Swapping-based Data Sanitization Method for Hiding Sensitive Frequent Itemset in Transaction Database

Dedi Gunawan*, Yusuf Sulistyo Nugroho, Maryam
Informatics Engineering Department
Universitas Muhammadiyah Surakarta

***Abstract*—Sharing a transaction database with other parties for exploring valuable information becomes more recognized by business institutions, i.e., retails and supermarkets. It offers various benefits for the institutions, such as finding customer shopping behavior and frequently bought items, known as frequent itemsets. Due to the importance of such information, some institutions may consider certain frequent itemsets as sensitive information that should be kept private. Therefore, prior to handling a database, the institutions should consider privacy preserving data mining (PPDM) techniques for preventing sensitive information breaches. Presently, several PPDM methods, such as item suppression-based methods and item insertion-based methods have been developed. Unfortunately, the methods result in significant changes to the database and induce some side effects such as hiding failure, significant data dissimilarity, misses cost, and artificial frequent itemset occurrence. In this paper, we propose a swapping-based data sanitization method that can hide the sensitive frequent itemset while at the same time it can minimize the side effects of the data sanitization process. Experimental results indicate that the proposed method outperforms existing methods in terms of minimizing the side effects.**

***Keywords*—Transaction database; data sanitization; data mining; sensitive frequent itemset; swapping-based method**

I. Introduction

Retails and supermarkets are actively collecting their customers' data transactions. The collected data is then stored in a database, and it is referred to as a transaction database. A transaction database $\mathcal{D}$ contains a set of transactions such as in Table I. In general, a set of transaction records T is a non-empty set where $T = \{t_1, t_2, t_3, \ldots, t_x\}$. Each transaction t is composed of a transaction id Tid, customer name or id number $Cname$, and a set of items bought by the customer, IID. The transaction database provides various benefits for the business institutions when they perform data analysis, such as using data mining technology. Unfortunately, analyzing such a transaction database by using data mining techniques is not a trivial task for these institutions since many of them do not have sufficient resources, i.e., computation resources and human resources, to perform the data mining task. Therefore, they opt to handle the transaction database to other parties, for example, a data mining company to conduct the task. Even though this solution may solve the problem, sharing the transaction database may bring a hidden threat since there might be sensitive information resides the database.

One of the data mining tasks that are widely employed in various domains is frequent itemset mining [1]. The frequent itemset mining is very useful to find the frequently bought items as well as to analyze customer buying patterns in transaction databases. Moreover, understanding such information allows the companies to enhance their marketing strategy as a way to increase business revenue. Referring to the Table I as an illustration, a company, defines that an itemset $\{1, 3\}$ has valuable information that should be learned by others. The table shows that item id 1, $iid = 1$ and item id 3, $iid = 3$, are frequently appear together in several transactions such as in t_1, t_5, t_7, and t_{10}. Due to the importance of this information, the company does not want any other parties exploring such an itemset. Concealing sensitive information is mandatory prior to sharing databases [2]. Therefore, data sanitization methods should be taken into account by the database owner to enable database sharing while at the same time preserving sensitive frequent itemset from being disclosed by external parties during the data mining process.

Recently, various data sanitization methods have been proposed with different settings and assumptions. Most of them rely on item suppression-based and item insertion-based strategies to address the aforementioned problem. However, the methods that follow suppression-based strategy [3], [4] incur significant side effects such as hiding failure, significant data dissimilarity, misses cost, and artificial frequent itemset occurrence. Accordingly, the data utility of the sanitized one degrades drastically, leading to induce inaccurate information for data recipients. The term hiding failure represents the percentage of sensitive frequent itemset that fail to be hidden by the data sanitization algorithm. Meanwhile, data dissimilarity measures the difference between an original database and its anonymized version in terms of its items frequency. Misses cost indicates the percentage of non-sensitive frequent itemsets that cannot be discovered in a sanitized database. Simultaneously, artificial frequent itemset corresponds to any frequent itemset that previously do not exist in an original database; however, it newly appears as the frequent itemset in a sanitized database.

Therefore, in this paper, a distinct data sanitization method is proposed. The proposed method follows the swapping-based strategy to ensure privacy protection in a database while at the same time preventing excessive side effects of the data sanitization process. The method follows a recent data swapping method that has been developed in [5] to generate an anonymized database. The proposed method uses an item collision detection strategy, and it carefully selects a pair of

TABLE I. EXAMPLE OF CUSTOMER TRANSACTION DATABASE $\mathcal{D}$

Tid	Cname	IID
t_1	John	1,2,3,8,10
t_2	Alice	2,7,8,10
t_3	Mark	5,6,7,10,12
t_4	Martin	2,3,8,9
t_5	Amar	1,3,5,9,10
t_6	Felix	4,6,7,9,10,12
t_7	Nita	1,3,5,8,11
t_8	Marta	1,6,4,7,9
t_9	Ben	5,12
t_{10}	Doet	1,3

transaction records for swapping by evaluating item similarity in the transaction records. To the best of our knowledge, our proposed method is the first method which uses the swapping technique in PPDM to hide sensitive frequent itemset.

The rest of the paper is organized as follows: Section 2 explores related work. The proposed method is explained in Section 3. Section 4 and 5 describe the experimental result and conclusion, respectively.

II. RELATED WORK

A. Frequent Itemset Mining

Frequent itemset mining is a data mining task which aims to explore all combinations of itemset contained in transaction records under a certain number of occurrence frequency threshold [6], [7]. Prior to performing frequent itemset mining, a database owner needs to determine a minimum support threshold value. In addition, there is no certain fixed number of minimum support thresholds, and thus if a database owner sets the frequency threshold too low, the database may output a significant number of frequent itemset and vice versa.

Suppose we have a transaction database denoted as $\mathcal{D}$. Support $supp$ of an itemset X, is the total number of transactions in $\mathcal{D}$ containing X. We denote the support of itemset X in a database $\mathcal{D}$ as $supp(X, \mathcal{D})$. To compute the $supp(X, \mathcal{D})$, one can divide the frequency of itemset $X \in \mathcal{D}$, $f(X)$, over the total number of transaction records in the database $|\mathcal{D}|$. An itemset X is called frequent itemset FI if $supp(X, \mathcal{D})$ is greater or equal to the number of determined minimum support $minSupp$ [8]. Thus, any itemset having the support value below the $minSupp$ can be referred as FI. To compute the $supp$ of itemset X in $\mathcal{D}$ we can refer to (1).

$$supp(X, \mathcal{D}) = \frac{f(X)}{|\mathcal{D}|} \tag{1}$$

B. Sensitive Frequent Itemset

Sensitive frequent itemset refers to any frequent itemset in which if such itemset are disclosed during the mining process conducted by other parties, and the database owner may lose their interest. In general, the database owners determine a set of a sensitive frequent itemset. Thus, if we formally denote the sensitive frequent itemset $Fs(X, \mathcal{D})$ as frequent sensitive itemset, then $Fs(X, \mathcal{D}) \subset FI$. Any other frequent itemset which is not considered as sensitive can be referred as non-sensitive frequent itemset Fn, where $Fs \neq Fn$ and $FI = Fs \cup Fn$. The relation between Sensitive frequent itemset and frequent itemset can be depicted in Fig. 1.

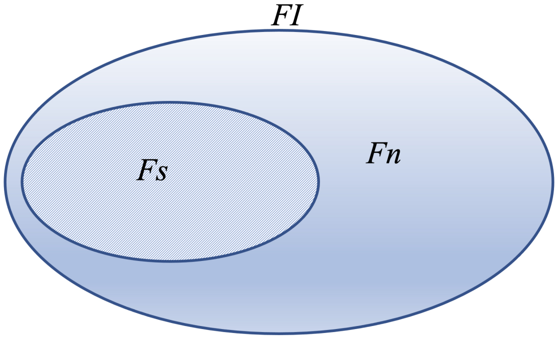


Fig. 1. Relation Among Fs, Fn and FI.

C. Data Sanitization Method

Data sanitization methods can be grouped into three main categories such as perturbation-based method, cryptographic-based method, and heuristics-based methods [9]. It has been proved that achieving a sanitized database that guarantees privacy protection and preserves maximum database utility is an NP-Hard problem [10], [11]. Therefore, various data sanitization methods with distinct parameters and settings have been proposed to address the issue. In addition, each proposed method is application-specific where it is designed for a particular problem, and it may not be adequate to work on another problem. For example, a data sanitization method that is intended for protecting sensitive frequent itemset mining is not suitable for privacy preserving data clustering. Thus, there is no one method fits all.

1) Perturbation-based Method: A perturbation-based method relies on a perturbing database either by removing items or inserting artificial items into transactions in the database. An initial data sanitization which follows the concept of the reconstruction-based to hide sensitive frequent itemset has been proposed in [4]. One of the solutions in the proposed method is called Naïve approach. It removes all the sensitive itemsets from a transaction database such that the sensitive information cannot be disclosed. While the technique effectively addresses the privacy problem, it causes significant item loss due to the removing process.

In reality, items in the transaction database may have a different level of importance. For example, item x is an item that is less important than item y since x generates low profit in a business process while item y is considered as an essential item due to its economic value. Therefore, a method that considers various threshold sensitivity has been proposed in [12]. The technique does not arbitrarily suppress all the sensitive frequent itemsets; instead, it creates a template containing possible victim items to disguise them. Another perturbation-based method has also been proposed in [13] namely rotation perturbation. However, the method is specifically designed to address sensitive information issues in clustering data mining. To solve the item loss issue, a technique that uses transaction insertion has been introduced in [14]. However, the method results in a significant difference between an original database and the sanitized one.

To optimize the performance of data sanitization, a method which based on particle swarm optimization (PSO) have been proposed in [15]. The method achieves a sanitized database by removing sensitive items in specific transaction record while at the same time reducing the side effects. The size of database is also another challenge to solve. Concerning that issue a method called MR-OVnTSA have been proposed in [16]. The method hides frequent sensitive itemsets in big data environment by removing items and transactions that can balance the privacy and knowledge in the database.

2) Cryptographic-based Method: Realizing that transaction database is potentially analyzed by several geographically separated parties, another scenario of hiding frequent sensitive itemset in a distributed system has also been intensively studied. Pioneering work in this area is proposed in [17], [18]. The methods use a secure multi-party computation technique to where several parties perform data mining analysis. To improve the quality of the sanitized database, a more recent approach in [19] proposes a cryptographic technique to hide sensitive rules in transaction databases. The method successfully protects the transaction database from inference attacks. A recent method in [20] proposed employs a cryptographic technique where it improves the mining process by disjoining the encrypted transactions into a certain number of blocks and only uses bilinear pairs of ciphertexts from the blocks. Therefore, the approach becomes more applicable in real-life cases. Even though the cryptographic-based method provides a strong privacy guarantee, however, when it meets a huge-sized transactional database, the performance decreases drastically due to the encrypt and decrypt process.

3) Heuristic-based Method: As it has been mentioned that finding maximum privacy guarantee and maximum database utility is an NP-Hard problem, a close to an exact solution which is based on a heuristic approach needs to be devised to address the problems in a real-life scenario. Presently, various heuristic-based methods have been proposed under different settings and parameters. One of the pioneering works in this area, such as in [4], [21]. In literature, most of the heuristics-based methods apply either item pruning or artificial transaction insertion strategy to reduce the support of itemset, and therefore it successfully hides the sensitive frequent itemset in a database.

Distinct from the previous approach, [22] proposed a method which uses a unique strategy where it does not reduce the support of itemset to hide the sensitive frequent itemset; instead, it considers representative rules to remove the rules at the beginning. Another recent study proposed in [3] also adopts heuristic-based data sanitization method where the method performs item pruning strategy, and it successfully hides sensitive itemset in a database. To select the items for the pruning process, the method considers calculating the frequency of sensitive items and removing the one which causes a minimum item loss.

It is undoubtedly true that the heuristic-based method which uses either items pruning strategy or artificial transaction insertion can successfully hide sensitive frequent itemset in a database. Unfortunately, such strategies lead the database to lose its useful information due to some items are missing from the database. In addition, artificial transaction insertion strategy results in excessive changes to the database as a result, the item composition between an original database and the sanitized one differs significantly.

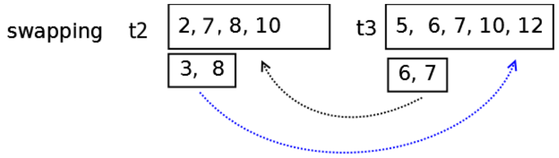


Fig. 2. Swapping Item from t_2 to t_3 and Vice Versa.

D. Swapping Techniques

The principle of data swapping technique is moving items from a certain transaction record to another record and vice versa. Therefore, it does not remove or add items in the transaction records, as a result, the database content can be well preserved. The data swapping techniques have been widely adopted for controlling statistical disclosure in micro dataset sharing. Pioneering work to protect sensitive information using the swapping technique was developed in [23], [24]. The method has successfully protects sensitive information in numerical and categorical attributes.

Regardless of a debate on its side effect, i.e., the techniques cause information incorrectness at a record level due to items of transaction records being swapped to another record. However, the techniques can successfully maintain items in the transaction database from loss. Thus, data recipients may perform data exploration to obtain all information of the items in the sanitized database.

The illustration of the swapping technique in transaction database is described in Fig. 2.

III. Proposed Method

To successfully hide sensitive frequent itemsets while at the same time maintaining the database utility, in this research, we propose a swapping-based data sanitization method. To the best of our knowledge, our proposal is the first data sanitization method that adopts swapping strategy. The swapping strategy does not remove items from a database and inserts new artificial transactions into the database; instead, it swaps items from one transaction to another. Accordingly, the side effect such as the number of artificial frequent itemset in the sanitized database can be minimized. An initial work in swapping strategy is firstly introduced in [25] to control data from disclosure. In this paper, the proposed method is distinct from the initial work which relies on a randomization strategy to protect the database. Our solution framework can be described in Fig. 3.

To evaluate whether an itemset is called a frequent itemset in $\mathcal{D}$, the data owner needs to determine a certain value called minimum support threshold, $minSupp$ and perform frequent itemset mining. All the obtained itemsets having support value greater than or equal to the $minSupp$ is called frequent itemsets, FI. The next step is the database owner defines a set of sensitive frequent itemsets Fs from the FI, where $Fs \subset FI$. The Fs is a non empty set containing sensitive frequent itemset si, thus $Fs = \{si_1, si_2, \ldots, si_n\}$. Meanwhile, all the frequent itemsets that are not considered as Fs are called non-sensitive frequent itemset Fn, and it does not need to be hidden in a $\mathcal{D}$, such that $FI = Fs \cup Fn$.

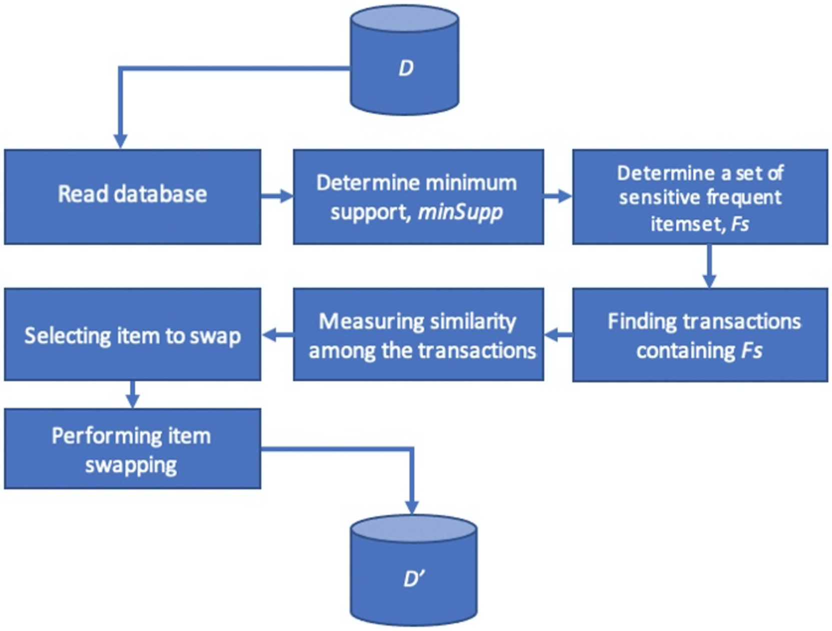

D
Read database
Determine minimum support, minSupp
Determine a set of sensitive frequent itemset, Fs
Selecting item to swap
Measuring similarity among the transactions
Finding transactions containing Fs
Performing item swapping
D'

Fig. 3. Swapping-based Data Sanitization Framework.

In general, database owners can determine Fs in two ways. The first is database owners define Fs according to their intention from business perspective, and the second is customers can freely determine their purchased items as either sensitive or non-sensitive itemset [26]. In this research, we follow the first approach where the database owners determine a set of itemsets in which according to his/her point of view it is considered as sensitive information.

A. Reading and Segmenting Database

Initially, our proposed method scans a database $\mathcal{D}$ and reads each transaction record $t_x \in \mathcal{D}$. During the reading process, the method identifies each t_x to check whether it contains sensitive frequent itemset si. For each t_x containing si, append the t_x to a bucket TFs otherwise append it to another bucket TFn. In this step, TFs and TFn have influence in separating the sensitive and non-sensitive transactions in database. Therefore, the TFs only contains a set of transactions containing si, while TFn is only containing a set of transactions not having si. The pseudo-code of this procedure is presented in the following Algorithm 1.

B. Measuring Transactions Similarity and Pairing the Transactions

Following the previous step, the proposed method measures similarity among transactions to obtain a pair of transactions for the swapping, P. P is used to simplify the pairing process of two transactions that will be used for swapping procedure. In this research, we follow the idea of [27] where the Jaccard coefficient is adopted to measure the similarity of transactions. In essence the Jaccard coefficient Jc computes the number of items that coexist in the two records over the number of the total item from those records. The formula of Jc measurement is depicted in (2).

$$Jc(t_x, t'_x) = Max\frac{(t_x \cap t'_x)}{(t_x \cup t'_x)} \tag{2}$$

Algorithm 1: Reading and segmenting database

```
Input: $\mathcal{D}$, $si \in SI$
Result: $TFs$ and $TFn$
Scan $\mathcal{D}$
$\forall\ t_x \in \mathcal{D}$
if $si \subseteq t_x$ then
    add the $t_x$ to $TFs$
else
    add the $t_x$ to $TFn$
end
```

Algorithm 2: Measuring similarity and finding a pair

```
Input: $TFs$
Result: $P$
$\forall\ t_x \in TFs$
select a record $t_x \in TFs$, randomly
select another record $t'_x \in TFs$
while $si \subseteq t_x \neq si \subseteq t'_x$ do
    compute $Jc(t_x, t'_x) = Min\frac{(t_x \cap t'_x)}{(t_x \cup t'_x)}$
end
select a pair $P$ having the minimum $Jc$
```

To avoid an item collision which may result in item loss and reduce the number of generated artificial frequent itemsets in a sanitized dataset, the proposed method implements two protocols. The first is our method only selects a pair of records that have the minimum similarity. Initially, the method selects a transaction $t_x \in TFs$ randomly, and then it picks another transaction $t'_x \in TFs$, selected transaction is referred as P.

The second step is our method ensures the sensitive itemsets si should not coexisting in both transactions, i.e., $si \in t_x \neq si' \in t'_x$ of P. While the si of both transaction are different, the algorithm computes the Jc. The next step is selecting a pair of records P which has the minimum Jc. Therefore, when the item $i \in si$ of the pair P are swapped to each other, the process does not cause item collision in both transactions significantly. In addition, such procedure can successfully ensure the hiding of sensitive frequent itemsets and minimize data dissimilarity. Algorithm 2 represents the pseudo-code of this procedure in detail.

1) Selecting Item for Swapping: Once the pair P have been determined, the following step is selecting items from the P to swap. In general, arbitrarily swapping items from these transactions may also hide the sensitive frequent itemset for both transactions. However, this action may distort item correlation in the transactions that result in significant changes in a sanitized database content [28]. To address this problem, in this research, the strategy in [5] is adopted. The key point of the strategy is checking whether items $i \in t_x$ that will be swapped are coexisting with that in t'_x.

Referring to Table I as an illustration, we aim to swap $si \in t_2$ with $si' \in t_3$. Let us denote item id as iid, for example, an item namely coffee has iid = 7 is a subset of sensitive frequent itemset si appears in t_2 and it also coexists in t_3. Swapping the iid= 7 from t_2 with another item such as bread i.e., an item with iid = 6 that presents in t_3 can successfully hides $si \in t_2$. However, due to the iid = 7 coexists in t_3, while the iid = 6 does not present in t_2, swapping the iid = 7 from $t_2$causes an item collision in to the transaction t_3, as a result, the t_3 looses one of its items i.e., iid = 6 and it is no longer exists in the t_3. Accordingly, to successfully hide the $si \in t_x$ while at the same time reduce the number of items loss in the transactions, the proposed method selects items that do not cause item collision.

In addition, to minimize the amount of data utility loss, the proposed method also selects the sensitive items $i \in si$ that have the minimum support Pr in the $\mathcal{D}$. Selecting items $i \in si$ with the lowest Pr can minimize the changes of item correlation in t_x. For example, suppose we have a sensitive itemset with iid = 2 and iid = 3, $\{2, 3\}$. Referring to the Table I, the Pr of iid = 2 is 3/10=0.33 while the Pr of iid = 3 is 5/10=0.50. To hide the sensitive itemset we would like to swap either iid = 2 or iid = 3. Suppose we select iid = 3 as the item to swap, the item correlation of iid = 2 with other items is significantly distracted since it appears five times in the $\mathcal{D}$. On the other hand, when iid = 2 is selected to swap, its item correlation with other items is not significantly reduced due to its appearance in the $\mathcal{D}$ is lower than that of the iid = 3, as a result, only small parts of the transactions in the $\mathcal{D}$ experience changes.

Thus, to be selected as the items for the swapping process, the items $i \in si$ have to satisfy these two conditions. Firstly, the items $i \in si$ should not collide with any other items $i \in t'_x$. Secondly, it should have the lowest probability distribution in $\mathcal{D}$. Thus, it can successfully minimize the number of artificial frequent itemsets in the sanitized database $\mathcal{D}'$. The detail of item selection is described in Algorithm 3.

Algorithm 3: Procedure of items selection for swapping

```
Input: $P$
Result: $i'$
1  calculate the $Pr$ of $i \in si$ of $t_x$
2  select $i \in si$ of $t_x$ that has the minimum $Pr$
3  check whether the $i$ exist in $t'_x$
4  if $i \neq i_j \in t'_x$ then
5      select the $i$ as the item for swapping
6  else
7      repeat step 4
8  end
9  return $i'$;
10 end;
```

Algorithm 4: Procedure of items swapping

```
Input: $P$, $TFn$
Result: $\mathcal{D}'$
    create Buffer $br_{t_k}$ and $br_{t'_k}$;
    $br_{t_k}$.add ($i \in si$ of $t_k$);
    $br_{t'_k}$.add ($i \in si$ of $t'_k$);
    append $br_{t'_k}$ to $t_k$;
    append $br_{t_k}$ to $t'_k$;
    merge $P'$ + $TFn$;
    save to $\mathcal{D}'$;
    end;
```

2) Swapping the Selected Items: Once the items for the swapping process have been determined, the next step is performing item swapping between that of t_x and t'_x. To swap the items, the proposed method creates two buffers for storing the items $i \in t_x$ and $i' \in tx'$. At first, the item from t_x is stored in buffer br_{tk} and that of tx' is stored in $br_{t'k}$. In this stage, br is a buffer to temporarily store the modified transaction records in swapping process. The second step is taking the items in $br_{t'k}$ and appending it to the t_x. Following that, items in br_{tk} is appended to t'_x. The procedure is performed until all i' from the pairs of records P have been swapped. Once the swapping process is finished, the algorithm can combine all the transaction records from TFn to successfully generate a sanitized database $\mathcal{D}'$. Algorithm 4 represents the pseudo-code of item swapping in detail.

IV. Experimental Results

To evaluate the effectiveness of the proposed method, we conduct several extensive experiments using several real datasets such as the foodmart dataset [29]. The properties of the dataset are described in Table II, while the testing parameters are presented in Table III. We implement the algorithm in JAVA code and run it in UNIX operating system with memory of 8 GB and storage of 256 GB. An additional tool, namely SPMF [30] is also adopted to generate frequent itemset by utilizing FP-Growth algorithm [31].

A. Evaluation Metrics

To verify the performance of the proposed method, we compare the proposed method, SW with several existing sen-

TABLE II. DATASETS PROPERTIES

Properties	Datasets
	$FoodMart$
# transactions	4,141
# items	18,319
# distinct items	1,559
Average tuple length	11.75

TABLE III. TESTING PARAMETERS

Parameter	Dataset
	$FoodMart$
$minSupp$	0.03% - 0.1%
$\|Fs\|$	$\|FI\|$*0.5
Avg. si length	4

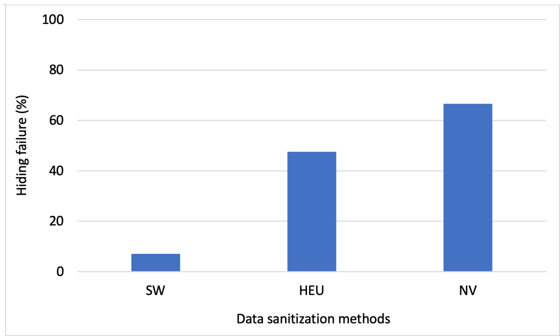


Fig. 4. Hiding Failure.

TABLE IV. NUMBER OF MC OF $\mathcal{D}'$

Methods	$minSupp$			
	0.03%	0.05%	0.07%	0.10%
SW	0.064	0.006	0.006	0
HEU	0.034	0.005	0.005	0
NV	0.276	0.029	0.029	0.034

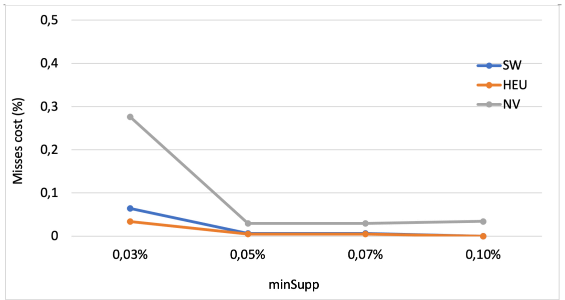


Fig. 5. Misses Cost.

sitive frequent itemset methods, i.e., heuristic method, HEU [3] and naïve method, NV [4]. Testing parameters are also determined in this experiment, and the detail is presented in Table III. Several metrics are adopted to evaluate the performance of the proposed method, such as hiding failure, misses cost, dissimilarity, and artificial frequent itemset [32].

1) Hiding Failure: Hiding failure, HF is a metric to evaluate the percentage of sensitive frequent itemsets that fail to be hidden. Ideally, a data sanitization method should be able to hide all the sensitive frequent itemsets in a database, i.e., the HF is 0. However, in some cases because of the data sanitization method's inaccuracy, several sensitive frequent itemsets are failed to hide. The metric to evaluate HF is presented in (3), where $\#Fs(\mathcal{D})$ represents the number of sensitive frequent itemsets in an original database and $\#Fs(\mathcal{D}')$ refers to that of the sanitized one.

Referring to Fig. 4, we can observe that the proposed method results in the lowest percentage of hiding failure. Even though SW fails to hide some si, the percentage of the failure is insignificant compared to that of other methods. The percentage of HF induced by the SW is around 7.143%, while the percentage of HF resulted from HEU and NV are 47.619% and 66.667%, respectively. The method successfully achieves the results since it takes a pair of records and swaps the items in si of the records.

$$HF = \frac{\#Fs(\mathcal{D})}{\#Fs(\mathcal{D})} \tag{3}$$

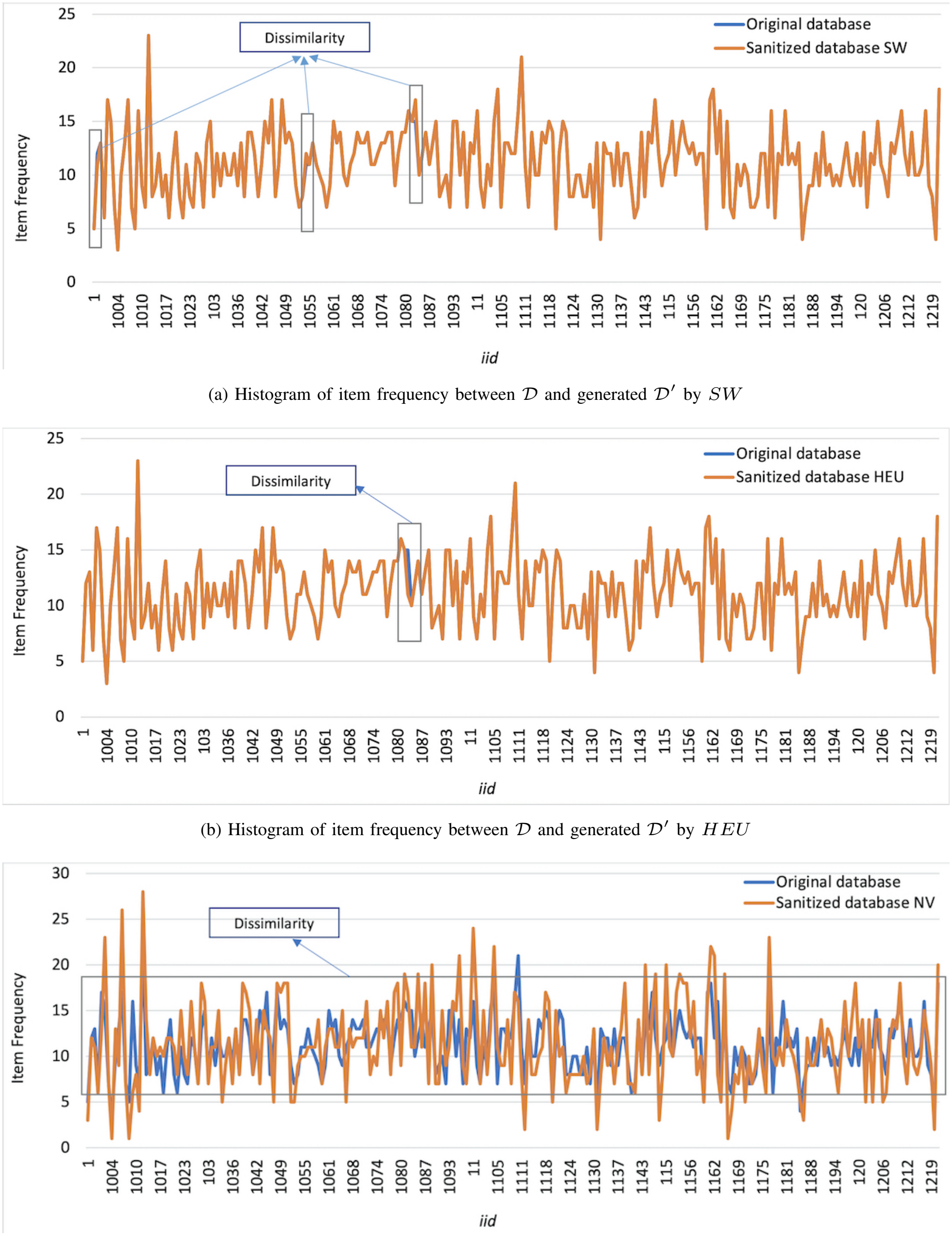
2) Misses Cost: The term misses cost, MC refers to the percentage of non-sensitive frequent itemsets F_n that are accidentally hidden when performing data sanitization. Ideally, the percentage of MC is 0%. The formula to compute MC is described in equation 4, where $\#Fn(\mathcal{D})$ and $\#Fn(\mathcal{D}')$ represent a set of frequent itemset that can be explored in $\mathcal{D}$ and a set of non-sensitive itemset that cannot be discovered in $\mathcal{D}'$.

As can be observed in Fig. 5, when the sanitized database resulted from SW is mined under $minSupp$ = 0.03%, our proposal induces a slightly higher percentage of MC compared to that of HEU. However, as the $minSupp$ value increases to $minSupp$ = 0.1% the proposed scheme achieves the same results as HEU. In addition, the SW successfully achieves better results compared to that of NV in terms of minimizing MC in all the varying $minSupp$ values. The detail values of MC among the methods are described in Table IV.

The main motivation of such results is due to the proposed method does not limit the number of modified records like in HEU. The HEU lefts some records containing si are kept unmodified to reduce the MC. However, such a strategy allows the si remain discoverable when data recipients perform frequent itemset mining using a lower confidence value than the $minnSupp$ value. As our goal is designing strong data sanitization, the proposed method does not apply the same strategy in HEU.

$$MC = \frac{\#Fn(\mathcal{D}) - \#Fn(\mathcal{D}')}{\#Fn(\mathcal{D})} \times 100\% \tag{4}$$

3) Dissimilarity: Applying data sanitization methods to a database always results in some changes to the database content. The changes in database content are considered as a side effect of the data sanitization methods, and it is referred to as dissimilarity. To evaluate the dissimilarity between an



(a) Histogram of item frequency between $\mathcal{D}$ and generated $\mathcal{D}'$ by SW

(b) Histogram of item frequency between $\mathcal{D}$ and generated $\mathcal{D}'$ by HEU

(c) Histogram of item frequency between $\mathcal{D}$ and generated $\mathcal{D}'$ by NV

Fig. 6. Histogram of Item Frequency Comparison.

original database and its sanitized version, one can compare the items' frequency in both databases. The formula to evaluate the dissimilarity $Diss$ is presented in (5), where $f\mathcal{D}(i)$ represents the frequency of item i in an original database $\mathcal{D}$ and $f\widetilde{\mathcal{D}}(i)$ refers to that of the sanitized one.

$$Diss(\mathcal{D}, \widetilde{\mathcal{D}}) = \frac{1}{\sum_{i=1}^{d} f\mathcal{D}(i)} \times \left| \sum_{i=1}^{d} f\mathcal{D}(i) - \sum_{i=1}^{\widetilde{d}} f\widetilde{\mathcal{D}}(i) \right| \quad (5)$$

As can be observed from Fig. 6a, the item frequency of the sanitized database $\mathcal{D}'$ generated by our proposed method SW is almost the same as that of the original database $\mathcal{D}$. Even though there are some differences in certain item frequency between the two databases, it does not significantly deviate. Referring to Fig. 6b, the item frequency in the sanitized database $\mathcal{D}'$ generated by HEU also experiences a small dissimilarity. Meanwhile, in Fig. 6c we can see that the item frequency in $\mathcal{D}'$ obtained from NV has a significant difference compared to the item frequency in the original database $\mathcal{D}$.

The summary of data dissimilarity of those databases is presented in Fig. 7. According to the figure, we can observe that the proposed method results in the lowest $Diss$ value

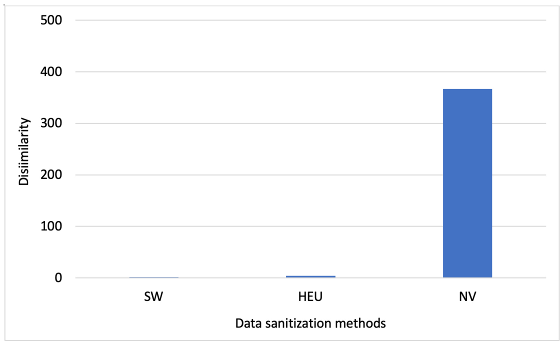


Fig. 7. Dissimilarity Value.

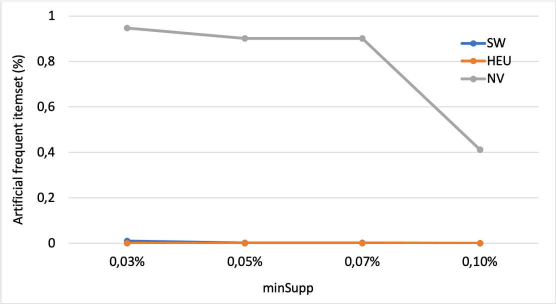


Fig. 8. Artificial Frequent Itemset.

compared to that of other methods. The $Diss$ value resulted from the proposed method is 1.372, while that of the HEU and NV are 4.327 and 366.436, respectively. The result is achieved because the proposed method can minimize the number of item losses in the sanitized database. Meanwhile, since the other two methods adopt a suppression strategy that removes items from a database, their dissimilarity values are higher than that of our proposed method.

4) Artificial Frequent Itemset: Artificial frequent itemsets, AFI is defined as a percentage of all frequent itemsets that do not present in an original database. However, it newly appears in the sanitized one. Ideally, the percentage is 0. The formula to compute the AFI is stated in equation (6).

$$AFI = \frac{|\widetilde{FI}| - |\widetilde{FI} \cap FI|}{|\widetilde{FI}|} \tag{6}$$

The notations $|\widetilde{FI}|$ and $|FI|$ represent the cardinality of frequent itemset in a sanitized database and that of the original database, respectively. As can be seen in Fig. 8, the sanitized database resulted by SW results in considerably lower AFI than that of NV. While, it has the same AFI as the HEU, when the $minSupport$ value is more than 0.03%. The proposed method, SW can minimize the AFI due to it does not remove or add items to a database. Therefore, the frequent itemset in $\widetilde{\mathcal{D}}$ remain the same as that of the original one. The detail values of the AFI is presented in Table V.

TABLE V. Number of AFI in $\mathcal{D}'$

Methods	$minSupp$			
	0.03%	0.05%	0.07%	0.10%
SW	0.009	0.001	0.001	0
HEU	0	0	0	0
NV	0.947	0.902	0.902	0.411

V. Threats to Validity

Threats to the *construct validity* relates to the proposed method's performance in handling various database with different properties. In our study, we only used one transaction database as described in the Table II. Even though we only used single database, however, it has more complex data properties compared to other databases that are usually used in PPDM areas such as $BMS-WebView1$ and $BMS-WebView2$ [33], specifically in the number of distinct items and the average of tuple length. Thus, we consider that the impact of using various database is not significant.

The second threats to validity is related to the performance of the proposed method compared to other more recent methods. Even though NV is not considered as the recent one, however, recent researches in PPDM [34], [35] still consider the method as the benchmark to evaluate the performance of their proposed method. Therefore, the impact of using other recent methods is small.

VI. Conclusion

In this paper, a data sanitization method based on a swapping approach called SW have been proposed. The main property of the proposed method is that it does not add or remove items in the database. The method has several steps to obtain a sanitized database. The main idea of the proposed method is finding transactions containing frequent sensitive itemset, measuring their similarity to determining a pair of records, and deciding items in the sensitive frequent itemset for the swapping process.

Experimental results show that in general the proposed method has a better performance compared to some existing methods. The method successfully hides the sensitive frequent itemsets with the lowest HF compared to that of several existing methods, indicating it provides stronger privacy protections in the sanitized database. In addition, since the method does not remove or add items in a database, the dissimilarity value between the original database and the sanitized one resulted from our method is lower than that of HUE and NV. In terms of data utility preservation, our method has a similar performance with HEU where the percentage of AFI is close to zero.

In the future, a more deeper analysis to the proposed method needs to be conducted, specifically in handling various transaction databases that have different properties and also evaluating the algorithm complexity. The proposed method SW also needs to be compared to more recent existing works in the same field to evaluate its performance.

Acknowledgment

This research is funded by Hibah Integrasi Tridharma (HIT) Universitas Muhammadiyah Surakarta under the grand number

015/A.3-III/FKI/I/2021.

REFERENCES

[1] D. Apiletti, E. Baralis, T. Cerquitelli, P. Garza, F. Pulvirenti, and L. Venturini, "Frequent Itemsets Mining for Big Data: A Comparative Analysis," *Big Data Research*, vol. 9, pp. 67–83, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.bdr.2017.06.006

[2] J. C.-W. Lin, T.-Y. Wu, P. Fournier-Viger, G. Lin, J. Zhan, and M. Voznak, "Fast algorithms for hiding sensitive high-utility itemsets in privacy-preserving utility mining," *Engineering Applications of Artificial Intelligence*, vol. 55, pp. 269–284, 2016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0952197616301282

[3] D. Gunawan and L. Guanling, "Heuristic Approach on Protecting Sensitive Frequent Itemsets in Parallel Computing Environment," in *The 1ST UMM International Conference on Pure and Applied Research (UMM-ICOPAR 2015)*, Malang, East Java, Indonesia, 2015, pp. 41–49.

[4] S. Oliveira and O. Zaiane, "Privacy preserving frequent itemset mining," *Proceedings of the IEEE international conference on Privacy, security and data mining-Volume 14*, vol. 14, pp. 43–54, 2002. [Online]. Available: http://portal.acm.org/citation.cfm?id=850782.850789

[5] D. Gunawan and M. Mambo, "Set-valued data anonymization maintaining data utility and data property," in *ACM International Conference Proceeding Series*. Association for Computing Machinery, jan 2018.

[6] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 2012.

[7] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical Machine Learning Tools and Techniques*, 2016.

[8] X. Cheng, S. Su, S. Xu, P. Tang, and Z. Li, "Differentially private maximal frequent sequence mining," *Computers and Security*, 2015.

[9] D. Gunawan, "Classification of Privacy Preserving Data Mining Algorithms : A review," *Jurnal Elektronika dan Telekomunikasi (JET)*, vol. 20, no. 2, pp. 36–46, 2020.

[10] R. Agrawal and R. Srikant, "Privacy-preserving Data Mining," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '00. New York, NY, USA: ACM, 2000, pp. 439–450. [Online]. Available: http://doi.acm.org/10.1145/342009.335438

[11] C. C. Aggarwal, J. Pei, and B. Zhang, "On privacy preservation against adversarial data mining," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[12] Y. P. Kuo, P. Y. Lin, and B. R. Dai, "Hiding frequent patterns under multiple sensitive thresholds," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5181 LNCS, pp. 5–18, 2008.

[13] K. Chen and L. Liu, "Privacy preserving data classification with rotation perturbation," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*, 2005, pp. 4 pp.–.

[14] L. Chun-Wei, H. Tzung-Pei, C. Chia-Ching, and W. Shyue-Liang, "A Greedy-based Approach for Hiding Sensitive Itemsets by Transaction Insertion," *Journal of Information Hiding and Multimedia Signal Processing.*, vol. 4, no. 4, pp. 201–2014, 2013.

[15] S. Jangra and D. Toshniwal, "VIDPSO: Victim item deletion based PSO inspired sensitive pattern hiding algorithm for dense datasets," *Information Processing and Management*, vol. 57, no. 5, p. 102255, 2020. [Online]. Available: https://doi.org/10.1016/j.ipm.2020.102255

[16] S. Sharma and D. Toshniwal, "MR-OVnTSA: a heuristics based sensitive pattern hiding approach for big data," *Applied Intelligence*, vol. 50, no. 12, pp. 4241–4260, 2020.

[17] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2000.

[18] ——, "Privacy preserving data mining," *Journal of Cryptology*, 2003.

[19] N. Rajesh and A. A. L. Selvakumar, "Association rules and deep learning for cryptographic algorithm in privacy preserving data mining," *Cluster Computing*, vol. 22, no. s1, pp. 119–131, 2019. [Online]. Available: https://doi.org/10.1007/s10586-018-1827-6

[20] C. Ma, B. Wang, K. Jooste, Z. Zhang, and Y. Ping, "Practical Privacy-Preserving Frequent Itemset Mining on Supermarket Transactions," *IEEE Systems Journal*, vol. 14, no. 2, pp. 1992–2002, 2020.

[21] A. HajYasien and V. Estivill-Castro, "Two new techniques for hiding sensitive itemsets and their empirical evaluation," *Data Warehousing Knowledge Discovery, Proc.*, vol. 4081, pp. 302–311, 2006.

[22] D. Jain, P. Khatri, R. Soni, and B. K. Chaurasia, "Hiding sensitive association rules without altering the support of sensitive item(s)," *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, vol. 84, no. PART 1, pp. 500–509, 2012.

[23] S. P. Reiss, M. J. Post, and T. Dalenius, "Non-reversible privacy transformations," in *Proceedings of the 1st ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, ser. PODS '82. New York, NY, USA: ACM, 1982, pp. 139–146. [Online]. Available: http://doi.acm.org/10.1145/588111.588134

[24] S. P. Reiss, "Practical data-swapping: The first steps," *ACM Trans. Database Syst.*, vol. 9, no. 1, pp. 20–37, Mar. 1984. [Online]. Available: http://doi.acm.org/10.1145/348.349

[25] T. Dalenius and S. P. Reiss, "Data-swapping: A technique for disclosure control," *Journal of Statistical Planning and Inference*, vol. 6, no. 1, pp. 73–85, 1982. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0378375882900581

[26] T. P. Hong, C. W. Lin, K. T. Yang, and S. L. Wang, "Using TF-IDF to hide sensitive itemsets," *Applied Intelligence*, vol. 38, no. 4, pp. 502–510, 2013.

[27] J. Wicker and S. Kramer, "The best privacy defense is a good privacy offense: obfuscating a search engine user's profile," *Data Mining and Knowledge Discovery*, vol. 31, no. 5, pp. 1419–1443, 2017.

[28] D. Gunawan and M. Mambo, "Data anonymization for hiding personal tendency in set-valued database publication," *Future Internet*, vol. 11, no. 6, 2019.

[29] P. Fournier-Viger, "Foodmart dataset," 2020. [Online]. Available: http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php

[30] P. Fournier-Viger, J. C. W. Lin, A. Gomariz, T. Gueniche, A. Soltani, Z. Deng, and H. T. Lam, "The SPMF open-source data mining library version 2," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9853 LNCS, pp. 36–40, 2016.

[31] G. Grahne and J. Zhu, "Fast algorithms for frequent itemset mining using FP-trees," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 10, pp. 1347–1362, 2005.

[32] L. Zhang, W. Wang, and Y. Zhang, "Privacy Preserving Association Rule Mining: Taxonomy, Techniques, and Metrics," *IEEE Access*, vol. 7, pp. 45 032–45 047, 2019.

[33] KDD-CUP, "KDD CUP 2000: Online Retailer Website Clickstream Analysis ," https://www.kdd.org/kdd-cup/view/kdd-cup-2000, 2000, [Online; accessed 7-June-2019].

[34] W. Wu, M. Xian, U. Parampalli, and B. Lu, "Efficient privacy-preserving frequent itemset query over semantically secure encrypted cloud database," *World Wide Web*, vol. 1, pp. 607–629, 2021.

[35] H. Chen, A. A. Heidari, X. Zhao, L. Zhang, and H. Chen, "Advanced orthogonal learning-driven multi-swarm sine cosine optimization: Framework and case studies," *Expert Systems with Applications*, vol. 144, p. 113113, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417419308309