Coefficient-Based Exact Approach for Frequent Itemset Hiding

Engin Leloglu
Dept. of Computer Engineering
Izmir Institute of Technology
Izmir, TURKEY
enginleloglu@ieee.org

Tolga Ayav
Dept. of Computer Engineering
Izmir Institute of Technology
Izmir, TURKEY
tolgaayav@iyte.edu.tr

Belgin Ergenc
Dept. of Computer Engineering
Izmir Institute of Technology
Izmir, TURKEY
belginergenc@iyte.edu.tr

***Abstract*—Concealing sensitive relationships before sharing a database is of utmost importance in many circumstances. This implies to hide the frequent itemsets corresponding to sensitive association rules by removing some items of the database. Research efforts generally aim at finding out more effective methods in terms of convenience, execution time and side-effect. This paper presents a practical approach for hiding sensitive patterns while allowing as much nonsensitive patterns as possible in the sanitized database. We model the itemset hiding problem as integer programming whereas the objective coefficients allow finding out a solution with minimum loss of nonsensitive itemsets. We evaluate our method using three real datasets and compared the results with a previous work. The results show that information loss is dramatically minimized without sacrificing the accuracy.**

Keywords—frequent itemset hiding; exact approach; information loss

I. Introduction

Progresses in the technology give an opportunity to establish transactional databases that can reserve large volumes of data. Analyzing data and extracting meaningful information from these huge piles of data come up as a result of these advances. Data mining field has efficient techniques for this knowledge discovery process. However, improper use of these techniques caused a rise of privacy concerns. Unauthorized access to not only sensitive personal information that is stored or inferred from the data, but also commercial information that provides remarkable benefit over rivals induces privacy issues. That is why comprehensive sanitization on databases is required when the information or data from these databases is shared or published.

Sharing databases allows researchers and policy-makers to examine the data and gain significant information benefiting the society as a whole, such as the strength of a medicine or treatment, social-economic inferences that can be the guide on the road to efficient public policies, and the factors that cause vital diseases. In other words, publishing databases eventuates in utility gain for the society as a whole [13]. However, due to privacy concerns, a privacy preserving method is needed to be applied on the databases. These methods make data imprecise and/or distorted so that no sensitive knowledge is disclosed. But, this distortion causes unwanted information loss and losses in potential utility gain.

Frequent itemset hiding is one of the important and widely used methods of privacy preserving data mining field. There are several frequent itemset hiding algorithms of which methodology can be classified, such as heuristic [6], border-based [16, 17] and exact [8, 10, 11, 14]. They aim to impose small deviation in the original database to expose no sensitive itemsets. This deviation is tried to be minimized by various techniques with different quality metrics as a common feature of these algorithms. One determines the relative frequency of remaining itemsets [16] as a quality parameter while another approach uses the term of accuracy [14] that shows the impact of sanitization on transactions of the database. In addition to this, the information loss which is to conceal nonsensitive itemsets on the original database while hiding sensitive knowledge is another critical point of the hiding process [16]. Studies generally concentrate on achieving the result database which has no sensitive knowledge with small deviation and minimum information loss.

Exact approaches produce more accurate solution than other types of approaches in frequent itemset hiding. However, they are impractical when the number of itemsets and length of itemsets increase. In addition to this, they mostly focus on minimizing deviation in terms of accuracy or distance. To our knowledge there is no practical solution providing frequent itemset hiding with the objective of minimum information loss and accuracy. In this paper, we propose an exact approach for frequent itemset hiding where all sensitive patterns are concealed. Our approach is based on the combination of integer programming and heuristic sanitization. While it prevents revealing sensitive information on published database, minimum information loss and maximum accuracy are also provided.

Our approach proposes the use of coefficients in the objective function of integer programming to minimize information loss. These coefficients reminding the approaches used by utility-based mining algorithms [19] are pre-computed such that they give a measure of information loss. Integer programming allows finding the optimum solution deciding about the transactions to be sanitized. Then, heuristic sanitization algorithm is executed to remove the sensitive itemsets. The experiments with real datasets demonstrate the efficacy of our approach and give useful insight into the efforts of minimizing nonsensitive information loss.

Copyright (c) IARIA, 2014. ISBN: 978-1-61208-329-2

TABLE I. EXAMPLE DATABASE $\mathcal{D}$

Id	Items
T_1	1 2 3 7 8 10
T_2	3 9 10
T_3	4 5 6
T_4	1 2 3 6 7 8 9
T_5	1 2 3 6 7
T_6	10
T_7	4
T_8	3 6 7 8 9
T_9	3 8 9
T_{10}	5 6 7

TABLE II. FREQUENT (NONSINGLETON) ITEMSETS FOR $\mathcal{D}$ AT $\sigma_{min} = 2$

Itemsets	σ_j	Itemsets	σ_j	Itemsets	σ_j	Itemsets	σ_j
5, 6	2	9, 6	2	2, 8, 7	2	1, 2, 8, 7	2
1, 2	3	9, 7	2	2, 8, 3	2	1, 2, 8, 3	2
1, 8	2	9, 3	4	2, 6, 7	2	1, 2, 6, 7	2
1, 6	2	r_3 →**6, 7**	**4**	2, 6, 3	2	1, 2, 6, 3	2
1, 7	3	6, 3	3	2, 7, 3	3	1, 2, 7, 3	3
1, 3	3	7, 3	4	8, 9, 6	2	1, 8, 7, 3	2
2, 8	2	1, 2, 8	2	8, 9, 7	2	1, 6, 7, 3	2
2, 6	2	1, 2, 6	2	8, 9, 3	3	2, 8, 7, 3	2
2, 7	3	1, 2, 7	3	8, 6, 7	2	2, 6, 7, 3	2
2, 3	3	r_4 →**1, 2, 3**	**3**	8, 6, 3	2	8, 9, 6, 7	2
10, 3	2	1, 8, 7	2	8, 7, 3	3	8, 9, 6, 3	2
r_1 →**8, 9**	**3**	1, 8, 3	2	9, 6, 7	2	8, 9, 7, 3	2
8, 6	2	1, 6, 7	2	9, 6, 3	2	8, 6, 7, 3	2
8, 7	3	1, 6, 3	2	9, 7, 3	2	9, 6, 7, 3	2
r_2 →**8, 3**	**4**	1, 7, 3	3	6, 7, 3	3	1, 2, 8, 7, 3	2
						1, 2, 6, 7, 3	2
						8, 9, 6, 7, 3	2

The following sections are organized as follows: Section 2 gives the background of the problem with terms, concepts and considerations. Section 3 presents our approach in detail. Section 4 is an overview of the leading studies about privacy preserving data mining. Section 5 shows the results and evaluations of the experiments to prove the effectiveness of the technique we propose. Finally, we conclude in Section 6.

II. BACKGROUND

Let $\mathcal{F}$ be a set of items. An itemset is a subset of $\mathcal{F}$ and any transaction defined over $\mathcal{F}$ is tuple $< k, \mathcal{T}_k >$, where k is the transaction id and $\mathcal{T}_k$ is the itemset. A transaction $< k, \mathcal{T}_k >$ is said to contain an itemset X iff $\mathcal{T}_k \supseteq X$. A database $\mathcal{D}$ is a set of transactions. Given a database $\mathcal{D}$, the support of an itemset $\mathcal{T}_k$ in the database $\mathcal{D}$ is denoted as *the support* $\sigma(\mathcal{T}_k, \mathcal{D})$. $\sigma(\mathcal{T}_k, \mathcal{D})$ can be represented simply as σ_k for notational convenience. For a given threshold σ_{min}, $\mathcal{T}_k$ is said to be *frequent* if $\sigma(\mathcal{T}_k, \mathcal{D}) \geq \sigma_{min}$. The set of frequent itemsets $\mathcal{F}(\sigma_{min})$ at minimum support level σ_{min} is the set of all itemsets with a minimum support of σ_{min}.

$\mathcal{F}^R(\sigma_{min}) \subseteq \mathcal{F}(\sigma_{min})$ is a group of restrictive patterns that the owner of the data would like to conceal while publishing. A transaction that supports any of these patterns is said to be *sanitized* if any alteration is made on it in such a way that it no longer supports any itemset in $\mathcal{F}^R(\sigma_{min})$. This sanitization implies reducing the support for every $j \in \mathcal{F}^R(\sigma_{min})$ below σ_{min} and concealing itemset j.

In the process of transforming a database $\mathcal{D}$ to a sanitized $\mathcal{D}'$, we have the following considerations:

1) Suppose that $\mathcal{F}'(\sigma_{min})$ be the set of frequent itemsets in the sanitized $\mathcal{D}'$. Any $j \in \mathcal{F}^R(\sigma_{min})$ in $\mathcal{D}$ should not be in $\mathcal{F}'(\sigma_{min})$. In other words, it is aimed that no sensitive knowledge is involved in the sanitized database.

2) The accuracy, which is the ratio of the number of transactions that are not sanitized and the total number of transactions in the database $\mathcal{D}$, should be maximized by keeping the number of sanitized transactions at minimum.

3) Suppose that $\mathcal{F}^n(\sigma_{min})$ be the set of non-sensitive frequent itemsets determined by $\mathcal{F}(\sigma_{min})/\mathcal{F}^R(\sigma_{min})$ in database $\mathcal{D}$. $|\mathcal{F}^n(\sigma_{min})-\mathcal{F}'(\sigma_{min})|$ should be minimized to avoid overconcealing nonsensitive frequent itemsets and keeping the information loss at minimum.

Table 1 represents a database $\mathcal{D}$ which includes 10 transactions and 10 items. Nonsingleton frequent itemsets with the support values bigger than or equal to 2 are listed in Table 2. For example, we assume that the sensitive patterns are {8, 9}, {8, 3}, {6, 7}, {1, 2, 3} that are bold and represented with r_1, r_2, r_3 and r_4. Although, it is possible to define different support thresholds for each sensitive pattern, we assume that the support threshold $\sigma_{min} = 2$ for all patterns, which is practical and common in many circumstances. At the end of the process, we expect that 28 nonsensitive frequent itemsets that are supersets of the sensitive ones would also get concealed as the process of hiding the sensitive itemsets. The question is how to transform $\mathcal{D}$ into the sanitized database $\mathcal{D}'$ in an effective way such that aforementioned considerations 1, 2 and 3 are maintained.

Depending on the consideration 1, support values of our sensitive patterns in the database $\mathcal{D}$ should be dropped below 2, that is minimum support value. For example, the support value of r_2 is 4 and to satisfy the consideration 1, transactions that include r_2 should be found and at least one of two items in r_2 should be deleted from as many transactions as needed. The proper selection of transactions to be sanitized and the items to be removed is of paramount importance, since the number of sanitized transactions and/or the number of items to be removed should be kept at minimum. Moreover, nonsensitive itemsets that contain one of items, 8 or 3, are in danger of being concealed while the support value of r_2 is decreased.

According to consideration 3, the number of nonsensitive itemsets that are concealed should be at minimum.

III. Coefficient-Based Itemset Hiding

In this section, we introduce a novel method for the itemset hiding problem. We first define the problem using integer programming and then simply augment the objective function with coefficients in order to reduce the information loss. Thus, the method consists of three essential parts: Coefficient Computation, Integer Programming Solution and Heuristic Sanitization.

Modeling the itemset hiding problem with integer programming can be done in several ways. We follow the way of Menon et al. [14] such that the objective achieves the maximum accuracy. Our method alters the objective function such that the binary variables indicating whether a transaction is chosen or not are multiplied by some pre-computed coefficients that reflect the amount of information loss. We compute the coefficients of transactions that support sensitive patterns only.

We first start with creating the constraint matrix by eliminating transactions, which do not support sensitive itemsets from consideration, as shown below:

$$\left(\begin{array}{ccccccc} & t_1 & t_2 & t_3 & t_4 & t_5 & t_6 \\ r_1 & 0 & 1 & 0 & 1 & 1 & 0 \\ r_2 & 1 & 1 & 0 & 1 & 1 & 0 \\ r_3 & 0 & 1 & 1 & 1 & 0 & 1 \\ r_4 & 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right) \qquad (1)$$

6 columns represents respectively: $t_1 \rightarrow T_1, t_2 \rightarrow T_4, t_3 \rightarrow T_5, t_4 \rightarrow T_8, t_5 \rightarrow T_9, t_6 \rightarrow T_{10}$. While t_1 supports sensitive itemsets $r_2$and r_4, t_6 contains only one sensitive itemset that is r_3.

A. *Coefficient Computation*

To minimize the impact on nonsensitive frequent itemsets, coefficient computation is made for each transaction which supports sensitive patterns as the first step. A coefficient of the transaction gives the information about a risk of overconcealing nonsensitive frequent itemsets on this transaction. If the value of coefficient is high, it means that the number of concealed nonsensitive itemsets included in the transaction would be high after the sanitization.

This coefficient computation is typically organized by taking into account that initial utility worth of each nonsensitive itemsets on studied database is the same. Calculating a risk of overconcealing nonsensitive frequent itemsets is made based on this assumption. If some information on the database has different worth of utility based on the area where the shared database is utilized, relevant coefficients can be computed independently of Coefficient Computation Algorithm thereby paying regard to requirements of the area.

```
1:  for transactions $i \in \mathscr{D}$ such that $i$ is to be sanitized
2:      identify all sensitive frequent item sets $\mathscr{F}_i{}^R \in \mathscr{F}^R(\sigma_{min})$ supported by $i$
3:      identify all nonsensitive frequent item sets $\mathscr{F}_i{}^n \in \mathscr{F}^n(\sigma_{min})$ supported by $i$
4:      while $\mathscr{F}_i{}^R \neq \emptyset$
5:          calculate $f_j = |\{k \in \mathscr{F}_i{}^R \mid j \in k\}|$,        $\forall$ items $j$ in $i$
6:          calculate $j^* = argmax_j\{f_j\}$
7:          calculate $g_j = |\{k \in \mathscr{F}_i{}^n \mid j^* \in k\}|$,       $\forall$ items $j$ in $i$
8:          update coefficient $c_i = c_i + g_j$
9:          update $\mathscr{F}_i{}^R = \mathscr{F}_i{}^R \backslash \{k \in \mathscr{F}_i{}^R | j^* \in k\}$
10:     end while
11: end for
```

Fig. 1. Coefficient Computation Algorithm.

A coefficient, which is represented by the Coefficient Computation Algorithm in Figure 1 as “c_i”, is calculated for each transaction that is included in the constraint matrix. For example, we may choose transaction T_1 to explain this calculation (on line 1). First, sensitive frequent itemsets and nonsensitive frequent itemsets are identified for transaction T_1 (on line 2 and 3). The item appearing in the most number of sensitive patterns supported by that transaction is selected from all items in $r_2 \cup r_4$ (Items 1, 2, 3, 8) (on line 5 and 6). The item “3” is selected. Record the number of appearances of the item “3” in the non-sensitive frequent itemsets supported by the transaction (on line 7). For our example, there are 6 appearances of item “3” in the nonsensitive frequent itemsets such as {1, 3}, {2, 3}, {10, 3}, {7, 3}, {1, 7, 3}, {2, 7, 3}. Remove all sensitive itemsets supported by the transaction contain the selected item “3” (on line 9). If sensitive itemsets remain supported by T_1, repeat the procedure (on line 4) and sum the appearances of new selected item in the nonsensitive frequent itemsets supported by the transaction with recorded value (on line 8). There is no sensitive itemset left in our example. Hence, the total summation for transaction T_1 is 6. After, all transactions which are included in the constraint matrix are taken in consideration based on this procedure, a coefficient for each transaction is found such as $T_1 \rightarrow 6$, $T_4 \rightarrow 29$, $T_5 \rightarrow 14$, $T_8 \rightarrow 6$, $T_9 \rightarrow 0$, $T_{10} \rightarrow 1$.

B. *Integer Programming Solution*

In this section, we describe the integer programming formulation to solve Coefficient-Based Itemset Hiding problem. Initally, give a_{ij} a binary value. Be 1 if transaction $i \in \mathscr{D}$supports itemset $j \in \mathscr{F}^R(\sigma_{min})$. Otherwise, the value of a_{ij} is 0. For the variable x_i, it will be set to 1 if transaction $i \in \mathscr{D}$ is sanitized. Otherwise, the value of x_i is 0. σ_j represents the current support for itemset $j \in \mathscr{F}(\sigma_{min})$. Recall that c_i is the coefficient that is calculated in Coefficient Computation for transaction $i \in \mathscr{D}$, which contains at least one sensitive itemset.

In the light of this information, the formulation is generated as below:

$$min \sum_{i \in \mathscr{D}} c_i x_i \,, \tag{2}$$

$$s.t. \sum_{i \in \mathscr{D}} a_{ij}\, x_i \geq \sigma_j - \sigma_{min} + 1 \qquad \forall j \in \mathscr{F}^R(\sigma_{min}), \tag{3}$$

$$x_i \in \{0, 1\} \;\; \forall i \in \mathscr{D}. \tag{4}$$

Equation (2) represents the objective function that minimizes the number of transactions sanitized. Equation (3) includes the constraint that more than $(\sigma_j - \sigma_{min})$ transactions supporting each sensitive itemset have to be sanitized, that's why this line is generated for each sensitive itemset. Equation (4) imposes that x_i has only binary value. The integer programming formulation is reorganized based on the constraint matrix (1) and coefficients that are gained by Coefficient Computation in the previous section as:

$$min\, 6x_1 + 29x_2 + 14x_3 + 6x_4 + 0x_5 + 1x_6, \tag{5}$$

$$s.t. \;\; x_2 + x_4 + x_5 \;\geq\; 2, \tag{6}$$

$$x_1 + x_2 + x_4 + x_5 \;\geq\; 3, \tag{7}$$

$$x_2 + x_3 + x_4 + x_6 \;\geq\; 3, \tag{8}$$

$$x_1 + x_2 + x_3 \;\geq\; 2, \tag{9}$$

$$x_1, x_2, x_3, x_4, x_5, x_6 \;\in \{0, 1\}. \tag{10}$$

Solving this integer program results in optimal solution $x_1 = x_3 = x_4 = x_5 = x_6 = 1$ with the other variables being 0. The accuracy of the resulting sanitized database is 0.50.

C. Heuristic Sanitization

Sanitization is a kind of process that includes removing items from a transaction; thereby the sanitized version of the transaction supports no itemset in $\mathscr{F}^R(\sigma_{min})$. There are various sanitization approaches in the privacy preserving data mining literature. For instance, Verykios et al. offered two sanitization techniques in their study [18]. These are generally based on hiding itemsets that are already sorted with respect to their size and support, in a different fashion such as one-by-one and round-robin. Amiri [1] presented the Aggregate Algorithm based on removing the most sensitive and the least nonsensitive itemsets in selected transaction. The process is repeated until all the sensitive itemsets are hidden. Furthermore, three item restriction-based algorithms [15] that are known as Minimum Frequency Item Algorithm (MinFIA), Maximum Frequency Item Algorithm (MaxFIA) and Item Grouping Algorithm (IGA) selectively remove items from transactions that support the sensitive itemsets. Intelligent sanitization in the paper [14] is the variant of their IGA.

We do not focus on the development of the sanitization techniques. Since we compare our new method with the study of Menon et al. [14], we prefer to use one of heuristics in their study. One is blanket sanitization where only one item is retained from the original transaction. The sanitization occurs by eliminating support for every nonsingleton itemset supported by the transaction. The other is intelligent sanitization, where an attempt is made to remove the fewest number of items from the transaction that would result in eliminating the support for every itemset in $\mathscr{F}^R(\sigma_{min})$.

```
1:  for transactions $i \in \mathscr{D}$ such that $i$ is to be sanitized
2:      identify all sensitive frequent item sets $\mathscr{F}_i^R \in \mathscr{F}^R(\sigma_{min})$ supported by $i$
3:      while $\mathscr{F}_i^R \neq \emptyset$
4:          calculate $f_j = |\{k \in \mathscr{F}_i^R \mid j \in k\}|$,          $\forall$ items $j$ in $i$
5:          remove item$j^* = argmax_j\{f_j\} \in m$
6:          update $\mathscr{F}_i^R = \mathscr{F}_i^R \backslash \{k \in \mathscr{F}_i^R | j^* \in k\}$
7:      end while
8:  end for
```

Fig. 2. Intelligent Sanitization Algorithm [14].

It is shown that the intelligent sanitization produces less distortion on nonsensitive itemsets thereby removing less number of items when this is compared with the blanket sanitization. So, we prefer to use the intelligent sanitization to hide itemsets of transactions that are identified by the method described in the previous section. In order to be self-contained, we give intelligent sanitization algorithm in Figure 2.

Let us explain this with an example; we choose transaction T_1 that is represented in the constraint matrix by t_1 (on line 1). T_1 supports two sensitive itemsets - r_2 and r_4 (on line 2). First, the item appearing in the most number of sensitive patterns supported by that transaction is selected from all items in $r_2 \cup r_4$ (Items 1, 2, 3, 8). For the example, "3" is selected, because it appears twice while each one of the others appears once. Delete "3" (on line 4 and 5) and remove all sensitive itemsets that contain selected item (on line 6). This action eliminates r_2 and r_4 at the same time. If sensitive itemsets remain supported by T_1, repeat the procedure (on line 3). For our example, there is no sensitive itemset left. The sanitized transaction is $\{1, 2, 7, 8, 10\}$. When all transaction are put in process, the sanitized database is generated with the number of modifications on the database $\mathscr{D}$ is 7. After the entire process, 17 itemsets that were previously frequent are still frequent whereas 13 itemsets that were frequent before the sanitization are no longer frequent.

IV. Performance Evaluation

We performed Coefficient-Based Itemset Hiding and the study of Menon et al. [14] on real datasets using different parameters such as number of sensitive itemsets and minimum support value. Our code was implemented in Java on a Windows 7 - PC with Intel Core i5, 2.67 GHz processor. We performed exact parts of the experiments by using GNU GLPK [12]. In this section, features of datasets, selected parameters and results are explained in detail.

A. The Datasets

All datasets we use in our experiments are available through Frequent Itemset Mining Implementations Repository - FIMI.

TABLE III. CHARACTERISTICS OF THE REAL DATASETS

Database name	Number of transactions	Number of items	Avg. trans. length	Number of nonsingleton frequent itemsets (support level used in the experiments)
kosarak	990,002	41,270	8.10	1,462 (0.5%)
retail	88,162	16,470	10.30	5,472 (0.1%)
				15,316 (0.05%)
mushroom	8,124	119	23.00	53,540 (20%)

The kosarak dataset, which is provided by Ferenc Bodon [4], is a very large dataset containing 990,002 sequences of click-stream data from a Hungarian on-line news portal. It has medium-level of sparsity and medium-level of density. The retail is a sparse dataset and was reported in Brijs et al. 1999 [5]. It includes the retail market basket data from an anonymous Belgian retail store. The mushroom, which was generated by Roberto Bayardo from the UCI datasets and PUMSB [3], has high-level of density. These datasets have different characteristics such as the number of transactions, varieties of items and level of sparsity - density. These variations contribute to our experiment and give a chance to measure the efficacy of our study. Table 3 includes summary information about these datasets.

B. Evaluation Methodology

We compare our approach with the approach of Menon et al. [14] in terms of the number of nonsensitive itemsets that are lost (information loss) and the ratio of the number of transactions that are not sanitized and the total number of transactions in the database (accuracy).

Execution times for coefficient computation (C), integer programming (IP) and heuristic sanitization (H) are separately recorded to maintain the total time for Coefficient-Based Itemset Hiding. Because the complexity is one of main problems for the privacy preserving data mining, the time is illustrated in detail in our experiments.

With our original sensitive itemsets, supersets of them are become hidden in the databases, since any itemsets that contains sensitive itemsets should also be hidden. Original sensitive itemsets are specified with various lengths, such as 10, 20 and 50. In addition, we use two different minimum support thresholds for the retail dataset to evaluate the impact of threshold.

C. Experimental Results

In Table 4, accuracy, time and information loss performances of Menon et al. [14] and Coefficient-Based Itemset Hiding are given. Table 5 summarizes the differences between two approaches. Negative values represent the sacrifices of our approach while positive values show outclass performance of our new method over the approach of Menon et al.

When the tables are carefully examined, it is deduced that our new method makes progress with different fluctuations based on the characteristics of databases used in experiments. Firstly, it can be noticed that the proposed method decreased the number of lost nonsensitive itemsets successfully for all kinds of databases in the experiments. However, it is obviously seen that our approach works more powerfully for sparse databases such as Retail when we compare it with the other databases in Table 4 and 5. Support level used in

TABLE IV. RESULTS FROM THE REAL DATA

DB name (σ_{min})	Sensitive itemsets (with supersets)	Approach of Menon et al.					Coefficient-Based Approach					
		(%)	Time (sec)			Itemsets(#)	(%)	Time (sec)				Itemsets(#)
		Accuracy	IP	H	Total	Info. Loss	Accuracy	C	IP	H	Total	Info. Loss
kosarak (4,950)	10 (18)	99.59	11.5	129	140.5	98	99.27	71	11.3	124	206.3	19
	20 (31)	99.23	27.7	126	153.7	182	98.5	141	67.7	116	324.7	57
	50 (65)	98.95	35,976.1	115	36,091	310	97	360	6,543	120	7,023	58
retail (88)	10 (10)	99.83	0	7	7	10	99.8	1	0.1	8	9.1	2
	20 (20)	99.6	0.1	8	8.1	61	99.42	1	0.1	8	9.1	10
	50 (65)	99.05	0.1	8	8.1	98	98.43	3	0.3	8	11.3	26
retail (44)	10 (15)	99.37	0.1	7	7.1	85	99.34	2	0.1	7	9.1	43
	20 (32)	98.77	0.1	8	8.1	335	98.54	4	0.1	8	12.1	196
	50 (97)	97.46	0.2	8	8.2	664	96.62	8	0.3	8	16.3	364
mushroom (1,625)	10 (2336)	93.4	0.1	1	1.1	19,984	93.4	89	0.1	1	90.1	19,584
	20 (2395)	93.16	0.2	1	1.2	33,049	84.94	114	0.5	1	115.5	26,791
	50 (5341)	92.32	0.3	1	1.3	35,831	79.47	141	0.8	1	142.8	31,149

TABLE V. DIFFERENCE OF TWO APPROACHES

DB name (σ_{min})	Sensitive itemsets (with supersets)	Perc. (%)		
		Accuracy	Total Time	Info. Loss
kosarak (4,950)	10 (18)	-0,32%	-46,83%	80,61%
	20 (31)	-0,74%	-111,26%	68,68%
	50 (65)	-1,97%	80,54%	81,29%
retail (88)	10 (10)	-0,03%	-30,00%	80,00%
	20 (20)	-0,18%	-12,35%	83,61%
	50 (65)	-0,63%	-39,51%	73,47%
retail (44)	10 (15)	-0,03%	-28,17%	49,41%
	20 (32)	-0,23%	-49,38%	41,49%
	50 (97)	-0,86%	-98,78%	45,18%
mushroom (1,625)	10 (2336)	0,00%	-8090,91%	2,00%
	20 (2395)	-8,82%	-9525,00%	18,94%
	50 (5341)	-13,92%	-10884,62%	13,07%

experiments may be another critical point. Because decreasing the support value of itemsets below low support level needs sacrificing utility gain, low support level reduces the coefficient benefit for information loss, as Retail performance result at the support level – 44 (0.05%). However, the best performance in all experiments is attained for Retail database at the level of support - 88 (0.10%).

The performance result of Kosarak database shows that new approach works well with databases which have medium sparsity and density. It has up to 80% gain on information loss while there is not above 2% accuracy loss. On other hand, since Kosarak is a very large database, total time cost is a general problem for integer programming solutions. Despite this, it is also remarkable that in case of 50 sensitive itemsets in Kosarak, our method is approximately six times better in execution time. This is quite reasonable since coefficients help branch and cut algorithms of integer programming [7] by allowing more cuts.

When the result of Mushroom database in Table 5 is examined, it is deduced that although better performance than the approach of Menon et al. [14] has is gained for the information loss, we meet undesirable accuracy loss and time cost. This result shows that in some situation such having a need of use very dense database like Mushroom, using the methods of Menon et al. [14] or different exact methods is more useful and produces better solutions.

V. RELATED WORK

One of the earlier studies, which presented the principles of privacy preserving data mining, belongs to Atallah et al. [2]. Their study proves that "association rule hiding" is NP-hard problem, due to the existence of large databases. After this research, there has been remarkable growth on the number of research on this issue recently. They are generally classified based on their proposed approach as heuristic, border-based and exact. In addition to these, as another branch of privacy preservation, the utility that involves the term of information loss has examined in detail in utility-based privacy preserving data mining.

Dasseni et al. [6] generalize the hiding problem in the sense that they consider the hiding of both sensitive frequent itemsets and sensitive association rules. The authors propose three single rule heuristic hiding algorithms that are based on the reduction of either the support or the confidence of the sensitive rules, but not both. In all three approaches, the goal is to hide the sensitive rules while minimally affecting the support of the nonsensitive itemsets. In order to achieve this, transactions are modified by removing some items, or inserting new items depending on the hiding strategy. Verykios et al. [18] extend the previous work of Dasseni et al. [6] by improving and evaluating the association rule hiding algorithms of [6] for their performance under different sizes of input datasets and different sets of sensitive rules. Oliveira and Zaïane [15] contribute to this area with a variety of heuristics. Particularly, The Item Grouping Algorithm is based on grouping sensitive association rules sharing the same itemsets. The minimum impact on the disclosed database is provided by deleting the shared items. The intelligent sanitization we use as the sanitization technique in our study is the variant of this algorithm. Amiri [1] presented three effective, multiple association rule hiding heuristics that outperform the previous heuristics studies by offering higher data utility and lower distortion, at the expense of increased computational speed. Although the algorithms by Amiri are similar in philosophy to the previous approaches, the three proposed methodologies do a better job in modeling the overall objective of a rule hiding algorithm.

The paper by Sun and Yu [16, 17] is a pioneer of border-based researches which use the border theory to hide frequent itemsets. It aims at maintaining the frequency of nonsensitive itemsets to minimize the side-effects and evaluate the impact on the result database. Gkoulalas-Divanis and Verykios used this border concept in their works [8, 10, 11] to minimize overconcealing nonsensitive itemsets. They capture the itemsets hiding process as a border revision operation and they presented a set of algorithms which enable the computation of the revised borders that pertain to an exact hiding solution.

The paper written by Menon et al. [14] includes an interesting approach to the problem of privacy preserving data mining. They were the first to present an integer programming optimization method that consisted of an exact and a heuristic part to hide frequent itemsets. The exact part of the method uses the database to formulate an integer program trying to obtain the minimum number of transactions that have to be sanitized. The researches of Gkoulalas-Divanis and Verykios [8, 9, 10, 11] are based on this exact methodology. However, they organize the integer program formulation in a way of identifying itemsets to hide, instead of transactions.

The information loss, which is considered a loss of utility for data mining purposes, has been examined in detail in another research area, utility-based privacy preservation. This broad approach can preserve considerable utility of the data set without violating privacy. Li and Li [13] mention the importance of utility gained by publishing database for the society as a whole and claim that it is inappropriate to directly compare privacy with utility, because of several reasons, including both technical and philosophical ones. Furthermore, they propose an integrated framework for considering privacy-utility tradeoff, borrowing concepts from the Modern Portfolio Theory for financial investment.

Although, heuristic approaches seem scalable and practical, their results are less reliable about being exact solution and providing minimum side-effect. This is not acceptable in many situations. Border-based methods give better solution on the sensitive itemset hiding and side-effect problems. However, the evaluation brings high complexity. The research of Menon et al. [14] makes progress in a way of getting exact solution. Using integer programming and heuristic together gives evaluated impact on the result data than border-based approaches give. However, it causes failing to notice side-effect of information loss. Verykios et al. present exact approaches which find a way to decrease loss of nonsensitive itemsets. But, the complexity of these approaches and not being scalable are the reasons of researching for better solution. Furthermore, Li and Li [13] and Yeh and Hsu [19] inspired us that the utility that is directly about information loss is essential for frequent itemset hiding. We realized that solutions in the literature to the utility loss problem cannot always satisfy the need of databases at different level of utility. Approaches should be flexible to be specialized in terms of utility where necessary.

VI. Conclusion and Future Work

In this paper, we presented an efficient approach to minimize side-effects of accuracy and information loss in itemset hiding problem. The degree of side-effect is represented with coefficients that are placed into the objective function of integer programming. Experiments with real datasets show that our approach minimizes the number of concealed nonsensitive association rules efficiently.

Coefficient Computation Algorithm can be specialized based on the area where the published database is utilized. In this sense, coefficients in the objective function of the integer programming may be used in a more efficient way. Moreover, different optimization techniques can be achieved by exploiting the inherent characteristics of the constraints and objective function that are involved in the CSP, in a more advanced way.

References

[1] A. Amiri, “Dare to Share: Protecting Sensitive Knowledge with Data Sanitization“, Decision Support Systems, vol. 43, iss. 1, 2007, pp. 181-191.

[2] M. Atallah, A. Elmagarmid, M. Ibrahim, E. Bertino, and V. Verykios, “Disclosure Limitation of Sensitive Rules”, KDEX ’99: Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange, 1999, pp. 45-52.

[3] R. Bayardo, “Efficiently Mining Long Patterns from Databases”, Proceedings of the ACM SIGMOD, 1998, pp. 85-93.

[4] F. Bodon, “A fast APRIORI implementation”, Proceedings of Workshop Frequent Itemset Mining Implementations (FIMI’03), vol. 90, CEURWS.org, CEUR Workshop Proceedings, 2003, pp. 56-65.

[5] T. Brijs, G. Swinnen, K. Vanhoof, and G. Wets, “Using Association Rules for Product Assortment Decisions: A Case Study”, Proceeding of the 5th ACM SIGKDD Internat. Conf. Knowledge Discovery Data, Mining. ACM Press, 1999, pp. 254-260.

[6] E. Dasseni, V. S. Verykios, A. K. Elmagarmid, and E. Bertino, “Hiding Association Rules by Using Confidence and Support”, Proceedings of the 4th International Workshop on Information Hiding, 2001, pp. 369–383.

[7] M. Jünger et al., “50 Years of Integer Programming 1958-2008 From the Early Years to the State-of-the-Art”, Springer, 2010.

[8] A. Gkoulalas-Divanis and V. S. Verykios, “An Integer Programming Approach for Frequent Itemset Hiding.”, Proceedings of the ACM Conference on Information and Knowledge Management (CIKM ’06), November 2006, pp. 748-757.

[9] A. Gkoulalas-Divanis and V. S. Verykios, “A Parallelization Framework for Exact Knowledge Hiding in Transactional Databases”, Proceedings of The IFIP TC-11 23rd International Information Security Conference, IFIP 20th World Computer Congress, IFIP SEC 2008, vol. 278, September 2008, pp. 349-363, Springer.

[10] A. Gkoulalas-Divanis and V. S. Verykios, “Exact Knowledge Hiding through Database Extension”, IEEE Transactions on Knowledge and Data Engineering, vol. 21, iss. 5, May 2009, pp. 699-713.

[11] A. Gkoulalas-Divanis and V. S. Verykios, “Hiding Sensitive Knowledge without Side Effects”, Knowledge and Information Systems, vol. 20, iss. 3, August 2009, pp. 263-299.

[12] GLPK. GNU GLPK 4.32 User’s Manual. Free Software Foundation Inc., Boston, MA, 2008. Available at <http://www.gnu.org/software/glpk/glpk.html> 27.10.2013.

[13] T. Li and N. Li, “On the Tradeoff between Privacy and Utility in Data Publishing”, Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 517-525.

[14] S. Menon, S. Sarkar, and S. Mukherjee, “Maximizing Accuracy of Shared Databases when Concealing Sensitive Patterns”, Information Systems Research, vol. 16, no. 3, September 2005, pp. 256–270.

[15] S. R. M. Oliveira and O. R. Zaïane, “Privacy Preserving Frequent Itemset Mining”, Proceedings of the IEEE ICDM Workshop Privacy, Security Data Mining, 2002, pp. 43–54, Australian Computer Society.

[16] X. Sun and P. S. Yu, “A Border-Based Approach for Hiding Sensitive Frequent Itemsets”, Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM’05), 2005, pp. 426–433.

[17] X. Sun and P. S. Yu, “Hiding Sensitive Frequent Itemsets by a Border–Based Approach”, Computing Science and Engineering, vol. 1, iss. 1, September 2007, pp. 74–94.

[18] V. Verykios, A. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, “Association Rule Hiding”, IEEE Transactions on Knowledge and Data Engineering, vol. 16, iss. 4, April 2004, pp. 434-447.

[19] J. S. Yeh and P. C. Hsu, “HHUIF and MSICF: Novel Algorithms for Privacy Preserving Utility Mining”, Expert Systems with Applications, vol. 37, iss. 7, 2010, pp. 4779-4786.