Apply Filter for Hidden Sensitive Association Rule with Limited Side Effects

[1]B.Venkateswarlu, [2]Vagesna Anusha
[1,2]Dept. of CSE, Avanthi Institute of Engg. and Tech., Narsipatnam, Visakhapatnam, AP, India

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 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 and applies the filters for redundant rules in hidden sensitive rules . 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 applies filters. 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. The data which has been processed are supports for the better decision making.

Keywords
Association Rules, Data Mining, Mining Methods and Algorithms, Rule Hiding, DSS

I. 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 SupI) 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 SupXUY divided by SupX. We say that $X \rightarrow Y$ holds in the database if X U 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]. However, the misuse of them may bring undesired effects to people. An example mentioned in [9] is as follows [22]. 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:
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
3. No false rule
4. No redundant rule?

Since violating the last two constraints means the production of side effects, focusing on the first constraint like the work in 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 three 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 belongs X U Y from transactions that contain X U Y will decrease both SupXuY and Confr. Moreover, inserting item I belongs to X into transactions that contain X but fig 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, [22]. 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.

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 and apply the filters to reduce the redundant rule. 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.

II. Related Works

The proliferation of information technologies and the internet in the past two decades has brought a wealth of individual information into the hands of commercial companies and government agencies. As hardware costs go down, it easier than ever to keep any piece of information acquired from the ongoing activities of their clients. Data owners constantly seek to make better use of the data they possess, and utilize data mining tools to extract useful knowledge and patterns from the data. In result, there is a growing concern about the ability of data owners, such as large corporations and government agencies, to abuse this knowledge and compromise the privacy of their clients { concern which has been reflected in the actions of legislative bodies (e.g., the debate about and subsequent elimination of the Total Information Awareness project in the US. This concern is exacerbated by actual incidents that demonstrate how difficult it is to use and share information while protecting individuals' privacy. 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 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. For information privacy, Clifton and Marks [9], give the example mentioned [22], discuss the related

topics, including the sensitive rule hiding . 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. 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 Zaı¨ne [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. The work proposed by Verykios et al. in, 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 four kinds of rules (sensitive, nonsensitive, spurious and redundant).

A. Problem Formulation

Let T= $\{i_1,i_2,i_3,\ldots\ldots i_n\}$ and D=$\{t_1,t_2,t_3,\ldots\ldots t_n\}$ where every t_j is a subset of T, 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 $b_1b_2 \ldots b_n$ where, b_k =1

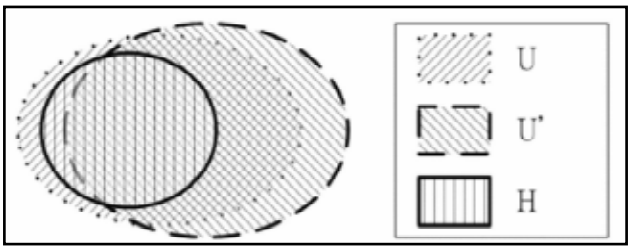


Fig. 1: The relationships among the rule set U, U_0, and H.

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.

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. $Conf_{X\rightarrow Y}(= C_{X\cup Y}/C_X) \geq MCT$.

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*).*

B. 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, Scheme 2: Modify entries from 0s to 1s, Scheme 3: Modify entries from 1s to 0s or from 0s to 1s, Scheme 4: Change 0s and 1s to ?s, Scheme 5: Swap 0 and 1 between two transactions [22].

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 to b. Correlation with the other sensitive rule, say c to 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 to 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 to a.

3. Correlation with a spurious rule, say ab to 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 to c.

In this work we applying filter for to avoid the redundant rules for sensitive rule hiding.

III. Proposed Model

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 III. 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 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.

A. 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 [22].

B. Approach for Rule Hiding

1. Classification of Modifications—Templates

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 U Y can be expressed as [22]. Logical form, Template, Transaction containment, Sensitive rule coverage, Minus count of a template, Difference of a template, Joint template, Template reduction, Transaction group, Sensitive rule conflict, Nonsensitive rule coverage, Minus count of a transaction group, N-T-H bound of a transaction group, Spurious rule coverage, Addible count of a transaction group, N-T-G bound of a transaction group.

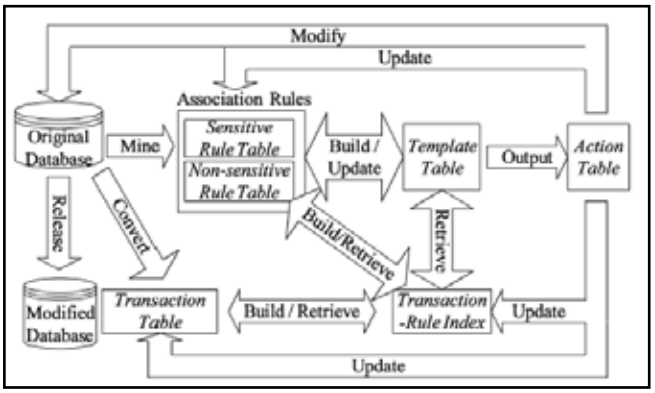


Fig. 2:

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

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, The CV and DF field is computed from Definition 3.4, while the DF field is computed.

We are modifying the previous system by Appling the filters for the sensitive rules to eliminate the redundant rules against sensitive rules hidden in the database.

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 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.

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	$a \wedge c \wedge d \Rightarrow a \wedge c \wedge \neg d$

Table 3: A Transaction Database in the Form of Bit-Vectors

TID		⇒	TID	a	b	c	d	e
1	abe		1	1	1	0	0	1
2	ce		2	0	0	1	0	1
3	ace		3	1	0	1	0	1
4	abde		4	1	1	0	1	1
5	b		5	0	1	0	0	0
6	ab		6	1	1	0	0	0

As depicted in fig. 2, the strong rules are distributed into two tables 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.

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),*

Table 5: The Encoding Table and the Converted Database

Encoding Table		TID	2	3	5	7	Product
a→2	⇒	1	1	1	1	0	*30=2×3×5*
b→3		2	0	0	1	1	*35=5×7*
e→5		3	1	0	1	1	*70=2×5×7*
c→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*

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*

With this index, the transactions and nonsensitive rules satisfying a logical form can be quickly identified. For instance, the transactions that contain item set 6 but do not contain itemset 35. are in the first and the fifth buckets of Table 6. After the index and tables are built, the templates are generated as described in Section III. 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. A complete flowchart of our approach is shown in fig. 4.

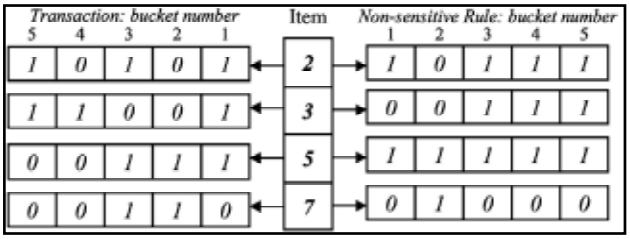


Fig. 3. The Transaction-Rule Index

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	$\neg a \wedge \neg b \wedge \neg e \Rightarrow a \wedge \neg b \wedge \neg e$

IV. Performance Evaluation

We adopt the IBM data generator 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).We apply filters after sensitive rule selection. 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 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 U0. 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 jRj. By Definition 2.4, we adopt three measures to evaluate the performance of our approach on sensitive rule hiding under constraints:[22] 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.

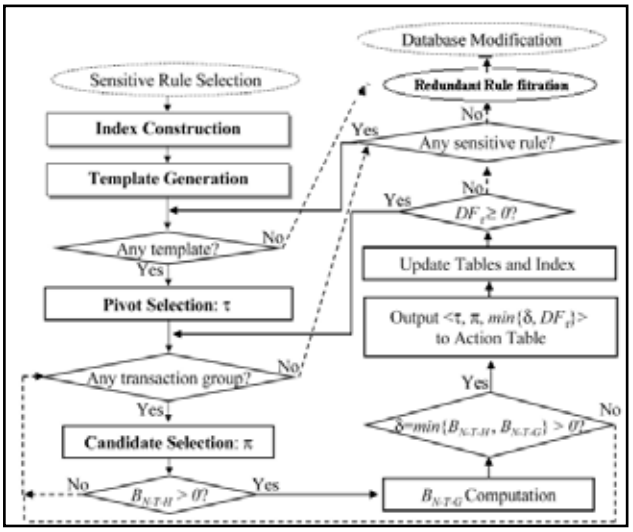


Fig .4: The Flowchart of Proposed Approach

Sensitive Rules with Common Items, Sensitive Rules with High Overlap [22] are same the previous system.

V. Conclusions

In this paper, we present a novel approach that modifies the database to hide sensitive rules with limited side effects by filtering the redundant rules by comparing against the modified database. 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. Finally the proposed method is effective and efficient to support the decision making purpose.

References

[1] D. Agrawal, C.C. Aggarwal,"On the Design and Association Rules", ACM SIGMOD Record, Vol. 30, No. 4, pp. 45-54, 2001.

[2] Y. Saygin, V.S. Verykios, A.K. Elmagarmid,"Privacy Preserving Association Rule Mining", Proc. IEEE Workshop Research Issues in Data Eng., 2002.

[3] J. Vaidya, C. Clifton,"Privacy Preserving Association Rule Mining in Vertically Partitioned Data", Proc. ACM Conf. Knowledge Discovery and Data Mining, 2002.

[4] V.S. Verykios, A.K. Elmagarmid, E. Bertino, Y. Saygin, E. Dasseni,"Association Rule Hiding", IEEE Trans. Knowledge and Data Eng., Vol. 16, No. 4, pp. 434-447, 2004.

[5] V.S. Verykios, E. Bertino, I.N. Fovino, L.P. Provenza, Y. Saygin, Y. Theodoridis,"State-of-the-Art in Privacy Preserving Data Mining," ACM SIGMOD Record, Vol. 33, No. 1, 2004.

[6] S.J. Yen, 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.

[7] M.J. Zaki,"Scalable Algorithms for Association Mining", IEEE Trans. Knowledge and Data Eng., Vol. 12, No. 3, pp. 372-390, 2000.

[8] [Online Available: http://www.almaden.ibm.com/software/quest/Resources/index.shtml, 2003. "Quantification of Privacy Preserving Data Mining Algorithms", Proc. ACM Symp. Principles of Database Systems, pp. 247-255, 2001.

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

[10] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, 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.

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

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

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

[14] 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.

[15] C. Clifton,"Protecting against Data Mining through Samples", Proc. IFIP Conf. Database Security, pp. 193-207, 1999.

[9] C. Clifton, 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, E. Bertino, "Hiding Association Rules by Using Confidence and Support", Proc. Information Hiding Workshop, pp. 369-383, 2001.

[11] V. Estivill-Castro, 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, J. Gehrke, "Privacy Preserving Mining of Association Rules", Information Systems, Vol. 29, pp. 343-364, 2004.

[13] J. Han, J. Pei, Y. Yin, 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, 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] G.J. Kowalski, M.T. Maybury,"Information Storage and Retrieval Systems Theory and Implementation", eds. Kluwer Academic Publishers, 1997.
[17] J. Liu, Y. Pan, K. Wang, J. Han,"Mining Frequent Item Sets by Opportunistic Projection", Proc. ACM Conf. Knowledge Discovery and Data Mining, 2002.
[18] S.R.M. Oliveira, O.R. Zaı¨ne,"Privacy Preserving Frequent Itemset Mining", Proc. IEEE ICDM Workshop Privacy, Security, and Data Mining, pp. 43-54, 2002.
[19] S.J. Rizvi, 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, C. Clifton,"Using Unknowns to Prevent Discovery of
[22] IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No. 1, January 2007.



Venkateswarlu Bondu received the Master's Degree in Computer Science and Systems Engineering from Andhra University College of Engineering, pursuing Ph.D in Computer Science in Andhra University. He is an Associate Professor in the Department of Compute Science in Avanthi Institute of Engineering and Technology. His research areas of interests are Software Engineering & Data Modeling.



Anusha Vegesna received B.TECH in Computer science and Information Technology from Vignan's Institute Of Information & Technology, Affliated to JNTUK, Pursuing M.TECH in Information Technology from Avanthi institute of Engineering & Technology Affiliated to JNTUK. She is a live student Member of IEEE. Her research areas of interest are Data mining & Information Security.