, let $\Sigma_{(X \cup Y)-Z}$ be the set of all transactions that contain $X \cup Y$ but do not contain Z. $X \rightarrow Y$ cannot be hidden without generating $Y \rightarrow Z$ by applying Scheme 1 to $\Sigma_{(X \cup Y)-Z}$ if $MC_{X \rightarrow Y} = MCCC_{X \rightarrow Y}$ and $MCCC_{X \rightarrow Y} \geq RCC_{Y \rightarrow Z}$.*

Proof. If $X \rightarrow Y$ is hidden, by Definition 2.5 $\Rightarrow$ there are $MCCC_{X \rightarrow Y}$ transactions in $\Sigma_{(X \cup Y)-Z}$ in which an item in Y is deleted.

$\Rightarrow Y \rightarrow Z$ is also generated because $RCC_{Y \rightarrow Z} \leq MCCC_{X \rightarrow Y}$. □

Property 11. *Given a sensitive rule $X \rightarrow Y$ and a spurious rule $X \rightarrow Z$, let $\Sigma_{(X \cup Z)-Y}$ be the set of all transactions that contain $X \cup Z$ but do not contain Y. $X \rightarrow Y$ cannot be hidden without generating $X \rightarrow Z$ by applying Scheme 2 to $\Sigma_{(X \cup Z)-Y}$ if $PC_{X \rightarrow Y} \geq AC_{X \rightarrow Z}$.*

Proof. If $X \rightarrow Y$ is hidden, by Definition 2.6, there are $PC_{X \rightarrow Y}$ transactions added into $\Sigma_{(X \cup Z)-Y} \Rightarrow X \rightarrow Z$ and it is also generated because $AC_{X \rightarrow Z} \leq PC_{X \rightarrow Y}$. □

References

[1] D. Agrawal and C.C. Aggarwal, "On the Design and Quantification of Privacy Preserving Data Mining Algorithms," *Proc. ACM Symp. Principles of Database Systems,* pp. 247-255, 2001.

[2] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," *Proc. ACM Conf. Management of Data,* pp. 207-216, 1993.

[3] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo, "Fast Discovery of Association Rules," *Advances in Knowledge Discovery and Data Mining,* chapter 12, U.M. Fayyad et al., eds., AAAI/MIT Press, pp. 307-328, 1996.

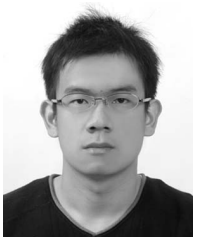
[4] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. Conf. Very Large Data Bases,* pp. 487-499, 1994.

[5] R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining," *Proc. ACM Conf. Management of Data,* pp. 14-19, 2000.

[6] M. Atallah et al., "Disclosure Limitation of Sensitive Rules," *Proc. IEEE Workshop Knowledge and Data Eng. Exchange,* pp. 45-52, 1999.

[7] C.M. Chiang, "A New Approach for Sensitive Rule Hiding by Considering Side Effects," master thesis, Dept. of Computer Science, Nat'l Tsing Hua Univ., Republic of China, 2003.
[8] C. Clifton, "Protecting against Data Mining through Samples," *Proc. IFIP Conf. Database Security*, pp. 193-207, 1999.
[9] C. Clifton and D. Marks, "Security and Privacy Implications of Data Mining," *Proc. ACM Workshop Research Issues in Data Mining and Knowledge Discovery*, 1996.
[10] E. Dasseni, V.S. Verykios, A.K. Elmagarmid, and E. Bertino, "Hiding Association Rules by Using Confidence and Support," *Proc. Information Hiding Workshop*, pp. 369-383, 2001.
[11] V. Estivill-Castro and L. Brankovic, "Data Swapping: Balancing Privacy against Precision in Mining for Logic Rules," *Proc. Conf. Data Warehousing and Knowledge Discovery*, pp. 389-398, 1999.
[12] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy Preserving Mining of Association Rules," *Information Systems*, vol. 29, pp. 343-364, 2004.
[13] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," *Data Mining and Knowledge Discovery*, vol. 8, no. 1, pp. 53-87, 2004.
[14] V.S. Iyengar, "Transforming Data to Satisfy Privacy Constraints," *Proc. ACM Conf. Knowledge Discovery and Data Mining*, pp. 279-288, 2002.
[15] M. Kantarcioglu and C. Clifton, "Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data," *Proc. ACM Workshop Research Issues in Data Mining and Knowledge Discovery*, 2002.
[16] *Information Storage and Retrieval Systems Theory and Implementation*, G.J. Kowalski and M.T. Maybury, eds. Kluwer Academic Publishers, 1997.
[17] J. Liu, Y. Pan, K. Wang, and J. Han, "Mining Frequent Item Sets by Opportunistic Projection," *Proc. ACM Conf. Knowledge Discovery and Data Mining*, 2002.
[18] S.R.M. Oliveira and O.R. Zaïne, "Privacy Preserving Frequent Itemset Mining," *Proc. IEEE ICDM Workshop Privacy, Security, and Data Mining*, pp. 43-54, 2002.
[19] S.J. Rizvi and J.R. Haritsa, "Maintaining Data Privacy in Association Rule Mining," *Proc. Conf. Very Large Data Bases*, pp. 682-693, 2002.
[20] P. Samarati, "Protecting Respondents' Identities in Microdata Release," *IEEE Trans. Knowledge and Data Eng.*, vol. 13, no. 6, pp. 1010-1027, 2001.
[21] 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.
[22] Y. Saygin, V.S. Verykios, and A.K. Elmagarmid, "Privacy Preserving Association Rule Mining," *Proc. IEEE Workshop Research Issues in Data Eng.*, 2002.
[23] J. Vaidya and C. Clifton, "Privacy Preserving Association Rule Mining in Vertically Partitioned Data," *Proc. ACM Conf. Knowledge Discovery and Data Mining*, 2002.
[24] V.S. Verykios, A.K. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, "Association Rule Hiding," *IEEE Trans. Knowledge and Data Eng.*, vol. 16, no. 4, pp. 434-447, 2004.
[25] V.S. Verykios, E. Bertino, I.N. Fovino, L.P. Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-Art in Privacy Preserving Data Mining," *ACM SIGMOD Record*, vol. 33, no. 1, 2004.
[26] S.J. Yen and A.L.P. Chen, "A Graph-Based Approach for Discovering Various Types of Association Rules," *IEEE Trans. Knowledge and Data Eng.*, vol. 13, no. 5, 2001.
[27] M.J. Zaki, "Scalable Algorithms for Association Mining," *IEEE Trans. Knowledge and Data Eng.*, vol. 12, no. 3, pp. 372-390, 2000.
[28] http://www.almaden.ibm.com/software/quest/Resources/index.shtml, 2003.

Yi-Hung Wu received the PhD degree in computer science from National Tsing Hua University, Taiwan, in 2001. He worked at the Computer & Communication Research Center at National Tsing Hua University, Taiwan, as a postdoctoral fellow, from 2001 to 2006. In 2006, he joined the Department of Information and Computer Engineering, Chung Yuan Christian University, Taiwan, as an assistant professor. His current research interests are in the area of data mining, with emphasis on data stream analysis techniques and applications. He also works on the problems related to Web data mining, multimedia data mining, and privacy-preserving data mining.



Chia-Ming Chiang received the BS degree in computer science from National Dong Hwa University, Taiwan, in 2001, and a MS degree in computer science from National Tsing Hua University, Taiwan, in 2003. He works at VIA Communications, Inc. as a section chief of the System Application Software Department now. His previous research interests are focused on data analysis and knowledge engineering. He works on system software architecture design and framework development now.

Arbee L.P. Chen received the PhD degree in computer engineering from the University of Southern California, Los Angeles, in 1984. Dr. Chen is currently Dean of the College of Science and Chengda Chair Professor at National Chengchi University, Taiwan. He also holds a joint professorship at National Tsing Hua University where he has been a professor since 1990. Before returning to Taiwan in 1990, Dr. Chen was a member of technical staff at Bell Communications Research, New Jersey, from 1987 to 1990, an adjunct associate professor in the Department of Electrical Engineering and Computer Science, Polytechnic University, New York, and a research scientist at Unisys, California, from 1985 to 1986. His current research interests include multimedia databases, data mining, and data stream management systems. Dr. Chen organized the 1995 IEEE Data Engineering Conference, the 1999 International Conference on Database Systems for Advanced Applications, and the 2002 International Computer Symposium in Taiwan. He will be a program committee cochair of the 2008 IEEE Data Engineering Conference to be held in Mexico. He was invited to deliver a speech at the US National Science Foundation-sponsored Inaugural International Symposium on Music Information Retrieval at Plymouth, USA, 2000, and in the IEEE Shannon Lecture Series at Stanford University, California 2005. Dr. Chen was a visiting scholar at Tsinghua University, China, 1999, Kyoto University, Japan, 1999, Stanford University, 2003-2005, and King's College London, UK, 2005. He has published more than 170 papers in international journals and conference proceedings, and has been a recipient of the National Science Council's Distinguished Research Award since 1996. He is a senior member of the IEEE Computer Society.