Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 7, No. 1, February 2016), Pages: 67-87
www.jacr.iausari.ac.ir

Double Clustering Method in Hiding Association Rules

Zahra Kiani Abari✉, **Mohammad Naderi Dehkordi**

Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad, Isfahan, Iran

zahrakiani@sco.iaun.ac.ir; naderi@iaun.ac.ir

Received: 2015/03/24; Accepted: 2015/07/26

Abstract

Association rules are among important techniques in data mining which are used for extracting hidden patterns and knowledge in large volumes of data. Association rules help individuals and organizations take strategic decisions and improve their business processes. Extracted association rules from a database contain important and confidential information that if published, the privacy of individuals may be threatened. Therefore, the process of hiding sensitive association rules should be performed prior to sharing the database. This is done through changing the database transactions. These changes must be made in such a way that all sensitive association rules are hidden and a maximum number of non-sensitive association rules are extractable from the sanitized database. In fact, a balance is to be established between hiding the sensitive rules and extracting the non-sensitive rules. A new algorithm is presented in this paper to create a balance between preserving privacy and extracting knowledge. The items of sensitive rules are clustered in the proposed algorithm, in order to reduce changes. In fact, reduction of changes and clustering of rules are applied in order to reduce the side effects of the hiding process on non-sensitive rules.

Keywords: *Data Mining, Association Rules, Frequent Item-sets, Privacy Preserving Data Mining, Clustering.*

1. Introduction

Data mining is a powerful tool for analyzing and summarizing data [1]. Data mining can be used to extract the hidden information and knowledge in large volumes of data. Nowadays, data mining is widely used in marketing, medical and business analysis [2]. The knowledge extracted by data mining tools includes sensitive and important information that if published, the privacy of individuals and organizations may be threatened. Privacy preserving in data mining (PPDM) protects sensitive information against data mining algorithms [3]. PPDM algorithms create a sanitized database by changing the source database. The sanitized database is created by removing unrequired items from some database transactions or adding some items to a number of them. These changes to the source database must be made in such a way for the sensitive data not to be able to be extracted from the sanitized database. The main challenge in PPDM is the impact of hiding sensitive information on discovering the non-sensitive information. In fact, making these changes to the source database for creating a sanitized database may result in hiding some non-sensitive information, as well [4].

This paper focuses on preserving privacy in association rule mining. Association rules mining is one of the most widely used techniques in data mining, which is used to extract dependency patterns from the database [5]. Association rules extracted from the database contain sensitive rules that the database owner does not wish to disclose. Association rule hiding algorithms attempt to prevent the extraction of sensitive rules by association rule mining algorithms [6]. The changes made to the source database by these hiding algorithms affect the discovery of non-sensitive rules and causes some of them to become non-extractable after the hiding process [7]. An algorithm for hiding sensitive association rules is presented in this paper. The proposed algorithm is able to hide any sensitive rules with no restriction on their forms. In order to reduce side effects such as "Missed Cost", the proposed algorithm clusters the sensitive rules. In fact, two clustering operations are carried out on the sensitive rules based on common RHS and LHS, after which the best cluster is selected for the hiding process. Either the reduce support or the reduce confidence technique would be used, if the cluster, which is based on the common RHS was selected, while the reduce confidence technique would be used if the cluster which is based on the common LHS was chosen. Taking the structure of the sensitive rules into account as well as choosing the most suitable technique, while running the algorithm, helps to reduce both the number of modifications to the database and the number of missed costs. In the proposed algorithm, the number of required changes for the hiding process is calculated prior to starting it during which the disclosure threshold ψ is used, to establish a balance between hiding the sensitive rules and discovering the non-sensitive ones. ψ has a user-specified value between .1 and 1. The closer ψ is to .1, the more extractable will be the non-sensitive rules from the sanitized database after the hiding process. Reducing the changes to the source database makes the more non-sensitive rules to become extractable from the sanitized. Similarly, as the value of ψ approaches 1, more changes are made to the database, which in turn prevents the extraction of sensitive association rules. In fact, defining a disclosure threshold of ψ gives the database owner the right to choose and enables him to make the desired balance between hiding sensitive rules and mining non-sensitive rules. Besides, this calculation leads into a significant reduction in CPU time of the hiding process.

The rest of this paper is organized as follows: Section 2 examines the related work done on the subject. In section 3, the important concepts in hiding association rules are introduced. The proposed algorithm is presented in Section 4. Section 5 deals with evaluating the performance of the proposed algorithm, and finally, the conclusions are presented in Section 6.

2. Related works

Information sharing is often beneficial for database owners. However, in some cases, it may disclose personal information. Privacy preserving techniques in data mining prevents unauthorized access to the information. Our focus in this paper is to hide sensitive association rules. In this section, algorithms that have been introduced in recent years to hide the association rules will be evaluated.

In 2001, Saygm et al. proposed two algorithms to hide sensitive association rules. The first one focuses on hiding the rules by reducing the minimum support of the item-

sets that generated these rules. The second one focuses on reducing the minimum confidence of the rules [8].

In 2002, Oliveiraet et al. proposed four algorithms, namely Naïve, MinFIA, MaxFIA and IGA to hide sensitive association rules. Each algorithm selects the sensitive transactions to be sanitized, based on their degree of conflict. Naïve Algorithm removes all items of selected transaction except for the item with the highest frequency in the database. MinFIA algorithm selects the item with the smallest support in the pattern as a victim item and removes the victim item from the sensitive transactions. Unlike MinFIA, MaxFIA algorithm selects the item with the maximum support in the restrictive pattern as a victim item. IGA algorithm groups restricted patterns in patterns sharing the same item- sets so that all sensitive patterns in the group will be hidden in one step [9].

In 2004, Verykios et al. presented three algorithms 1.a, 1.b and 2.a, for hiding sensitive association rules. Algorithm "1.a" hides association rules by increasing the support of the rule's antecedent until the rule confidence decreases below the minimum confidence threshold. 1.b algorithm hides sensitive rules by decreasing the consequent frequency until either the confidence or the support of the rule is below the threshold. Algorithm "2.a" decreases the support of the sensitive rules until either their confidence is below the minimum confidence threshold or their support is below the minimum support threshold. Large number of new frequent item-sets is introduced in 1.a algorithm and, therefore, an increasing number of new rules are generated. Algorithms 1.b and 2.a affect the number of non-sensitive rules in database due to removal of items from the transaction [10].

In 2005, Wang et al. proposed ISL and DSR algorithms to hide sensitive association rules. ISL with increasing support of rules' LHS, reduces confidence under the threshold, and thus the sensitive association rules will be hidden. DSR decreases the whole rule's support and confidence below the threshold to hide sensitive association rules. Hiding the sensitive items and the arrangement of database transactions affects the result in the operations of both algorithms. DSR has no hiding failure; notwithstanding, ISL will fail if there is no suitable transactions to be added [11].

In 2007, Wang et al. proposed two algorithms, DCIS (Decrease Confidence by Decrease Support) and DCDS (Decrease Confidence by Decrease Support) to automatically hide collaborative recommendation association rules without pre-mining and selection of hidden rules. DCIS algorithms try to increase the support of left hand side of the rule and DCDS algorithms try to decrease the support of the right hand side of the rule [12].

In 2008, Weng et al. proposed FHSAR (Fast Hiding Sensitive Association Rules) to hide association rules for fast hiding sensitive association rules. This algorithm can completely hide given sensitive association rules by scanning database only once, which significantly reduced the execution time. In this algorithm, the correlations between the sensitive association rules and each transaction in the original database, which can effectively select the proper item to modify are analyzed [13].

In 2010, Modi et al. introduced DSRRC algorithm. In this algorithm, sensitive rules are clustered based on similar RHS, and then hiding operation are performed. Hiding association rules reduces both amounts of changes in the database and the side effects, by using clusters instead of single rules collectively. DSRRC algorithm sorts the database after each change, which increases the hiding process time. This algorithm depends on the database orientation and the result of the outcome will vary by any modifications in the database [14].

In 2011, Kumar Jain et al. proposed a heuristic algorithm for hiding association rules that are based on ISL and DSR. It operates based on both ISL and DSR techniques, by which not only does increase the LHS support, but also the total support will be decreased. Although, this algorithm has no failure in hiding, the database will be changed a lot due to simultaneous reduction of rules' support and confidence [15].

In 2012, Komal Shah et al. proposed improved algorithms, called ADSRRC and RRLR, to reform DSRRC limitations. Similar to DSSRC, ADSRRC tries to cluster sensitive rules based on similar RHS. In this algorithm at first, the sensitivity of the transactions is calculated, and then they will be sorted in descending order. For this reason, arrangements of transactions have no effect on algorithm result. RRLR has been designed to hide various association rules with different RHS. In this algorithm, the process of concealment is done by reducing the confidence of sensitive rules below the threshold. Since these two algorithms do two sorting operations, they perform quicker, in term of runtime, than DSRRC [16].

In 2012, Nikunj et al. proposed MDSRRC to hide association rules. MDSRRC can hide rules with multiple RHS and LHS. At first, the sensitivity of items in rules' RHS is calculated and then the most sensitive item will be selected to be deleted. MDSRRC, in comparison with DSRRC, reduces database modification and side effects, by deleting the effective candidate item [17].

In 2012, Jain et al. introduced an algorithm, which hides sensitive association rules without altering the support of frequent item-sets. It has been tried in this algorithm to use a new concept named Representative rule, in which all sensitive rules can be inferred by the help of the Representative rule and without any access to the main database. This algorithm changes the position of items, instead of removing any items in transactions, to hide association rules. Thus, it causes no modification in frequent item-sets' support, size of database, and finally it hides the maximum number of sensitive association rules by fewer changes in the database. This is due to the existence of suitable transactions in the database to alter the position of sensitive items; otherwise hiding process will be failed [18].

In 2014, Wei Lin et al. introduced the cpGA5DT algorithm for hiding the sensitive item set which was developed based on genetic algorithms. In this algorithm, a fitness function with three adjustable weights is designed to find the appropriate transaction. The algorithm deletes the transaction that have been identified by the fitness function in order to reduce the support of the sensitive item set. What the fitness function tries to achieve is to select those transactions with the least amount of side effects upon removal [19].

In 2014, Jadav et al. introduced the RHID algorithm for hiding sensitive association rules in incremental databases, which is based on the MDSRRC algorithm. The distinction between these two algorithms, however, is that the MDSRRC algorithm is used in static databases, while the RHID algorithm is able to hide the sensitive rules, in incremental databases. In order to hide sensitive association rules, the RHID algorithm makes use of a template table created by the MDSRRC algorithm [20].

In 2015, Fouladfar et al. proposed FHA algorithm in order to hide the association rules. Overlapping the sensitive rules, FHA algorithm hides them through only one time scan of the database. This algorithm changes the position of sensitive items in order to hide the sensitive rules. Since no item is removed from the database, the support of sensitive rules remain unchanged and the rules are hidden by only reducing the confidence. This act of not removing any sensitive items from the database results into a decrease in the number of missed costs [21].

3. Basic Concepts

Transactional databases, association rules, concepts related to hiding association rules and side effects of the hiding process are briefly introduced in this section.

3.1 Transactional Database

A transactional database consists of a series of transactions, where each transaction t is represented as t= [TID, list_of_element]. TID is a unique identifier for each transaction and the "list_of_element" is the set of constituent items of the transaction [22].

3.2 Association Rules

Association rule mining is one of the most widely used tools in data mining, which is used to extract dependency patterns from a database. An association rule shows the connection between seemingly unrelated items in the database. The rule consists of two parts: left-hand side (the antecedent) and right- hand side (the consequent). An association rule is defined as follows: Let I={i1,i2,…,in} be a set of items, D be a database of transactions, in which each transaction $t \subset I$ Then, an association rule is defined as: $X \longrightarrow Y$ where $X \subseteq I$, $Y \subseteq I$and $X \cap Y = \varphi$ [23]. The support for any rule in database is calculated using formula (1):

$$Support(X \to Y) = \frac{|X \cup Y|}{|N|} \quad (1)$$

where $|X \cup Y|$ is the number of transactions that contain X and Y, and $|N|$ is the whole number of transactions in the database. Confidence for each rule can be obtained through formula (2):

$$Confidence(X \to Y) = \frac{|X \cup Y|}{|X|} \quad (2)$$

where $|X|$ is the number of transactions that contain X. The support indicates the frequency of a rule in the database and the confidence threshold represents the strength of the connection between the items in the rule. Association rule mining consists of two steps: First, using algorithms such as Apriori [24] Frequent item-set (a set of items whose support is higher than the minimum support threshold defined by user) for mining association rules is extracted from the large volume of data. In the second step, the strong association rules (those whose confidence is higher than the minimum confidence threshold specified by user) are extracted from the frequent k-item-set.

3.3 Privacy Preserving in Data Mining

Association rules extracted from the database contain sensitive rules that the database owner does not wish to disclose and as a result must be hidden prior to publishing the database [4]. For example, consider two retailers, Ali and Reza. Ali is an experienced retailer and Reza is a novice one. As a novice retailer, Reza wants to identify the bestselling products. Therefore, he needs to examine the association rules extracted from Ali's store. Suppose each customer also buys tea from Ali's store, when buying milk. For our example, this rule is considered as a sensitive rule for Ali. If this rule is placed at Reza's disposal, he can offer a discount for buying milk and tea, and increase his sales. In this case, Ali's sale of milk and tea gradually drops, and Reza monopolizes the market. The problem of hiding sensitive association rules can involve the following: Transactional database D , minimum support threshold "MST", minimum confidence threshold "MCT", a set of association rules R extracted from D, a set of sensitive rules $R_{Sen} \subseteq R$ that must be hidden, the database D' which is created in such a way that $R_{non-Sen} = R - R_{Sen}$ rules can be extracted from it [25]. The problem of hiding sensitive association rules is shown in Figures 1 to 3.

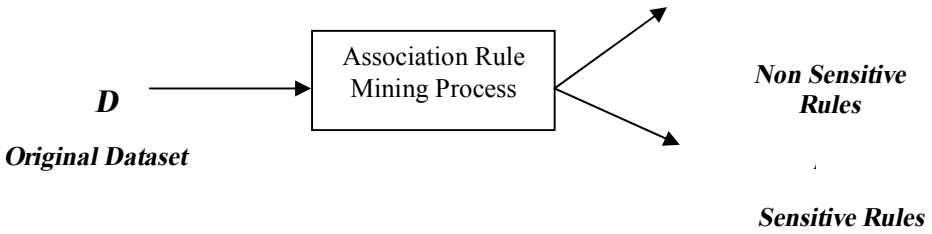

Figure 1. Association Rule Mining process input and outputs

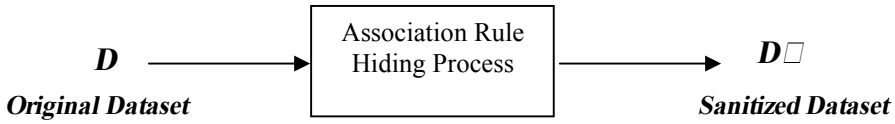

Figure 2. Association rule hiding process input and outputs

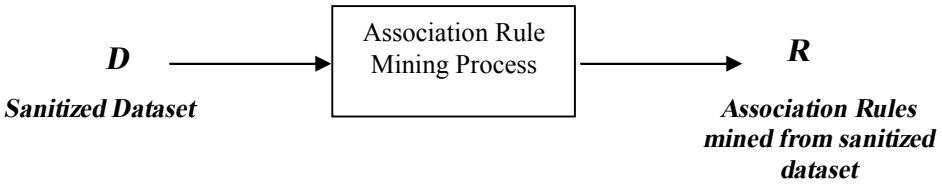

Figure 3. Association Rule Mining after Association Rule Hiding

In recent years, many algorithms have been introduced for hiding sensitive association rules, which can be categorized in five groups: Heuristic based approaches, border based approaches, exact approaches, reconstruction based approaches and cryptography based approaches [26]. These available methods are briefly considered, as follows:

Heuristic Based Approaches: Heuristic algorithms are fast and scalable, but do not guarantee to find the best solution. However, they usually offer close to the best solution, in the shortest time [27]. Heuristic algorithms are categorized into two groups: "Data distortion" and "data blocking" groups. In data distortion, some items are added to or deleted from transactions. Insertion and removal of these items not only cause the support and confidence of the sensitive rules to drop below the corresponding thresholds, but also hide the association rules [14]. 1a, 1b [15], ISL, DSR [11], DSC [28], FHSAR [13], DSA [29], ADSRRC [16] and DSRRC [14] algorithms use the data distortion technique for hiding the association rules. Instead of distorting the data, some of the values of the database are replaced by an unknown value represented by "?" symbol, in the blocking technique, This replacement makes it difficult for unauthorized individuals to explore the data behind "?" symbol [8]. In [30] , blocking based algorithms for hiding association rules have been introduced.

Border Based Approaches: A pre-processing operation is performed in border based approaches on the sensitive rules, prior to starting the hiding process and then, a small subset of the sensitive rules is selected as the input for the hiding process. Pruning the sensitive rules maintains the quality of the database and reduces the side effects of the hiding process [31]. In [32] and [33], border based techniques are used to hide the association rules.

Exact Techniques: Exact approaches formulate the problem of hiding the association rules as a constraint satisfaction problem (CSP), and solve it using binary integer programming (BIP). The best solution compared to other available techniques is presented through satisfying all constraints in these techniques. The time complexity in Exact Approaches is high, due to using CSP [34]. In [34], Exact Approaches are used for hiding the association rules.

Data Reconstruction Based Techniques: In data reconstruction based approaches, the hiding process is not performed on the source database. In fact, a new database is created by extracting the sensitive characteristics from the source database, and then the hiding process is performed on this new one. These techniques have fewer side effects compared to heuristic based techniques [26]. An algorithm is presented in [35] for hiding the association rules, in which FP-tree-based data reconstruction is used to reduce the side effects of the hiding process.

Cryptography Based Techniques: Instead of distortion, the database is encrypted in cryptography based approaches. These techniques are used in multiparty computation [26]. In [36], a cryptography based algorithm is introduced for hiding association rules.

3.4 Side effects of the process regarding hiding association rules

Hiding association rules by any of the techniques presented in the previous section are somehow associated with some side effects, which are examined in this section.

A. **Hiding failure**: Hiding failure occurs when some of the sensitive rules can be extracted from the sanitized database [22]. Hiding failure is calculated using formula (3):

$$HF = \frac{\#R_R(D')}{\#R_R(D)} \tag{3}$$

where $\#R_R(x)$ is the number of extracted association rules from database X.

B. **Missed Cost**: This side effect occurs when the hiding process would hide some non-sensitive rules [22]. This side effect is calculated by formula (4):

$$MC = \frac{\#\sim R_R(D) - \#\sim R_R(D')}{\#\sim R_R(D)} \tag{4}$$

where $\#\sim R_R(X)$ is the number of non-sensitive rules extracted from database X.

C. **Artificial Pattern**: This side effect occurs when the artificial rules which are not supported by the source database are extracted from the sanitized database [22]. Artificial pattern is computed by formula (5):

$$AP = \frac{|R'| - |R \cap R'|}{|R'|} \tag{5}$$

where $|X|$ represents the cardinality of X. The side effects of the hiding process are shown in figure 4.

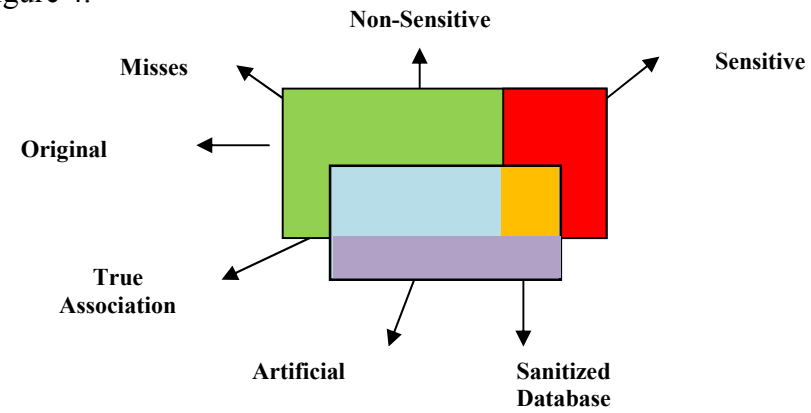

Figure 4. Side effect of association rule hiding

4. The proposed algorithm

A heuristic algorithm called DCMHAR (Double Clustering Method in Hiding Association Rules) is presented in this paper, for hiding association rules. This algorithm can hide the association rules with multiple RHS and LHS, with no restriction on the form of input rules. Either the support reduction or the confidence reduction or both are used in this algorithm. Choosing the appropriate technique for hiding the association rules is done when running the algorithm and is based on the structure of sensitive rules. In order to reduce the changes to the database, the items of sensitive rules are clustered according to their presence in RHS or LHS, and they then are hidden as a cluster. If the items in RHS group are selected, the hiding process is done by reducing the support and confidence, and if those in LHS are selected, the hiding process is done by reducing confidence threshold. A disclosure threshold of ψ with a value between .1 and 1 is used to make a balance between hiding the sensitive rules and extracting the insensitive ones. The value for ψ is determined by the database owner. As the value of ψ gets closer to .1, the amount of changes to the database and the number of missed costs decrease. However, as this value approaches 1, the distance of confidence of sensitive rules from the threshold increases and the possibility of extracting these rules are reduced. The proposed algorithm consists of three parts, as follows:

In the first phase, the items of sensitive rules are clustered according to their presence in RHS or LHS. Then, RHS_{Set} and LHS_{Set} are created, which consist of the items in the RHS and LHS of sensitive rules, respectively.

In the second phase, the smaller of the two RHS_{Set} and LHS_{Set} is selected and the hiding process is performed based on that choice. With regard to the chosen set, the number of changes necessary to hide the sensitive rules is calculated. This calculation is as follows:

If RHS_{Set} is selected for the hiding process, the number of required removal operations can be obtained using formulas (6) and (7):

$$TR(\sup) = |X \cup Y| - \left(|X \cup Y| * \left(|N| * (MST - \psi) / |X \cup Y|\right)\right) \quad (6)$$

$$TR(Con) = |X \cup Y| - \left(|X \cup Y| * \left(|X| * (MCT - \psi) / |X \cup Y|\right)\right) \quad (7)$$

In the above formulas, the rule with maximum confidence in each category is used, where |X∪Y| is the number of transactions, which completely support the rule, |N| is the whole number of database transactions and |X| is the number of transactions that support the LHS of the rule. The required number of removal operations is equal to the smallest value calculated by formulas (6) and (7).

If LHS_{Set} is selected for the hiding process, the number of required insertion operations will be calculated by formula (8):

$$TI = \left(|X \cup Y| - \left((MCT - \psi) * |X|\right)\right) / (MCT - \psi) \quad (8)$$

In the above formula, the rule with maximum confidence in each category is used, where |X∪Y| is the number of transactions that completely support the rule and |X| is the number of transactions that support the LHS of the rule.

In the third phrase, the hiding process is performed. If RHS_{Set} is selected, its items will be removed from corresponding transactions. And, if LHS_{Set} set is selected, its items will be inserted in transactions.

DCMHAR algorithm

```
Input: Original database D, Sensitive rules SR, MST, MCT, ψ
Output: Sanitize database D'
begin
      SI← Find sensitive items
      // Step 1: clustering sensitive item
      foreach sensitive item si_i ∈ SI do
        if si_i present in the RHS of sensitive rules then
              RHS_Set ←RHS_Set+ si_i
        end if
        if si_i present in the LHS of sensitive rules then
              LHS_Set ← LHS_Set+ si_i
          end if
     end foreach
       // Step 2: choose the best cluster
       if RHS_Set < LHS_Set then
            SS← RHS_Set
            flag ←true
           foreach ss_i ∈ SS do
                  Calculate TR_Supssi
                Calculate TR_Conssi
                  if TR_Supssi= < TR_Conssi then
                       TO_i← TR_Supssi
                  else
                       TO_i← TR_Conssi
                  end if
           end foreach
       else
            SS← LHS_Set
            flag ←false
            foreach ss_i ∈ SS do
                Calculate TI_ssi
                TO_i← TI_ssi
            end foreach
      end if
      // Step 3: hiding process
      foreach ss_i ∈ SS do
                 if flag=true then
                       find the  heavy transaction and sort
                    those in decreasing order of their
                    weight
                      for k=0to k=To_i
                        Remove ss_i from heavy transactions
                    end for
                else
                       find the light transaction and sort
                       those in ascending order of their
                       length
                       For k=0to k=TO_i
                         Insert ss_i into light transaction
                      end for
              end if
       end foreach
end begin
```

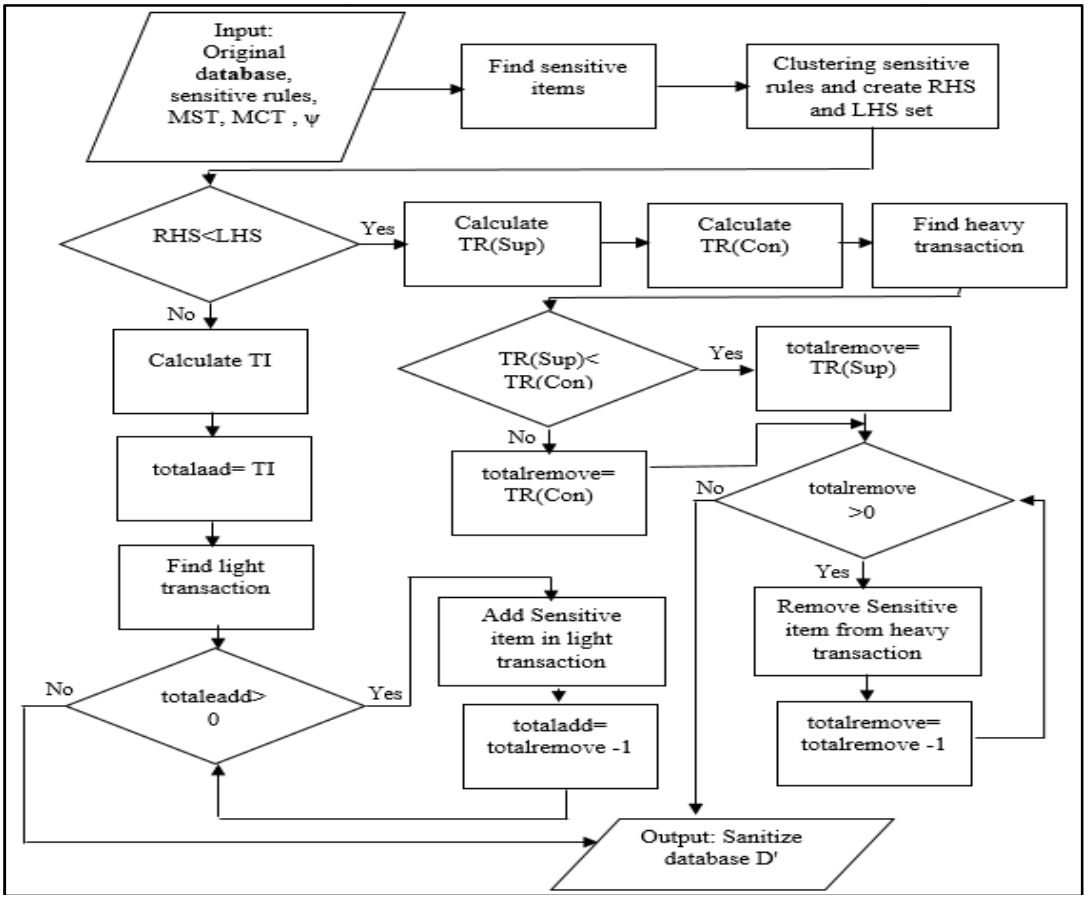

Figure 5. Flowchart of proposed DCMHAR algorithm

5. Evaluation the proposed algorithm

Some tests were designed and carried out in order to evaluate the proposed algorithm, which is presented in this section. Different forms of input rules have been used in each set of experiments, in order to demonstrate the performance of the proposed algorithm in hiding various forms of association rules. In addition to this, the real Chess database is used in all experiments. Chess database has been used which contains 75 items, 3196 transactions, and the average transaction length is 37. To demonstrate the effectiveness of the proposed algorithm, such criteria as hiding failure, missed cost, artificial pattern and time complexity are examined and the results from the proposed algorithm are compared with those of MDSRRC [17] and ADSRRC [16] algorithms.

In the first set of experiments, the sensitive rules are selected in a single→single form. The result of the experiments are depicted in Figures 6 to 9. Each of the three algorithms are able to hide the sensitive rules in this form, as shown in Figure 6, and they do not suffer from hiding failure. As it can be seen in Figure 7, illustrating the number of missing rules, the number of missed costs in the proposed algorithm has decreased by 9.15% and 9.54%, as compared to ADSRRC and MDSRRC algorithms, respectively. Figure 8 shows the number of artificial rules, which is equal 0% in each of the three algorithms. Finally, Figure 9 shows the CPU time of the hiding process, which

has reduced by 431.5 and 7096.33 milliseconds compared with ADSRRC and MDSRRC algorithms, respectively.

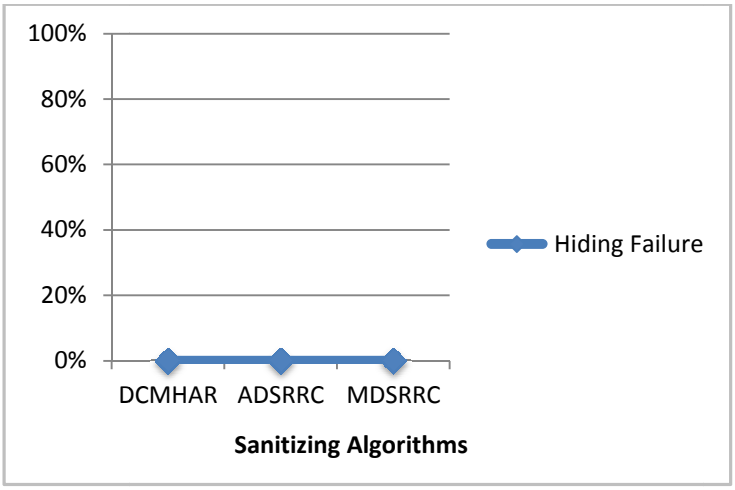

Figure 6. Failure in hiding rules in single→single form

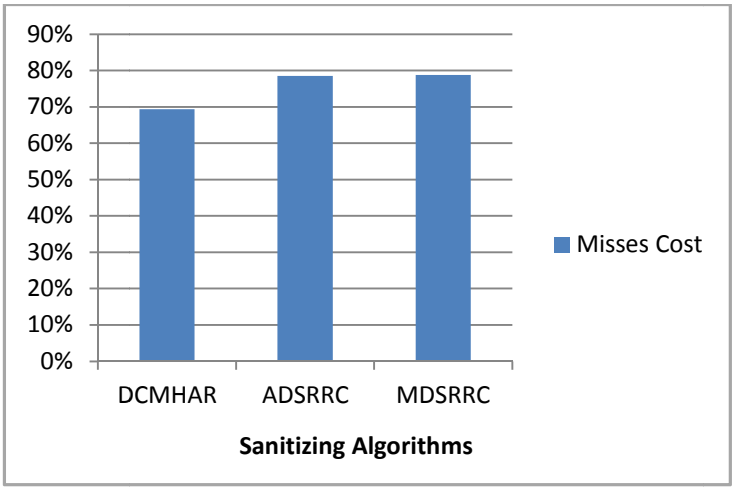

Figure 7. Missed costs in hiding rules in single→single form

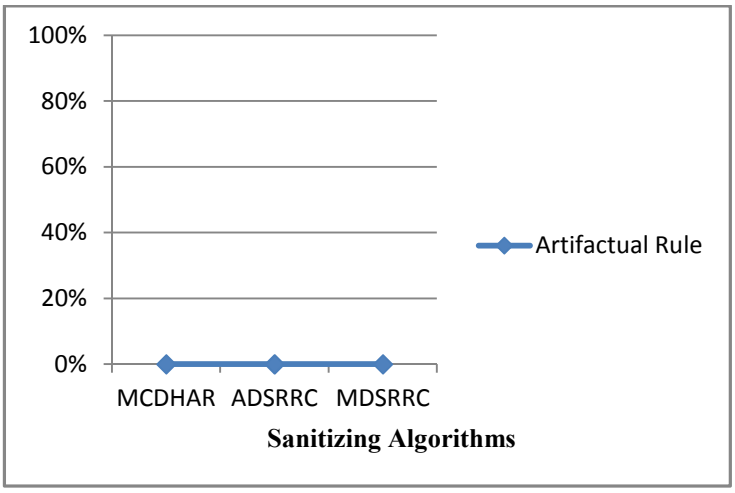

Figure 8. Artificial rules in hiding rules in single→single form

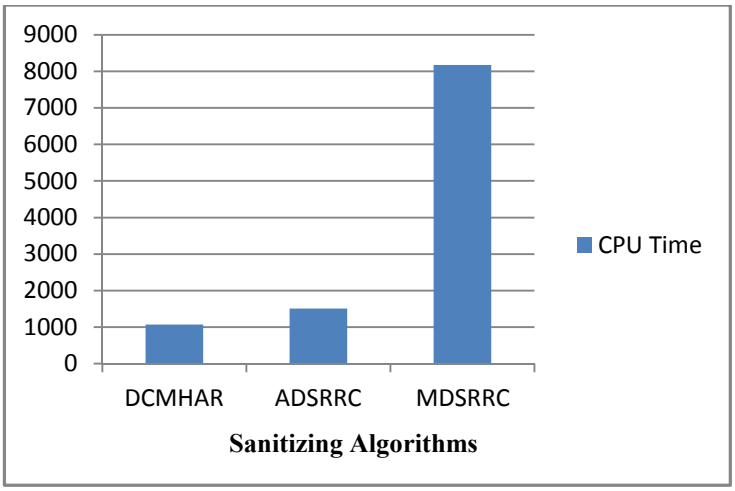

Figure 9. CPU time in hiding rules in single→single form

In the second set of experiments, however, the sensitive rules are selected in a single→ multiple form. The result of this set of experiments is illustrated in Figures 10 to 13. According to Figure 10, the ADSRRC algorithm is unable to hide this form of sensitive rules and the hiding failure rate for this algorithm is as much as 100%, which in turn makes it impossible to investigate the other criteria. The number of missed costs is depicted in Figure 11, which illustrates that this factor is decreased by 35.64% in the proposed algorithm, as compared to the MDSRRC algorithm. Figure 12 shows the number of artificial rules, which is equal 0% in each of the two algorithms. Figure 13, as the last figure in this set of experiments, shows the time complexity of the hiding process, which is 1406 milliseconds less than that of the MDSRRC algorithm.

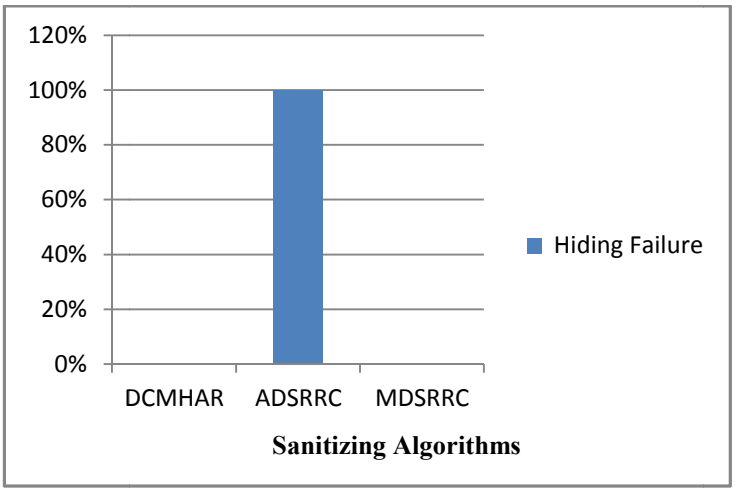

Figure 10. Failure in hiding rules in single→multiple form

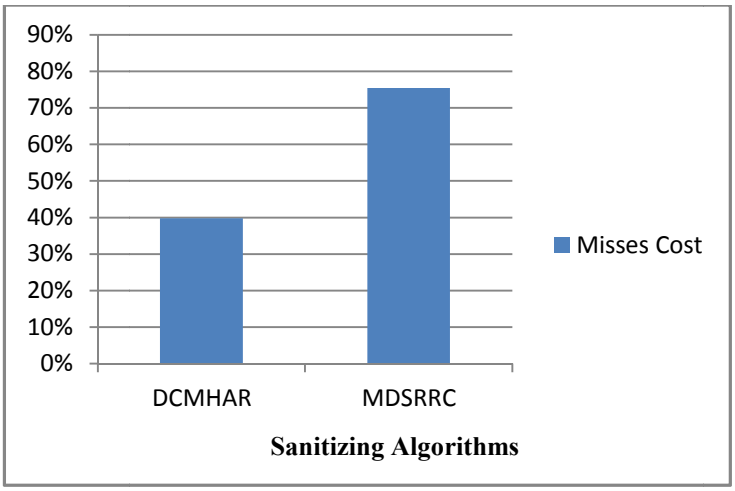

Figure 11. Missed costs in hiding rules in single→multiple form

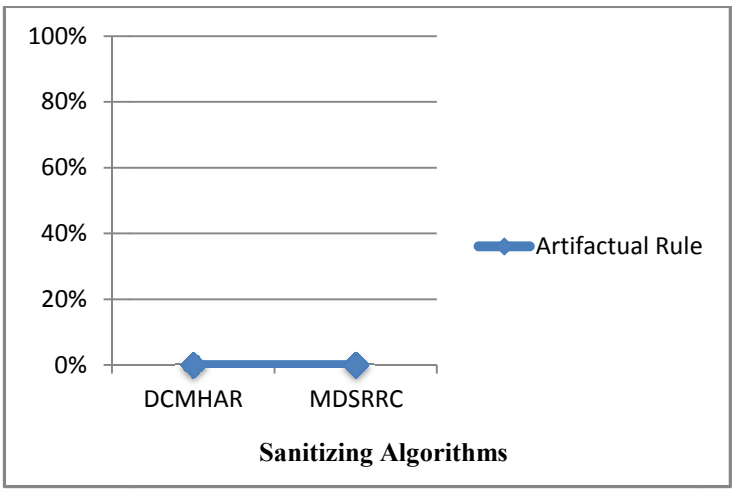

Figure 12. Artificial rules in hiding rules in single→multiple form

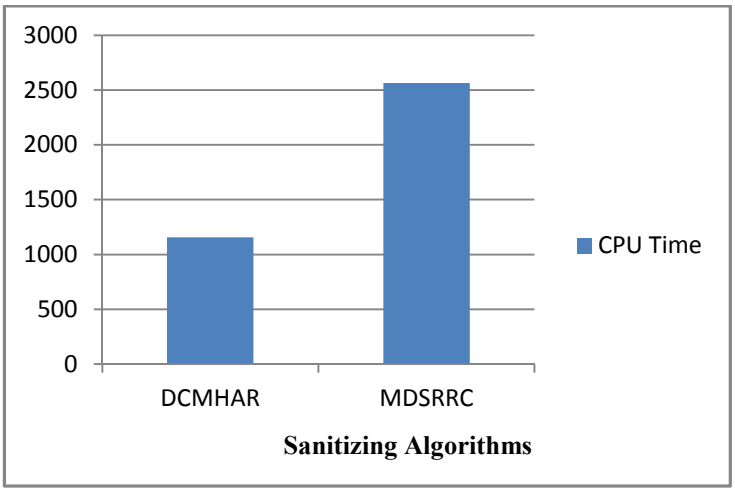

Figure 13. CPU time in hiding rules in single→multiple form

In the third set of experiments, the sensitive rules have been chosen in a multiple→single form, and the results are depicted in Figure 14 through Figure 17. As it is illustrated in Figure 14, ADSRRC algorithm is again unable to hide this form of sensitive rules and the hiding failure rate for this algorithm is as much as 100%, which in turn makes it impossible to investigate the other criteria. Figure 15 shows the number of missed costs according to which this criteria is reduced by 9.86% in the proposed algorithm compared to that of MDSRRC algorithm. Figure 16 illustrates the number of artificial rules, which is equal to 2% in the proposed algorithm and 0% in the MDSRRC algorithm. Finally, Figure 17 shows the time complexity of the hiding process, which is 3968.33 milliseconds less than that of the MDSRRC algorithm.

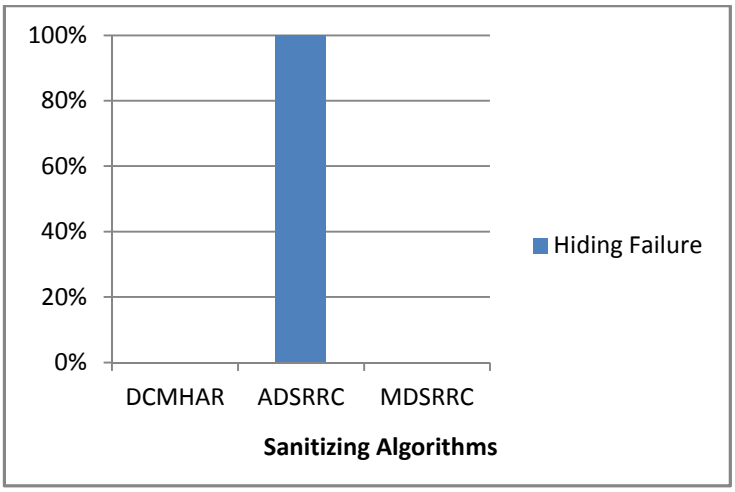

Figure 14. Failure in hiding rules in multiple→single form

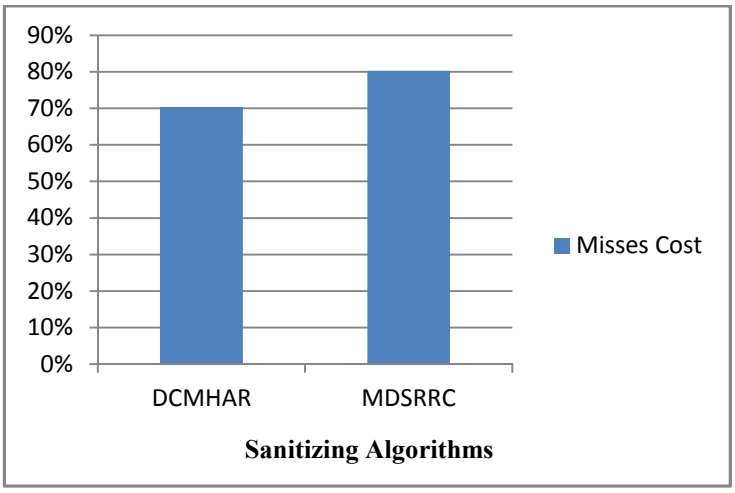

Figure 15. Missed costs in hiding rules in multiple→single form

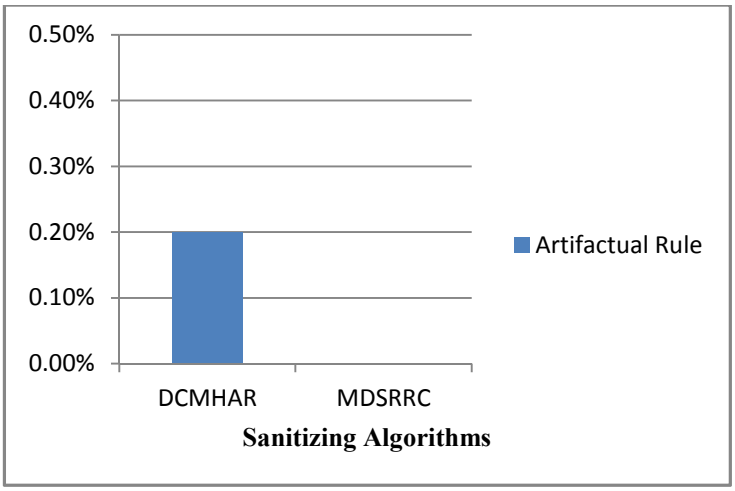

Figure 16. Artificial rules in hiding rules in multiple→single form

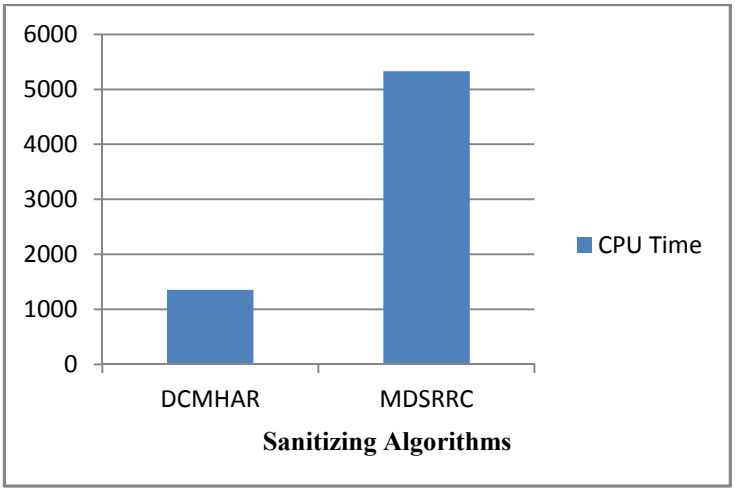

Figure 17. CPU time in hiding rules in multiple→single form

In the last set of experiments, the sensitive rules have been selected in a multiple→multiple form, and the results are shown in figures 18, 19, 20 and 21. As it is illustrated in Figure 18, ADSRRC algorithm is once more unable to hide this form of sensitive rules and the hiding failure rate for this algorithm is as much as 100%. This means that investigating the other criteria for this algorithm is impossible. Figure 19 shows the number of missed costs. Clearly, this criterion has experienced a reduction of 9.86% in the proposed algorithm compared to MDSRRC algorithm. Figure 20 indicates the number of artificial rules, which is equal to %0 for both algorithms. Figure 21 shows the time complexity of the hiding process, which has undergone a 3968.33-millisecond decrease in the proposed algorithm, compared to MDSRRC algorithm.

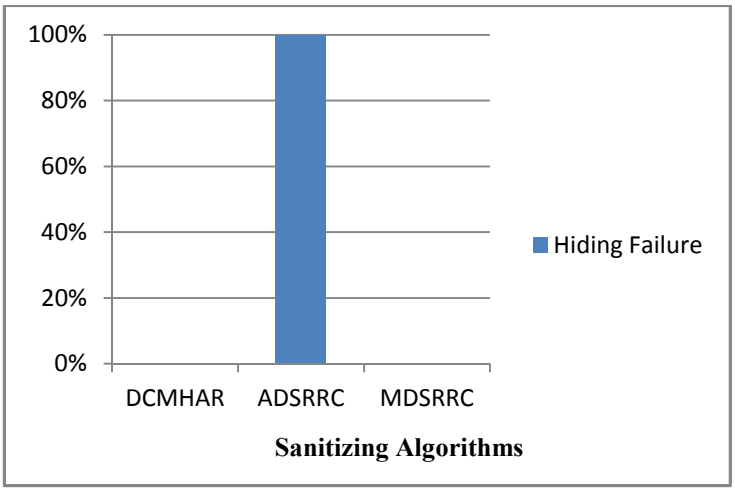

Figure 18. Failure in hiding rules in multiple→multiple form

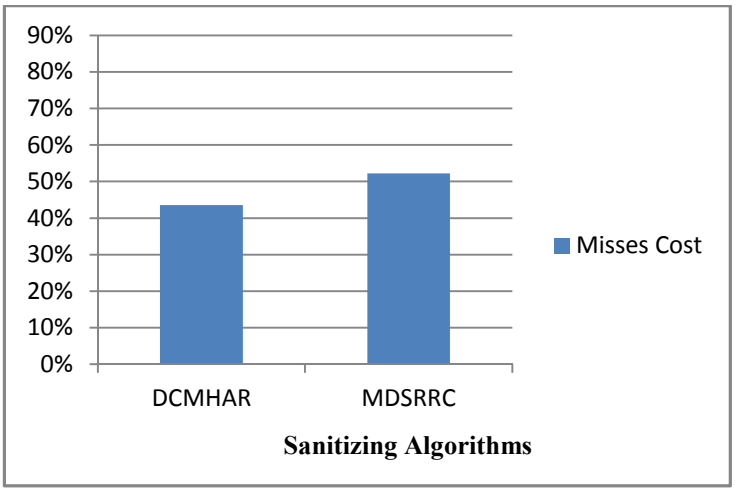

Figure 19. Missed costs in hiding rules in multiple→multiple form

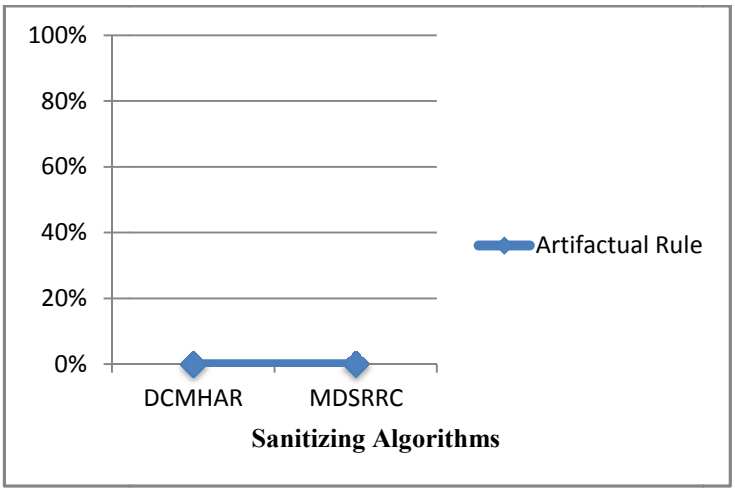

Figure 20. Artificial rules in hiding rules in multiple→multiple form

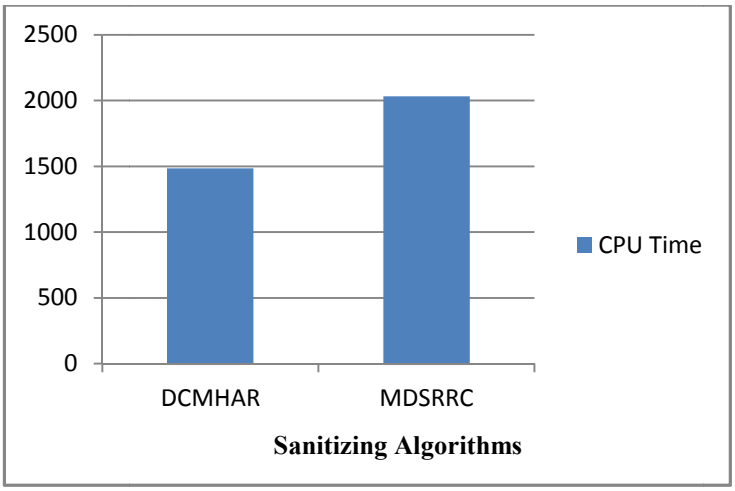

Figure 21. CPU time in hiding rules in multiple→multiple form

Finally, some experiments are designed and conducted to show the impact of the disclosure threshold on the number of missed costs and the non-sensitive rules, which can be extracted from the database. In each experiment, the disclosure threshold has been changed. Figure 22 shows the impact of the disclosure threshold ψ on the numbers of missed costs and the non-sensitive rules, which are extractable from the database. According to figure 22, as the user-specified value for ψ gets closer is to .1, more insensitive rules can be extracted from the database and the number of missed costs reduces. In fact, setting a small value for ψ causes the support and confidence of the sensitive rules to be hidden by a smaller distance from threshold. This, in turn, reduces the amount of changes made to the source database. As the amount of changes to the source database reduces, the number of missed costs decreases, and more rules that are non-sensitive may be extracted from the sanitized database. Similarly, as the user-defined value for ψ approaches 1, the support and confidence of sensitive rules are hidden by a larger distance from threshold. As the support and confidence of the sensitive rules get farther away from threshold, the number of changes made to the database increases, and so the number of missed costs.

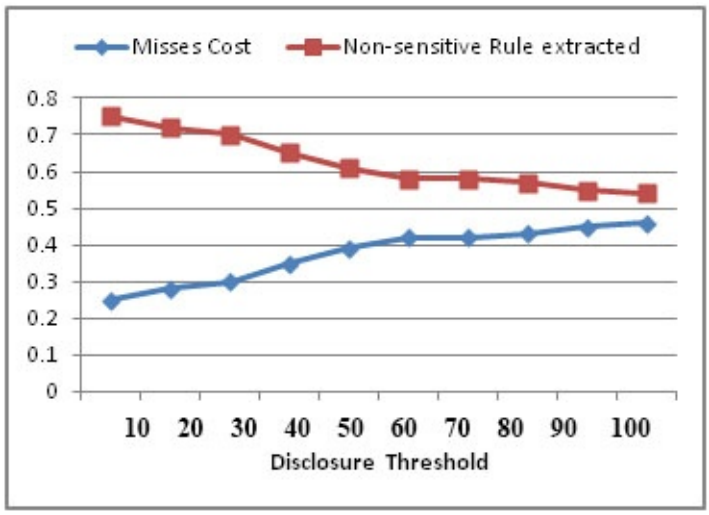

Figure 22. Effect of ψ on misses cost and non-sensitive rules

According to the experiments, the proposed algorithm does not suffer from hiding failure and is able to hide various forms of association rules. Clustering the sensitive rules and clustered hiding of the rules results in a minimum amount of change in the database, this in turn leads into a decrease in the number of the lost rules. The proposed algorithm chooses the proper technique for the hiding process at runtime, taking the structure of the sensitive rules into account. If reducing the support technique is used, no artificial rules will be created after the hiding process. However, if the reduction in confidence technique is applied, there will be a possibility of generating artificial rules, due to inserting sensitive items in the database transactions. As compared to ADSRRC and MDSRRC algorithms, it can be seen that the CPU time of the proposed algorithm in all sets of experiments, enjoyed a significant improvement, which was a result of calculating the number of required changes before actually starting the process.

6. Conclusion

An algorithm has been presented in this paper for association rule hiding. The proposed algorithm is able to hide any association rule with no restriction on the form of association rules. DCMHAR Algorithm does not suffer from hiding failure side effect and is able to hide the sensitive rules in different conditions of a database. In the proposed algorithm, the items of sensitive rules are clustered according to their presence in RHS or LHS sensitive rules and the smallest cluster is selected for the hiding process. Given the selected cluster, the suitable technique for hiding the sensitive rules is determined smartly, while running the algorithm and the rules are hidden using the selected technique. Additionally, by defining the disclosure threshold of ψ, the number of extractable non-sensitive rules from the sanitized database and the number of missed costs caused by the user are controlled. The results of experiments indicate that the sensitive rules are hidden with fewer changes to the database in DCMHAR algorithm, as compared to other algorithms such as ADSRRC and MDSRRC. Hiding sensitive rules with minimum changes has a significant impact on reducing the number of missed

costs in DCMHAR algorithm. Additionally, calculating the number of required changes prior to the hiding process helps reduce the CPU time. It may be possible to reduce the number of generated artificial rules in future via a new technique for inserting items during the selection phase of LHS_{Set}.

References

[1] C. Rygielski, J.C. Wang, D.C. Yen, "Data mining techniques for customer relationship management," Technology in society, 2002.

[2] Q. Zhao, S. Bhowmick, "Association rule mining: A survey. Nanyang Technological University," Singapore, 2003.

[3] B.C.M. Fung, K. Wang, R. Chen, P.S. Yu, "Privacy-preserving data publishing," ACM Computing, 2010.

[4] R. Shrivastava, R. Awasthy, B. Solanki, "New Improved Algorithm for Mining Privacy Preserving Frequent Itemsets," International Journal of Computer Science & Informatics, 2011.

[5] N. Pasquier, Y. Bastide, R. Taouil, L. Lakhal, "Efficient mining of association rules using closed itemset lattices," Information systems, 1999.

[6] N. Mohammed,B.C.M. Fung, P.C.K Hung, "A survey on the privacy preserving algorithm of association rule mining," Electronic Commerce and Security, 2009.

[7] M. Atallah, A. Elmagarmid, M. Ibrahim, "Disclosure limitation of sensitive rules. Knowledge and Data Engineering Exchange," 1999.

[8] Y. Saygin, V.S. Verykios, C. Clifton, "Using unknowns to prevent discovery of association rules,"ACM SIGMOD Record, 2001.

[9] S.R.M. Oliveira, O.R. Zaiane, "Privacy preserving frequent itemset mining," Proceedings of the IEEE international conference on Privacy, 2002.

[10] V.S. Verykios, A.K. Elmagarmid, E. Bertino, "Association rule hiding. Knowledge and Data Engineering," IEEE Transactions, 2004.

[11] S.L. Wang, A. Jafari, "Hiding sensitive predictive association rules," IEEE International Conference, 2005.

[12] S. Wang, T. Lai, T. Hong T, "Hiding collaborative recommendation association rules," Applied Intelligence, 2007.

[13] C.C. Weng, S.T. Chen, H.C. Lo, "Novel algorithm for completely hiding sensitive association rules," Intelligent Systems Design and Applications, 2008.

[14] C.N. Modi, U.P. Rao, D.R Patel, "Maintaining privacy and data quality in privacy preserving association rule mining," International Conference on. IEEE, 2010.

[15] I. Chandrakar, Y.U. Rani, M. Manasa, K. Renuka, "Hybrid algorithm for privacy preserving association rule mining," Journal of Computer Science, 2010.

[16] R.S. Effects, K. Shah, "Association Rule Hiding by Heuristic Approach to Reduce Side Effects and Hide Multiple R. H. S. Items," International Journal of Computer Applications, 2012.

[17] N.H. Domadiya, U.P Rao, "Hiding sensitive association rules to maintain privacy and data quality in database," Advance Computing Conference (IACC), 2013.

[18] D. Jain, P. Khatri, R. Soni, "Hiding Sensitive Association Rules without Altering the Support of Sensitive Item (s)," Advances in Computer Science and Information Technology, 2012.

[19] C. Wei Lin, B. Zhang, K. Tung, T. Hong, "Efficiently Hiding Sensitive Itemsets with Transaction Deletion Based on Genetic Algorithms," The Scientific World Journal, 2014.

[20] B. Jadav, J. Vania, R. Patel, "Efficient Hiding of Sensitive Association Rules for Incremental Datasets," International Journal of Innovations & Advancement in Computer Science (IJIACS), 2014.

[21] M. Fouladfar, M. Naderi, "A heuristic algorithm for quick hiding of association rules," Advances in Computer Science: an International Journal (ACSIJ), 2015.

[22] S. Oliveira, O. Zaiane, "Algorithms for balancing privacy and knowledge discovery in association rule mining," Database Engineering, 2003.

[23] M.J Zaki, S. Parthasarathy, M. Ogihara, "New Algorithms for Fast Discovery of Association Rules," Technology in society, 1997.

[24] R Agrawal, T. Imieliński, A. Swami, "Mining association rules between sets of items in large databases," ACM SIGMOD Record, 1993.
[25] M.N. Dehkordi, K. Badie, A.K.A. Zadeh, "A novel method for privacy preserving in association rule mining based on genetic algorithms," Journal of software, 2009.
[26] N. Radadiya, N. Prajapati, K. Shah, "Privacy Preserving in Association Rule mining," Information systems, 2013.
[27] S. Kasthuri, T. Meyyappan, "HIDING SENSITIVE ASSOCIATION RULE USING HEURISTIC APPROACH," International Journal of Data Mining & Knowledge Management Process, 2013.
[28] S.L. Wang, R. Maskey, A. Jafari, T.P. Hong, "Efficient sanitization of informative association rules," Expert Systems with Applications, 2008.
[29] S.R.M. Oliveira, O.R. Za, Y. Saygin, "Secure association rule sharing," Advances in Knowledge Discovery and Data Mining, Springer.
[30] V.S. Verykios, E.D. Pontikakis, Y. Theodoridis, L. Chang, "Efficient algorithms for distortion and blocking techniques in association rule hiding," Distributed and Parallel Databases, 2007.
[31] M. Naeem, S. Fong, S. Asghar, "Hiding sensitive association rules using central tendency," Advanced Information Management and Service (IMS), 2010.
[32] X. Sun, P.S. Yu, "A border-based approach for hiding sensitive frequent itemsets. Data Mining," Fifth IEEE International Conference, 2005.
[33] G. Moustakides, V.S. Verykios, "A MaxMin approach for hiding frequent itemsets," Data & Knowledge Engineering, 2008.
[34] S. Menon, S. Sarkar, S. Mukherjee, "Maximizing accuracy of shared databases when concealing sensitive patterns," Information Systems Research, 2005.
[35] Y. Guo, "Reconstruction-based association rule hiding," Ph. D. Workshop on Innovative Database Research, 2007.
[36] J. Vaidya, W. Lafayette, C. Clifton, "Privacy preserving association rule mining in vertically partitioned data," international conference on Knowledge discovery and data mining, 2002.