CIT. Journal of Computing and Information Technology, Vol. 25, No. 4, December 2017, 279–290
doi: 10.20532/cit.2017.1003680

An Efficient Rule-Hiding Method For Privacy Preserving in Transactional Databases

Farsad Zamani Boroujeni and Doryaneh Hossien Afshari

Department of Computer Engineering, Khorasgan (Isfahan) Branch, Islamic Azad University, Isfahan, Iran

One of the obstacles in using data mining techniques such as association rules is the risk of leakage of sensitive data after the data is released to the public. Therefore, a trade-off between the data privacy and data mining is of a great importance and must be managed carefully. In this study, an efficient algorithm is introduced for preserving the privacy of association rules according to distortion-based method, in which the sensitive association rules are hidden through deletion and reinsertion of items in the database. In this algorithm, in order to reduce the side effects on non-sensitive rules, the item correlation between sensitive and non-sensitive rules is calculated and the item with the minimum influence in non-sensitive rules is selected as the victim item. To reduce the degree of data distortion and preservation of data quality, transactions with highest number of sensitive items are selected for modification. The results show that the proposed algorithm has better performance in real non-dense databases as well as less side effects and less data loss compared to its performance in real dense databases. Further, more improvements are achieved for synthetic databases in comparison with real databases.

ACM CCS (2012) Classification: Information systems → Information systems applications → Data mining → Association rules

Security and privacy → Database and storage security → Data anonymization and sanitization

Security and privacy → Software and application security → Domain-specific security and privacy architectures

Keywords: association rules hiding, sensitive rule, distortion-based method

1. Introduction

Today, to enhance decision making, many companies and organizations use data mining in order to extract useful and desirable patterns from their repositories and databases. Contrary to owners' desire, this can lead to disclosure of sensitive information contained in the database which is a risk to them. Among many techniques, extracting the association rules is one of the widely-used data mining techniques showing hidden relations and dependencies between itemsets in the database. The extracted rules may include some sensitive information such that its public release can be extremely destructive. As an example in medical databases, sharing information about a disease will be useful for the progress of medical science. Yet, releasing the information of patients must be avoided. Therefore, to maintain the database security and to prevent the extraction of the sensitive patterns, a new discipline of data mining was introduced called Privacy Preserving Data Mining (PPDM). Most of the PPDM algorithms aim to apply changes in database so that sensitive information cannot be extracted even after applying well-known rule extraction techniques. These rule hiding algorithms mainly rely on minimal modifications of data items, either by replacing new values or removing/inserting items in a set of selected transactions to decrease the rules' interestingness measures, e.g. support and confidence.

The state-of-the-art approaches provide solutions for hiding sensitive association rules with multiple left-hand side (LHS) and right-hand side (RHS) items while maintaining the quality of the original data. Recently, Komal Shah *et al.* introduced RRLR algorithm which proved to be efficient in hiding sensitive rules [1]. However,

it suffers from some limitations like high level of hiding failure and inability to hide the rules with multiple LHS. items. This study presents a heuristic algorithm which aims at overcoming the limitations of RRLR algorithm. In contrast to its counterparts, the proposed algorithm does not require removing and inserting all LHS items to the transactions. To hide the sensitive rules, it only removes and reinserts a limited number of appropriate items leading to minimal side effects and maintaining the data quality.

The remainder of this paper is organized as follows: Section 2 presents an overview of the existing approaches and some theoretical background. Section 3 describes the proposed algorithm in detail and gives an example which illustrates how the proposed algorithm works. Section 4 is dedicated to performance evaluation of the proposed algorithm. Finally, the results of comparative performance experiments are presented in Section 5. Conclusions and future works are described in the last section.

2. Background and Related Works

Hiding sensitive association rules is the process of applying changes to the original database and converting it to a sanitized one. So that, except sensitive rules, all of the useful association rules can be extracted from the sanitized database. The problem of association rule hiding can be defined as follows:

The inputs of the algorithm include an original database D which is supposed to be released to the public, a minimum support threshold (*MST*), a minimum confidence threshold (*MCT*), a set of sensitive association rules $R_s \subseteq R$ to be hidden from the association rule extraction algorithms. The outputs include a sanitized database D' from which the non-sensitive rules $R - R_s$ can be extracted as much as possible [2].

In the rule hiding algorithms, the sensitive rules are hidden by altering their support and/or confidence through insertion or deletion of the items in the original database. These operations may result in the appearance of four important side effects which are defined and measured by the following quality metrics [3]:

1. Hiding Failure (hiding rate) which is used to measure the percentage of sensitive association rules that remain disclosed possibly due to the process of hiding some other sensitive rules;
2. Misses Cost (lost rules) which reflects the percentage of non-sensitive rules that are hidden inadvertently by the rule hiding algorithm;
3. Artificial Patterns (ghost rules) that measures the percentage of false association rules that can be extracted from the sanitized database among all of the rules whose confidence is below the pre-defined minimum confidence threshold;
4. Dissimilarity (altered rate) measures the percentage of altered transactions among the entire database.

Most of the association rule hiding algorithms aim at producing their output with minimal side effects such that:

1. Sensitive rules are hidden and are not extracted from the sanitized database (hiding failure reaches zero).
2. There is no lost rule; all non-sensitive and useful rules should be extracted from sanitized database.
3. Ghost rules should not be produced; in other words, non-existing rules in the main database cannot be produced from the sanitized database.
4. The lower rate of database modifications is performed. When more transactions are modified, more processing time would be required [3].

Let $I = \{I_1, \ldots, I_n\}$ be a set of items existing in database D. Each association rule is denoted as $X \rightarrow Y$, where X and Y are LHS and RHS itemsets respectively such that $X, Y \subset I$ and $X \cap Y = \varnothing$. The interestingness of an association rule is measured by its support and confidence which are calculated by equations (1) and (2) [4].

$$Support\left(X \rightarrow Y\right) = \frac{\left|X \cup Y\right|}{\left|D\right|} \tag{1}$$

$$Confidence\left(X \rightarrow Y\right) = \frac{\left|X \cup Y\right|}{\left|X\right|} \tag{2}$$

Where the support reflects the frequency of the rule and the confidence refers to the strength of the rule within the transactional database. In both equations, $|X \cup Y|$ denotes the number of transactions that contain both itemsets X and Y. $|D|$ is the whole number of the existing transactions in database and $|X|$ is the number of transactions of database that include itemset X. Along with these measures, two threshold quantities named minimum support threshold (*MST*) and minimum confidence threshold (*MCT*) are frequently used as gauges to assess the interestingness of the rules.

Approaches to hide association rules are commonly categorized as heuristic approaches [5] – [9], border based approaches [10] – [12], exact approaches [13] – [15] and reconstruction based approaches [16]. Heuristic techniques try to find a near-optimal solution for hiding as many sensitive rules as possible while minimizing the possible side effects. Since it is almost impossible to minimize all side effects simultaneously, they rely on optimizing certain sub-goals in the hiding process, while they do not guarantee optimality. The border based approach uses the theory of borders [17] to hide sensitive association rules. The idea is to define the borders by identifying itemsets which are at the position of the borderline separating the frequent and infrequent itemsets. The borders are greedily revised to hide association rules with minimal side effects. The third category involves exact and non-heuristic algorithms, which formulates the hiding process as an optimization problem which is solved by integer programming. The exact approaches guarantee the optimality, but may fail to provide a solution for all situations. The algorithms that follow the exact approach suffer from high computational burden required for searching a large state space. The fourth category involves reconstruction based algorithms that are used to accurately estimate the distribution of original data values. Using the estimated distributions, the goal of the algorithm is to build a classifier for distortion of values in the database.

This study is motivated by the application of heuristic approaches which are categorized into blocking-based and distortion-based methods. The former relies on replacing certain data items with an unknown value such as "?". The algorithms presented in [18], [19] make use of blocking technique to hide the sensitive rules. Although they do not distort values, they insert many unknown values into the database. The problem with this technique is that the transactions that contain items with unknown values can be used as cues to find sensitive information. In fact, all the rules that their generating itemsets contain question marks and their maximum confidence is above the *MCT* could be the sensitive rules that we want to hide from the adversary. The latter refers to a common approach in which a set of selected items is either removed from or inserted to the transactions with the aim of increasing or decreasing the support and confidence of the sensitive rules. Due to their undesirable side effects, the heuristic approaches are unable to provide an optimal method.

The subject of association rules hiding was first suggested by Attallah *et al*. [20] who developed an association rule hiding technique with reasonable time complexity in which a lattice-like graph has been utilized for concealment. Verykios *et al*. [21] have presented five different algorithms, namely 1.a, 1.b, 2.a, 2.b and 2.c; two of which use the support reduction strategy and the other three algorithms use reduction of confidence for hiding the sensitive rules. Although some of the algorithms mentioned above, e.g. 2.c and 2.b, produce minimal side effects, almost all of them can hide only one sensitive rule at a time, which limits the applicability of these algorithms. Oliveira and Zaïane [9] proposed the first generation of algorithms that hide multiple rules, e.g. Naïve, MinFIA, MaxFIA and IGA. Their idea was to hide sensitive rules based on their conflict level at the cost of high execution time. Another multiple rule hiding algorithm is DSRRC which effectively hides the sensitive association rules by means of clustering and use of a sensitivity measure [22]. However, its limitation in hiding sensitive rules with multiple RHS and LHS items makes it inappropriate for real-world applications. Later, the same research group developed a modified version of DSRRC, called MDSRRC algorithm [23], to prevail the limitations associated with DSRRC algorithm

In another study, Hong *et al*. [24] conducted a research in which the reduction of support technique is used for hiding association rules. Generating low amount of ghost rules is the main

advantage of this method, but requires high computation time that makes it unsuitable for practical applications. Later, Cheng *et al*. [25] tried to use a greedy approach for reduction of rules' support through removing right hand side items. The main drawback of this method is high number of database scans. It also requires large number of database modifications which results in low quality of the sanitized database.

Recently, two algorithms ADSRRC and RRLR were presented in [1] to overcome the other limitations of DSRRC algorithm. The first limitation is related to the dependency of DSRRC algorithm on the ordering of transactions, which is absolutely necessary in order to remove items from the database. Further, it requires to sort all the transactions in the database each time an item is removed from a transaction, which highly degrades the performance of the algorithm. These two issues have been addressed in ADSRRC algorithm. The ADSRRC algorithm tries to conceal sensitive association rules through reducing support and confidence of the rules using a clustering method. This technique requires two times scan of the database and has proved to be remarkably efficient in terms of low hiding failure. Conversely, it suffers from high rate of lost rules and high computational complexity. Another limitation of the DSRRC algorithm is that it does not have the ability to hide rules with multiple RHS items. To overcome this problem, the RRLR algorithm is suggested instead. The RRLR algorithm reduces the support and confidence of the rules to hide the sensitive ones, but is unable to hide the rules with multiple LHS items. This paper aims at tackling this problem by proposing an efficient algorithm called ARRLR which can hide the sensitive rules with multiple RHS and LHS items. In the next section, the technical details of ARRLR are presented.

3. The Proposed Algorithm

This section provides a brief overview of the underlying concepts to better understand the multiple-item rule hiding algorithms. Generally, hiding sensitive rules in the form of $M \rightarrow N$, can be performed by decreasing either the confidence or support of the rules to below the *MST* and *MCT*. In fact, the support of the rule $M \rightarrow N$ can be decreased by reducing the occurrences of itemset MN. Similarly, the confidence of the rules can be reduced by one of the following techniques:

1. increasing the support of the antecedent of the rule, i.e. LHS part, in transactions in which the consequence of rule is not present.
2. Decreasing the support of rule's consequence, i.e. RHS part, in transactions including both parts of the rule [21].

The proposed method aims at hiding sensitive rules with multiple items in LHS and multiple items in RHS. This type of rule is represented in the form of $aM \rightarrow bN$ where $a, b \in I$ and $M, N \subset I$. Here, 'a' and 'b' are single items selected by the algorithm to be inserted into or removed from LHS or RHS of the rule, respectively. The idea behind the proposed algorithm is to decrease both support and confidence measures to hide the sensitive rules. The support of the rules that are in the form of $aM \rightarrow bN$ can be decreased by reducing the support of the itemset $aMbN$. Also, to decrease the support of the large itemset $aMbN$, a suitable item is selected and removed from the LHS. items of the sensitive rule; and to decrease confidence of the sensitive rule, a selected item is inserted in the suitable transaction. To this end, the algorithm identifies a list of suitable transactions for modification that are called victim transactions. To identify a set of items to remove from the victim transactions, two objective parameters, namely α and β, are calculated for items that exist in the sensitive rules. They are also used to calculate the total sensitivity of the victim transactions. These parameters are defined as follows:

1. Parameter α: The number of occurrence for LHS items of the sensitive rules in the whole set of non-sensitive rules. This parameter is used to construct the list $L\alpha$ in which LHS items of the sensitive rules are sorted in ascending order, according to the value of α.
2. Parameter β: The frequency of an item in the set of sensitive rules, which is used to compute the sensitivity of each transaction.
3. Sensitivity of transaction: The sensitivity of a transaction is calculated as the sum of β values of all sensitive items included in that transaction.

In the first step, association rules are mined from the original database by using the Apriori algorithm. Then, sensitive rules are selected from the mined rules (R_s) and are sorted in descending order, according to their confidence value. To make sure that all sensitive rules are hidden, a Boolean variable named "State" is defined to maintain the hiding status of each sensitive rule. The states of all sensitive rules are initially set to *false*. Now, parameters α, β and sensitivity of transaction are computed by the algorithm and the transactions are arranged in descending order, based on their sensitivity and length. Next, $L\alpha$ is computed by the algorithm and the process of hiding sensitive rules starts from the first sensitive rule.

Among the LHS items of the first sensitive rule, an item with the least value of α (according to list $L\alpha$) is selected to be removed. Then, the selected item is removed from the first transaction that has the highest sensitivity and length. After removing, the selected item is inserted in the transactions which do not have the item (i.e. the large itemsets that partially support LHS of the rule and partially support, or do not support RHS of the rule). If the suitable transaction for the selected item is not found, the insertion will not be done. During the process of hiding, the sensitivity of transactions will not be updated. After each removal and insertion, support and confidence of the sensitive rules existing in R_s will be updated. If they reach below the *MST* and *MCT*, the *false* state of the sensitive rule is changed to true, without being removed from the list R_s. The rule state is changed from *true* to *false*, if a sensitive rule becomes disclosed because of inserting an item. In this situation, there will be no insertion and the rule is hidden just by removing a suitable item of the left side ones. As shown in Figure 1, a Boolean variable, called Disclosed, is used for identifying disclosed rules and preventing the algorithm from calling item insertion process. The hiding goes on until the state values of all sensitive rules become *true*.

Finally, the transactions in the original database are modified and constitute a new database that is a sanitized version of the original database *D*. This preserves the privacy of the sensitive data and keeps data quality. The main steps of the proposed algorithm named ARRLR (Advanced Remove and Reinsert LHS of Rule) are shown in Figure 1.

```
Input: Original database D, (MCT), (MST).
Output: The sanitized database D'.
Use Apriori, extract association Rules.
Select a set of sensitive rules R_sensitive (R_s)
Set the State and Disclosed variable of all sensitive
rules to false
Sort R_s in decreasing order based on their Confidence.
Calculate α, β and sensitivity of transactions.
Create Lα by sorting LHS items of the sensitive rules
in ascending order based on their α value
Arrange transactions in decreasing order of their sensi-
tivity and length.
while (states of all R_sensitive are not true)
{
 Find the first rule R_k from R_sensitive such that R_k. State
 is false
  Select item I from LHS of rule R_k according to Lα
  //Antecedent Deletion Process
  for m = 1 to no. of transactions in database
   {
   if (T_m supports both parts of rule R_k)
   Remove selected antecedent item I from
   transaction T_m
   if (R_k. Disclosed is false)
    {
     // Antecedent Insertion Process
      for n = m to no. of transactions in database
      {
       if (T_n does not include item I and partially
       supports rule R_k)
       Insert selected LHS item I in transaction T_n
      }
     }
  for each rule R in R_S
   {
    Recalculate Support & Confidence for R
    if (R.Support < MST or R.Confidence < MCT)
    Set R.State to true.
    else
    {
    If (R.State is true)
    Set R.Disclosed to true
    Set R.State to false.
    }
   }
}
```

Figure 1. Pseudo code of algorithm ARRLR (Advance Remove and Reinsert Left hand side Rule).

3.1. Case Study

In this section the function of ARRLR algorithm is explained by an example. Considering the transactional database D shown in Table 1, the rules that are shown in Table 2 are achieved after employing the Apriori algorithm on database D with $MST = 30\%$ and $MCT = 70\%$.

Table 1. Sample database.

TID	Itemset
1	*mop*
2	*mopt*
3	*mnopqrt*
4	*mnprs*
5	*nopq*
6	*mno*
7	*opqrt*
8	*mnos*
9	*noqrs*
10	*mops*
11	*nrs*
12	*mnopq*
13	*mopqrt*
14	*ns*
15	*mnopt*

Assume that three sensitive rules 5, 13 and 22 are sorted based on their confidence with confidence values of 100%, 83.33%, and 71.43% respectively. Since item "o" is included in all sensitive rules, its β sensitivity is 3. Accordingly, the β sensitivity of m, n, p are calculated as 2, 2 and 1. Next, the algorithm calculates the number of occurrences for each item that exists in LHS of sensitive rules, in all non-sensitive rules (α value), arranging those items in increasing order of their α value. Therefore, $L\alpha = \{n = 2, t = 4, m = 9, p = 16, o = 17\}$. To compute the transaction sensitivity, the sum of sensitivity values (β sensitivity) of itemset that exists in that transaction is calculated. Table 3 shows item sensitivity values of α and β as well as transaction sensitivity sorted in decreasing order of sensitivity and length. Then the algorithm selects a sensitive rule with highest confidence for hiding. In this example $pt \rightarrow o$ is selected as first rule for hiding. According to $L\alpha$ and rule $pt \rightarrow o$, item "t" is selected for deletion. Now, the algorithm selects a transaction with highest sensitivity (in this example, transaction $TID = 3$ is the most sensitive transaction) and deletes item "t" from that transaction. Thus, the support of large itemset "*opt*" is decreased. After deletion, LHS item should be inserted in the most appropriate transaction selected from the list of sorted transactions, as shown in Table 3. The most appropriate transaction is a candidate transaction that satisfies the following criteria:

Table 2. Mined association rules.

Extract association rules from sample Database	
1. $q \rightarrow o$	2. $t \rightarrow o$
3. $t \rightarrow p$	4. $pq \rightarrow o$
5. $pt \rightarrow o$	6. $ot \rightarrow p$
7. $t \rightarrow op$	8. $m \rightarrow o$
9. $p \rightarrow o$	10. $mp \rightarrow o$
11. $s \rightarrow n$	12. $q \rightarrow p$
13. $mn \rightarrow o$	14. $oq \rightarrow p$
15. $q \rightarrow op$	16. $p \rightarrow m$
17. $m \rightarrow p$	18. $op \rightarrow m$
19. $mo \rightarrow p$	20. $o \rightarrow m$
21. $o \rightarrow p$	22. $no \rightarrow m$
23. $n \rightarrow o$	24. $p \rightarrow mo$
25. $m \rightarrow op$	

1. if the sensitive rule has only one LHS item, the candidate transaction should have all items of sensitive rule except the LHS item that is considered for insertion.
2. If the sensitive rule has more than one LHS items, the candidate transaction should partially support them, i.e. it should contain all the LHS items of the sensitive rule except for the LHS item that is considered for insertion.

Furthermore, the candidate transaction should not have all the RHS items of the sensitive rule. In other words, the candidate transaction should lack one or more RHS items of the sensitive

rule. In this example, item "*t*" is inserted in the candidate transaction with *TID* = 4.

So, the confidence of the selected rule is decreased due to the increase in support of LHS itemset. If such a candidate transaction is not found, the algorithm ignores the insertion of the LHS item. For example, for rule $mn \rightarrow o$, item "*n*" is removed from transaction with *TID* = 3, but not inserted. After deletion and insertion, confidence and support of all sensitive rules are updated. If they reach below the *MST* and *MCT*, the false state of the rule is changed to True without being removed from the list R_s. The hiding goes on till the state values of all sensitive rules become true. The sanitized database is also shown in Table 3.

4. Performance Evaluation

We conducted two groups of experiments, both on real and synthetic datasets. The real dataset is collected from the website (fimi.ua.ac.be/data) [26]. All of the experiments were conducted on a PC with core-i5 2.40 GHz CPU and 4 GB memory, running under Windows 7 64-bit operating system. The characteristics of the dataset are shown in Table 4.

Table 4. Database characteristics.

Database	No. of Transactions	Avg. of Length	No. of Items
Mushroom	8.128	23	119
Chess	3.196	37	75
T1k	1000	5.4	30

4.1. Performance Measures

Performance measures reflect the quality of hiding in terms of hidden effects, side effects and database effects. The measures include Hiding Failure, Misses Cost, Artificial Patterns and Dissimilarity which are defined as follows [4]:

Hiding Failure (HF): Hiding Failure indicates the number of sensitive rules which can still be explored by the rule extraction algorithm. It is calculated by the relation between the number of sensitive rules in sanitized database and the number of sensitive rules in the original database. This measure is calculated by the following equation [4]:

$$HF = \frac{\left|R_s\left(D'\right)\right|}{\left|R_s\left(D\right)\right|} \tag{3}$$

Table 3. Item sensitivity, sorted transactions and final sanitized database.

Item sensitivity		Sorted Transactions				Sanitized Database	
		TID	Itemset	Sensitivities	Length	TID	Itemset
β	α	3	*mnopqrt*	9	7	1	*mop*
$m = 2$	$m = 9$	15	*mnopt*	9	5	2	*mopt*
$n = 2$	$n = 2$	12	*mnopq*	8	5	3	*mopqr*
$o = 3$	$o = 17$	13	*mopqrt*	7	6	4	*mnprst*
$p = 1$	$p = 16$	2	*mopt*	7	4	5	*nopq*
$t = 1$	$t = 4$	8	*mnos*	7	4	6	*mno*
		6	*mno*	7	3	7	*opqrt*
		5	*nopq*	6	4	8	*mnos*
		10	*mops*	6	4	9	*noqrs*
		1	*mop*	6	3	10	*mops*
		7	*opqrt*	5	5	11	*nrs*
		4	*mnprs*	5	5	12	*mnopq*
		9	*noqrs*	5	5	13	*mopqrt*
		11	*nrs*	2	3	14	*ns*
		14	*ns*	2	2	15	*mnopt*

Where, $|R_s(D')|$ and $|R_s(D)|$are the number of sensitive rules extracted from modified database D' and the original database D, respectively.

Misses Cost (MC): This performance measure is used to show the percentage of the non-sensitive rules that are hidden as a side-effect of the sanitization process. The misses cost is calculated as follows [4]:

$$MC = \frac{\left|\sim R_s(D)\right| - \left|\sim R_s(D')\right|}{\left|\sim \pi R_s(D)\right|} \tag{4}$$

Where $\sim R_s(D)$ denotes the numbers of non-sensitive rules discovered from the original database D, and $\sim R_s(D')$ denotes the number of non-sensitive rules discovered from modified database (D′).

Dissimilarity (Diss): the dissimilarity measure is calculated according to the formula suggested by [4]:

$$Diss(D, D') = \frac{1}{\sum_{i=1}^{n} f_D(i)} \times \sum_{i=n}^{n} \left[f_D(i) - f_{D'}(i) \right] \tag{5}$$

Where $f_D(i)$ denotes repetition of the i-th item in the database D, and n is the number of different items in the initial database D.

Artificial Patterns (AP): This performance factor is used to measure the percentage of the extracted rules that are ghost. It is calculated as follows [4]:

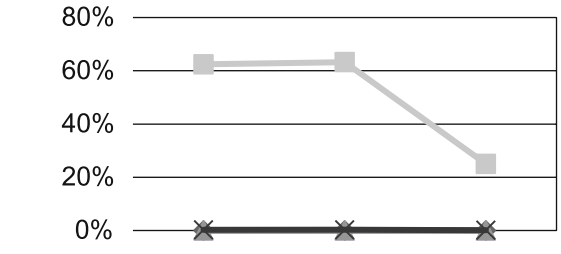
$$AP = \frac{\left|R'\right| - \left|R \cap R'\right|}{\left|R'\right|} \tag{6}$$

Where, $|R'|$ and $|R|$ are the numbers of rules extracted from D' and D, respectively.

4.2. Results and Discussion

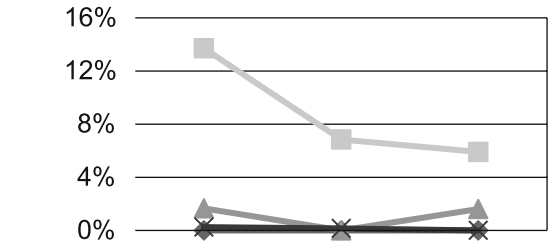
The side effect evaluations for our proposed algorithm are shown in Figures 2–4. To assess the performance of the proposed algorithm, we established three sets of association rules for each of the three datasets shown in Table 4. These sets include 3 and 5 and 7 rules that are randomly selected from the whole set of rules mined from each dataset.

Three different datasets including Chess dataset with 88% support and 90% confidence, Mushroom dataset with 40% support and 60% confidence and synthetic dataset T1K with 2% support and 7% confidence are chosen to extract the rules using the Apriori algorithm. Our algorithm tries to hide the sensitive rules through deletion and re-insertion of an item from the LHS items. Accordingly, two types of side effects occur: Misses Cost and Artificial Pattern. Figures 2–4 show that no sensitive rules have been extracted from the sanitized databases; in other words, hiding failure has not happened for none of the three datasets. Our algorithm has more misses cost for dense Chess dataset than synthetic dataset T1K and non-dense Mushroom dataset because of more difficulty in finding a suitable transaction in dense datasets for re-inserting an item. Since re-inserting an item in this dataset takes place infrequently, more removals are required by the algorithm to hide a sensitive rule and, consequently, more misses cost is produced. On the other hand, as can be seen in Figure 2, reduction of insertions or failure to insert items in Chess dataset, results in less production of artificial patterns. As can be seen in Figure 3, a suitable transaction in a non-dense dataset (Mushroom) can be found more easily for inserting an item than other datasets. As a result, sensitive rules are hidden with fewer removals and better results are achieved. Dissimilarity equals to zero if there are possibilities for removing or re-inserting items in a dataset. It can be inferred from Figures 2–4 that less probability of finding suitable transactions for inserting items will lead to some dissimilarity.



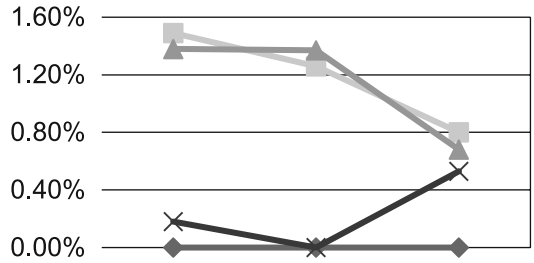
	7-Rules	5-Rules	3-Rules
HF	0%	0%	0%
MC	62.44%	63.22%	25.02%
AP	0.00%	0.00%	0.00%
DISS	0.28%	0.30%	0.11%

Figure 2. Evaluation of the proposed algorithm on Chess dataset.



	7-Rules	5-Rules	3-Rules
HF	0%	0%	0%
MC	13.71%	6.84%	5.91%
AP	1.66%	0.00%	1.61%
DISS	0.36%	0.25%	0.11%

Figure 3. Evaluation of the proposed algorithm on Mushroom dataset.



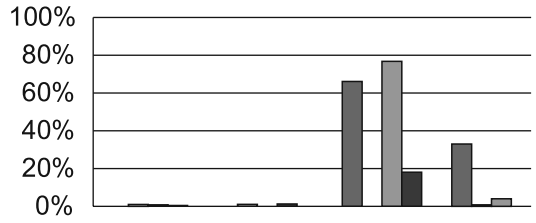
	7-Rules	5-Rules	3-Rules
HF	0%	0%	0%
MC	1.49%	1.26%	0.80%
AP	1.38%	1.37%	0.68%
DISS	0.18%	0.00%	0.53%

Figure 4. Evaluation of the proposed algorithm on Synthetic dataset.

Six experiments, in two categories, are performed to compare the performance of algorithms 1a, MDSRRC and basic RRLR with the proposed one on three database setups shown in Table 4. The first category of the experiments is done with the aim of hiding a set of 3 rules that have a single item in their LHS and the second category is conducted with a set of 7 rules having multiple RHS and LHS items. The results of the first category are shown in Figures 5, 7 and 9. In this category, the results of the proposed algorithm and RRLR are similar. So they are not discussed except for Figure 5 in which a synthetic dataset is used and the hiding failure of RRLR is higher than that of the proposed algorithm.

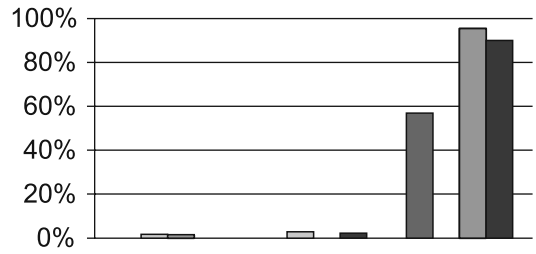
Figure 5 and Figure 6 show the experiments on synthetic dataset T1K. This dataset is dense, but the lengths of transactions are different and an item can be inserted. It is worth noting that, in Figure 5, the RRLR has a high rate of hiding failure. This is due to the fact that RRLR removes the rules that are hidden by the algorithm from the list of sensitive ones. Therefore, inserting an item to hide the next sensitive rule may lead to disclosure of the previous one and, consequently, a hiding failure will occur.

The result of the second category of experiments are presented in Figures 6, 8 and 10. Since RRLR is not able to hide the sensitive rules with multiple LHS items, it is not included in the second category of experiments and is not shown in the corresponding figures.



	ARRLR	MDSRRC	1.a	RRLR
HF	0%	0%	66%	33%
MC	0,80%	1,03%	0,00%	0,68%
AP	0,67%	0,00%	76,64%	3,97%
DISS	0,53%	1,29%	18,11%	0,00%

Figure 5. Comparative evaluation of algorithms with 3-rules on T1k dataset.



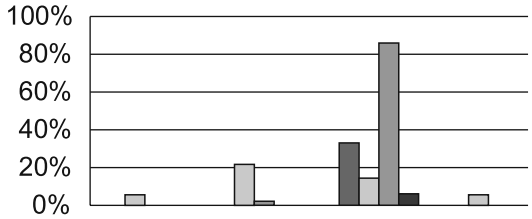
	ARRLR	MDSRRC	1.a
HF	0%	0%	57%
MC	1,49%	2,86%	0,00%
AP	1,38%	0,00%	95,46%
DISS	0,18%	2,23%	90,09%

Figure 6. Comprative evaluation with 7-rules on T1k dataset.

In all experiments performed on three datasets, no failure is reported using the proposed algorithm. Accordingly, in some cases, fewer number of artificial patterns are produced by MDSRRC, but misses cost of our algorithm is less than MDSRRC's.

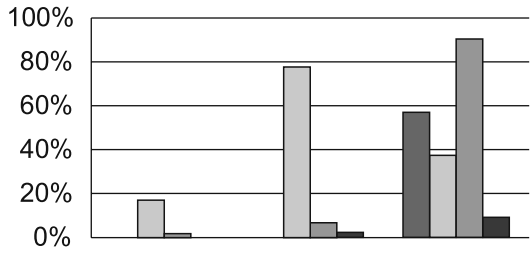
Comparing the results on Mushroom dataset shown in Figure 7 and Figure 8 indicates that misses cost and artificial pattern of the proposed algorithm are less than those of MDSRRC. This is because the proposed algorithm removes an item with the least frequency from the LHS items of the sensitive rules, and re-inserts it, if possible. On the other hand, removing an item with highest frequency by MDSRRC will result in an increase of misses cost and artificial pattern. The strategy of removing and re-inserting a LHS item, in order to hide sensitive rules, makes the proposed algorithm have better dissimilarity for all cases when compared with the others.

According to Figure 9 and Figure 10, the proposed algorithm has less Misses Cost than MDSRRC. However, its Misses Cost is increased relative to the previous experiments. The reason is that more removals are required to hide the sensitive rules due to the dataset's nature and inability to insert an item. In all experiments, the 1.a algorithm produces a large number of artificial patterns and is not able to hide the whole sensitive rules since it uses the strategy of inserting LHS. item. It should be noted that the number of Misses cost, in this algorithm, is not affected by the number of sensitive rules and database size. Low rate of Misses



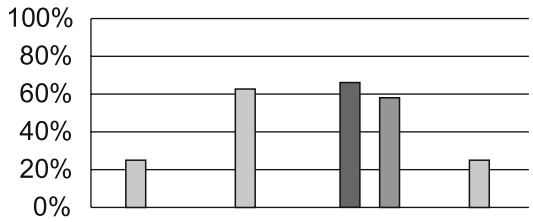
	ARRLR	MDSRRC	1.a	RRLR
HF	0%	0%	33%	0%
MC	5,56%	21,59%	14,29%	5,56%
AP	0,00%	1,71%	85,90%	0,00%
DISS	0,01%	0,29%	6,08%	0,01%

Figure 7. Comparative evaluation with 3-rules on Mushroom dataset.



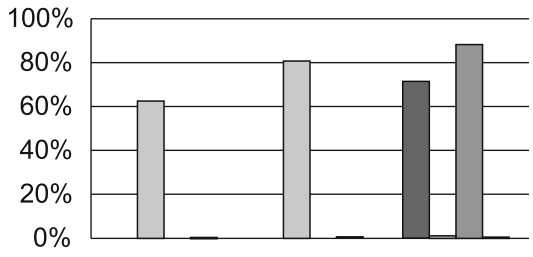
	ARRLR	MDSRRC	1.a
HF	0%	0%	57%
MC	17,04%	77,58%	37,34%
AP	1,73%	6,62%	90,31%
DISS	0,22%	2,32%	9,08%

Figure 8. Comprative evaluation with 7-rules on Mushroom dataset.



	ARRLR	MDSRRC	1.a	RRLR
HF	0%	0%	66%	0%
MC	25,02%	62,64%	0,17%	25,02%
AP	0,00%	0,02%	58,51%	0,00%
DISS	0,11%	0,13%	0,18%	0,11%

Figure 9. Comparative evaluation with 3-rules on Chess dataset.



	ARRLR	MDSRRC	1.a
HF	0%	0%	71%
MC	62,44%	80,73%	1,12%
AP	0,00%	0,00%	88,17%
DISS	0,28%	0,35%	0,20%

Figure 10. Comprative evaluation with 7-rules on Chess dataset.

cost is not considered as an advantage for 1.a, because it suffers from high rate of hiding failure.

Finally, it should be mentioned that the time used by the proposed algorithm to hide the sensitive rules is less than that of its counterparts. This is due to the fact that MDSRRC re-computes the sensitivity of transactions, as well as items, after each deletion. It also performs re-ordering of transactions. Moreover, for each case and each rule, 1.a finds and re-orders the transactions that partially support LHS. of a sensitive rule. For MDSRRC and 1.a, these procedures lead to more computational time when facing with more sensitive rules. As mentioned before, at the initial step of the proposed algorithm, the transactions are ordered based on the sensitivity of the items. However, the algorithm does not update the sensitivity of items and transactions during the hiding process and, consequently, no re-ordering is required.

5. Conclusion

This paper presents a heuristic algorithm named ARRLR which is developed to overcome the limitations of RRLR algorithm. The proposed algorithm is rule-oriented and enables us to hide the sensitive association rules with multiple RHS and LHS items. This approach was compared with MDSRRC, 1.a, RRLR algorithms. We used MDSRRC and 1.a and the proposed algorithm (ARRLR) to hide the three and seven sensitive rules on three databases. The experimental results are analyzed based on the following performance factors [4]: Hiding Failure (HF), Misses Cost (MC), Artificial Patterns (AP), and Dissimilarity (DISS). The results show that our algorithm outperforms MDSRRC, 1.a and RRLR in terms of the above performance factors. For future works, ARRLR algorithm can be improved to reduce its side effects on dense databases.

References

[1] K. Shah *et al*., "Association Rule Hiding by Heuristic Approach to Reduce Side Effects and Hide Multiple R. H. S. Items", *International Journal of Computer Applications*, vol. 45, no. 1, pp. 1–7, 2012.

[2] M. N. Dehkordi *et al*., "A Novel Method for Privacy Preserving in Association Rule Mining Based on Genetic Algorithms", *Journal of Software*, vol. 4, no. 6, pp. 555–562, 2009. http://dx.doi.org/10.4304/jsw.4.6.555-562

[3] H. Wang, "Quality Measurements for Association Rules Hiding", in *2013 AASRI Conference on Parallel and Distributed Computing and Systems*, Elsevier, 2013, vol. 5, pp. 228–234. http://dx.doi.org/10.1016/j.aasri.2013.10.083

[4] V. S. Verykios and A. Gkoulalas-Divanis, "A Survey of Association Rule Hiding Methods for Privacy", in *Privacy-Preserving Data Mining*, Springer, 2008, pp. 267–289. http://dx.doi.org/10.1007/978-0-387-70992-5_11

[5] A. Amiri, "Dare to Share: Protecting Sensitive Knowledge with Data Sanitization", *Decision Support Systems*, vol. 43, no. 1, pp. 181–191, 2007. http://dx.doi.org/10.1016/j.dss.2006.08.007

[6] S. R. Oliveira and O. R. Zaiane, "A Unified Framework for Protecting Sensitive Association Rules in Business Collaboration", *International Journal of Business Intelligence and Data Mining*, vol. 1, no. 3, pp. 247–287, 2006. http://dx.doi.org/10.1504/IJBIDM.2006.009135

[7] S. R. Oliveira and O. R. Zaiane, "Protecting Sensitive Knowledge by Data Sanitization", in *Data Mining, (ICDM 2003) 3rd IEEE Int. Conf.*, 2003, pp. 613–616. http://dx.doi.org/10.1109/ICDM.2003.1250990

[8] S. R. Oliveira and O. R. Zaiane, "Algorithms for Balancing Privacy and Knowledge Discovery in Association Rule Mining", in *Database Engineering and Applications Symp., 2003. Proc. of the 7th Int. Conf.*, 2003, pp. 54–63. http://dx.doi.org/10.1109/IDEAS.2003.1214911

[9] S. R. Oliveira and O. R. Zaiane, "Privacy Preserving Frequent Itemset Mining", in *Proc. of IEEE Int. Conf. Privacy, Security and Data Mining*, 2002, Australian Computer Society, Inc., vol. 14, pp. 43–54.

[10] G. V. Moustakides and V. S. Verykios, "A Max-Min Approach for Hiding Frequent Itemsets", *Data & Knowledge Engineering*, vol. 65, no. 1, pp. 75–89, 2008. http://dx.doi.org/10.1016/j.datak.2007.06.012

[11] X. Sun and S. Y. Philip, "Hiding Sensitive Frequent Itemsets by a Border-Based Approach", *Journal of Computing Science and Engineering*, vol. 1, no. 1, pp. 74–94, 2007. http://dx.doi.org/10.5626/JCSE.2007.1.1.074

[12] X. Sun and P. S. Yu, "A Border-Based Approach for Hiding Sensitive Frequent Itemsets", in *Data Mining, 5th IEEE Int. Conf. 2005*, pp. 74–82. http://dx.doi.org/10.1109/ICDM.2005.2

[13] A. Gkoulalas-Divanis and V. S. Verykios, "Exact Knowledge Hiding Through Database Ex-

tension", *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 5, pp. 699–713, 2009.
http://dx.doi.org/10.1109/TKDE.2008.199

[14] S. Menon and S. Sarkar, "Minimizing Information Loss and Preserving Privacy", *Management Science*, vol. 53, no. 1, pp. 101–116, 2007.
http://dx.doi.org/10.1287/mnsc.1060.0603

[15] S. Menon and S. Mukherjee, "Maximizing Accuracy of Shared Databases when Concealing Sensitive Patterns", *Information Systems Research*, vol. 16, no. 3, pp. 256–270, 2005.
http://dx.doi.org/10.1287/isre.1050.0056

[16] Y. Guo, "Reconstruction-Based Association Rule Hiding", in *Proc. of SIGMOD2007 Ph. D. Workshop on Innovative Database Research*, 2007, vol. 2007, pp. 51–56.

[17] H. Mannila and H. Toivonen, "Levelwise Search and Borders of Theories in Knowledge Discovery", *Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 241–258, 1997.

[18] Y. Saygin *et al*., "Privacy Preserving Association Rule Mining", in *Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems, (RIDE-2EC 2002) Proceedings of the 12th Int. Workshop*, 2002, pp. 151–158.
http://dx.doi.org/10.1109/RIDE.2002.995109

[19] Y. Saygin *et al*., "Using Unknowns to Prevent Discovery of Association Rules", *ACM SIGMOD Rec.*, vol. 30, no. 4, pp. 45–54, 2001.
http://dx.doi.org/10.1145/604264.604271

[20] M. Atallah *et al*., "Disclosure Limitation of Sensitive Rules", in *Knowledge and Data Engineering Exchange, (KDEX'99) Proc. of the Workshop*, 1999, pp. 45–52.
http://dx.doi.org/10.1109/KDEX.1999.836532

[21] V. S. Verykios *et al*., "Association Rule Hiding", *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 4, pp. 434–447, 2004.
http://dx.doi.org/10.1109/TKDE.2004.1269668

[22] C. N. Modi *et al*., "Maintaining Privacy and Data Quality in Privacy Preserving Association Rule Mining", in *Computing Communication and Networking Technologies (ICCCNT), 2010 Int. Conf.*, 2010, pp. 1–6.
http://dx.doi.org/10.1109/ICCCNT.2010.5592589

[23] N. H. Domadiya and U. P. Rao, "Hiding Sensitive Association Rules to Maintain Privacy and Data Quality in Database", in *Advance Computing Conf. (IACC), 2013 IEEE 3rd Int.*, 2013, pp. 1306–1310.
http://dx.doi.org/10.1109/IAdCC.2013.6514417

[24] T.-P. Hong *et al*., "Using TF-IDF to Hide Sensitive Itemsets", *Applied Intelligence*, vol. 38, no. 4, pp. 502–510, 2013.
http://dx.doi.org/10.1007/s10489-012-0377-5

[25] P. Cheng *et al*., "Privacy Preservation Through a Greedy Distortion-Based Rule-Hiding Method", *Applied Intelligence*, vol. 44, no. 2, pp. 295–306, 2016.
http://dx.doi.org/10.1007/s10489-015-0671-0

[26] B. Goethals, "Frequent Itemset Mining Implementations Repository", 2017.
Available: fimi.ua.ac.be/data

Received: July 2017
Revised: November 2017
Accepted: November 2017

Contact addresses:
Farsad Zamani Boroujeni
Department of Computer Engineering
Faculty of Engineering
Isfahan (Khorasgan) Branch
Islamic Azad University
Isfahan, Iran
e-mail: f.zamani@khuisf.ac.ir

Doryaneh Hossien Afshari
Faculty of Engineering
Isfahan (Khorasgan) Branch
Islamic Azad University
Isfahan, Iran
e-mail: d.h.afshari@gmail.com

Farsad Zamani Boroujeni received his PhD degree in computer science from the University Putra Malaysia in 2013. Currently, he is head of Computer Engineering Department and works as an assistant professor at the Faculty of Engineering, Isfahan (Khorasgan) Branch, Islamic Azad University, Isfahan, Iran. He has been an IEEE member since 2008. He has published several national and international journal articles and conference papers. He is collaborating with many reputed international journals and conferences as a reviewer. His areas of research interest include data mining, big data analytics and social networks.

Doryaneh Hossien Afshari received her BS and MSc degrees in computer engineering from the Islamic Azad University, Isfahan, Iran, in 2011 and 2014, respectively. Her main areas of research include privacy preserving in data mining, social networks, and recommender systems. Currently she is working as a researcher conducting research projects on privacy in utility mining and prediction in sequence mining.