www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 5 Issue 10 Oct. 2016, Page No. 18722-18725

Providing Privateness via Transaction Splitting Using PFP Growth Algorithm

M. Jhansi Rani[1], *K. Venkatagurunath Naidu*[2]

[1]Sri Padmavati Mahila Visvavidyalayam, School of Engineering and Technology,
Padmavati Nagar, Tirupati, Andhra Pradesh,INDIA
jhansimurari@gmail.com

[2]Sri Padmavati Mahila Visvavidyalayam, School of Engineering and Technology,
Padmavati Nagar, Tirupati, Andhra Pradesh,INDIA
venkatspmvv@gmail.com

Abstract: ***Frequent itemset mining is one of the most necessary problems in data extraction. The chance of devious a discrepancy private frequent itemset mining algorithm which can not only accomplish high data usefulness and a high level of secrecy, but also offer high time effectiveness. To this end, offer a discrepancy frequent itemset mining algorithm based on the Frequent Pattern-growth algorithm, which is referred to as Private Frequent Pattern-growth. The Private frequent Pattern -growth algorithm consists of a preprocessing phase and a mining phase. To improve the utility and secrecy tradeoff, a innovative smart splitting method is proposed to change the database in preprocessing phase. It needs to be performed only once for a given database. To compensate the information loss caused by transaction splitting, To estimate the definite support of itemsets in the original database in mining segment utilize run time estimation method. In accumulation, develop a dynamic reduction method to dynamically reduce the amount of noise added to guarantee privacy during the extracting process by leveraging the downward closure property. Private common pattern-growth algorithm is shown it is ε-differentially private through formal privacy analysis; explain that PFP-growth algorithm is ε- discrepancy secrecy. Extensive experiments on real datasets exemplify that our PFP-growth algorithm considerably outperforms the state-of-the-art techniques.***

Keywords: Frequent itemset mining, differential privacy, transaction splitting.

1. Introduction

Frequent itemset mining is one of the most fundamental problems in data extracting. It has practical significance in a wide range of application areas such as decision support. Frequent itemset mining tries to find itemsets that occur in transactions more commonly than a given threshold, where each operation contains a set of items for a given database. In spite of valuable insights the discovery of frequent itemsets can potentially provide, if the data is perceptive (e.g., web browsing history and medical records), releasing the discovered frequent itemsets might pose considerable threats to individual privacy. Differential privacy[1] has been planned as a way to address such problem. Unlike the anonymization-based solitude models (e.g.-anonymity[2] and –diversity[3]), discrepancy privacy offers strong hypothetical guarantee on the privacy of released data without making assumptions about an attacker's background information. In particular, Differential privacy assures that the output of a computation is insensitive to changes in any individual's record, and thus restricting secrecy leaks through the results by adding a carefully chosen amount of noise . For mining frequent itemsets a variety of algorithms have been proposed. The frequent pattern-growth[5] and Apriori[4] are the two most high-flying ones. In particular, Apriori is a breadth-first search, candidate set generation-and-test algorithm. If the maximal length of frequent itemsets is l it needs l database scan. In contrast, frequent pattern-growth which requires no candidate generation is a depth-first search. Compared with Apriori, frequent pattern-growth only performs two database scans, which makes common-growth an order of degree faster than Apriori. The interesting features of frequent pattern-growth motivate us to design a discrepancy private Frequent itemset mining algorithm into several subsets and guarantee each subset is under the limit. Develop a novel smart splitting method to transform the database. To ensure applying ε- discrepancy private algorithm on the transformed database still satisfies ε- discrepancy privacy for the original database, develop a weighted splitting operation. Moreover, to preserve more occurrence information in subsets. A graph-based approach to expose the association of items within transactions and make use of such correlation to guide the splitting method. In the extracting phase, motivated by the double standards method in , a run-time estimation method to reduce such information loss. Given the noisy support of an itemset in the database transformed by transaction splitting, firstly calculate its actual support in the transformed database, and then further compute its definite support in the original database. In addition, by maximizing the downward closure property[4] (i.e., any supersets of an infrequent itemset are infrequent), put forward a dynamic reduction method. During the extracting process, dynamically evaluation the number of support computations, so that progressively reduce the amount of noise required by discrepant confidentiality. Through formal privacy analysis, show that our secret common method of design -growth algorithm is ε-discrepancy secrecy. Wide experimental results on real datasets show that our algorithm outperforms existing discrepant private common itemset mining algorithms. Moreover, to exhibit the simplification of our transaction splitting techniques and further develop the application variety, apply transaction splitting techniques, including the smart splitting and run-time estimation methods, to Apriori by modifying the algorithm.

2 PRELIMINARIES

2.1 Differential Privacy

Differential privacy has progressively emerged as the de facto regular notion of privacy in data analysis. For two databases D and D', they are nearby databases if they diverge by at most one record. Formally, the differential privacy is defined as follows.

Definition1 :(ε- differential privacy). A private algorithm A satisfies ε- differential privacy iff for any two neighboring databases D and D', and any subset of outputs S ⊆ Range (A),Pr[A(D)∈ S]≤eε×Pr[A(D′)∈ S], where the possibility is taken over the arbitrariness of A . A fundamental concept in differential privacy is the sensitivity. The amount of injected noise is carefully calibrated to the sensitivity. The sensitivity of count queries is used to measure the maximum possible change in the outputs over any two neighboring databases.

Definition2: (Sensitivity). Given p count queries Q, for any neighboring databases D, D′, the sensitivity of Q is:ΔQ= max||Q(D)−Q(D′)||. For the computation whose outputs are real, D. The Laplace mechanism. The noise drawn randomly from the Laplace distribution is added laplace mechanism . The Laplace distribution with magnitude λ , i.e., Lap(λ), follows the probability density function as Pr[x|λ] =12λe−|x|/λ, where λ=ΔQε is determined by both the sensitivity ΔQ and the privacy budget ε.

Theorem1: Let Q denote a query sequence of length p with sensitivity Δ Q. Let⟨ξ1, ...,ξ p⟩ be a p-length vector ,where ξ i is drawn i.e. from a Laplace distribution with scale Δ Q /ε . Then, the algorithm A(D) =Q(D) +⟨ξ1, ..., ξp⟩ achieves ε-differential privacy. For the computation whose outputs are integer the Geometric mechanism has been proposed. The magnitude of injected noise conforms to a two-sided geometric distribution G (α) with the probability mass function

Pr [x|] =exp (α)−1exp(α)+1·exp(α)−|x| where α >0.

Theorem2: Let Q be a query sequence of length p with integer outputs, and its sensitivity is ΔQ. The algorithm A(D) =Q(D) +⟨ξ1, ..., ξp ⟩ gives ε-differentia privacy, where ξi .i.d. samples from a distribution G(ε/ΔQ).To support multiple differentially private computations, sequential symphony guarantees the overall privacy.

Theorem3 :(Chronological symphony). Let f_1... f_m be m randomized algorithms, where fi provides ε_i-differential privacy (1≤i≤m). A sequence of fi (D) over database D provides ($\sum \varepsilon_i$) -differential privacy.

3 Frequent Itemset Mining

Given the alphabet I= {i_1, ...,i_n}, and a transaction database D is a multiset of transactions a transaction is a subset of I. Each transaction represents an individual's record, shows a simple transaction database. an itemset is a non-empty set X⊂ I . The length of an itemset is the number of items in it. If itemset contains k Items ,itemset is called a k-itemset. A transaction contains an itemset X. If X is a subset of t. The support of itemset X is the number of transactions containing X in the database in the header table HT, by following the

Table 1
A Simple Transaction Database

TID	Items
101	f,a,b,c,d
202	b,c,e
303	e,f,a,c,g
404	b,c,d,e
505	a,c

linked list starting at ik in HT , branches that contain item ik are found. The portion of these branches from i k to the root forms ik's conditional pattern base $_{D\ i\ k}$. Then, for the first (k-1) items in HT, FP-growth computes their supports in D_{ik} and determines the frequent items in D_{ik}. For each frequent item i in D_{ik}, itemset {i, i k} is a frequent 2-itemset in the original databases. Next, based on the frequent items found in D_{ik}, FP-growth generates the header table HT $_{ik}$ and FP-tree ik for D_{ik}. The FP-tree constructed from Dik is called ik's conditional FP-tree. By using header table HT_{ik} and conditional FP-tree ik, FP-growth progressively grows each generated frequent 2-itemset by producing and mining its conditional pattern base. The above procedure is applied repetitively until no conditional pattern base can be generated.

4 A STRAIGHTFORWARDAPPROACH:

A straightforward approach to make FP-growth achieve ε-differential privacy. Foremost idea is to add noise to the support of itemsets and use their noisy supports to determine which itemsets are common. In particular, expect the maximal length of frequent itemsets is L_f and the alphabet size is n. We uniformly assign the support computations of i-itemsets a privacy budget ε/L_f(1≤i≤L_f). Then, we add geometric noisy G (εL_f×C_{1n}) to the support of items. If the noisy support of an item exceeds the threshold, we output it as a frequent 1-itemset. Next, based on the frequent 1-itemsets, we generate the header table and FP-tree, and progressively grow each frequent 1-itemset by producing and mining its conditional pattern base. Assume the conditional pattern base of an (i-1)-itemset X is D_X, and HT_X is the header table in D_X. For the k-th item ik in HT_X, we first create the conditional pattern base of itemset Y= {X∪ik}. Then, for the first (k-1) items in HT_X, we add geometric noisy (εLf×Cin) to their local supports in Y's conditional pattern base D Y. If the noisy support of an item i in DY exceeds the threshold, we output itemset {Y∪i} as a frequent (i+1)-itemset. Next, we create the header table and conditional common method of design -tree for D_Y. We progressively grow each new found frequent (i+1)-itemset by generating and mining its conditional pattern base.

Privacy Analysis:

We now give the privacy guarantee of above approach. In essence, the frequent itemsets are generated based on the noisy results of support computations. In particular, let Qi denote the support computations of i-itemsets. For a support computation q∈Qi, the amount of noise added in q depends on the sensitivity of Qi and the privacy budget allocated for Qi. Since adding (removing) one transaction, in the worst case, can affect the result of each support computation in Qi by one, the sensitivity of Qi is the number of support computations in Qi. However, unlike Apriori which computes the support of i-itemsets simultaneously, FP -growth is a depth-first search algorithm and computes the support of i-itemsets based on different conditional pattern bases. It is hard for FP-growth to obtain the exact number of support computations in Q_i during the mining process. To achieve differential privacy, we consider the number of all possible i-itemsets, C_{in}, as the sensitivity of

Q_i. Moreover, we uniformly assign Q_i a privacy budget ε/Lf($1 \le i \le L_f$). Thus, adding geometric noise G ($\varepsilon C_{in} \times L_f$) in Qi satisfies (ε/Lf) e-differential privacy. Since the maximal length of frequent itemsets is Lf, the mining process can be considered as a sequence of computations Q=⟨Q1, ...,Q_{Lf}⟩. Based on the sequential composition property , we can see this approach overall satisfies ε-differential privacy

Limitations of the Straightforward Approach:
The approach discussed above, however, is impractical. The major problem faced by this approach is the sensitivity of support computations is prohibitively high. It indicates that a large amount of noise is added to the support of itemsets. For instance, suppose the privacy budget is 1 and the maximal length of frequent itemsets is 10. We regularly assign the support computations of i-itemsets,
Q_i, a privacy budget 1/10($1 \le i \le 10$). Assume there are 10 4 items in the alphabet. It means the sensitivity of Q1 is 10 4. Therefore, we need to add geometric noisy G ($1 \div 10^4 \times 10$) to the support of each item, which will obviously produce invalid results.

Challenges of PFP –growth:
To improve the usefulness and privacy exchange, suggest to limit the length of transactions. They develop a differentially private Apriori-based frequent itemset mining algorithm by truncating transactions. For the mining of frequent i-itemsets, they re- scan the database and re-truncate transactions based on the discovered frequent (i-1)-itemsets to protect more frequency information. However, FP-growth scans the database only twice. We have no opportunity to re-truncate transactions. Thus, such approach is not suitable for FP-growth. Moreover, due to the downward closure property, it is needless to compute the support of an itemset if any of its subsets is known to be infrequent. Thus, for the support computations of i-itemsets Q_i , taking the number of all possible i-itemsets as the sensitivity of Q_i is clearly not best. To reduce the sensitivity of Q_i, a potential approach is to enforce the frequent (i-1)-itemsets to be generated simultaneously, such that we can obtain the exact number of support computations in Qi. However, in common method of design-growth, common itemsets of the same length are always generated based on different conditional common method of design-trees. If we enforce the itemsets of the same length to be generated simultaneously, the number of FP-trees stored in memory will grow at an exponential rate. Therefore, this approach is infeasible.

5 KEY METHODS:

In this section, we propose three key methods to address the challenges in designing a differentially private FIM algorithm based on the FP-growth algorithm. In particular, to limit the length of transactions without introducing much information loss, we propose our smart splitting method. Moreover, to offset the information loss caused by trans- action splitting, a run-time estimation method is used to estimate the actual support of itemsets in the mining process. Furthermore, to lower the amount of added noise, we develop a dynamic reduction method which dynamically reduces the sensitivity of support computations by decreasing the upper bound on the number of support computations. In the rest of this section, we discuss the details of the methods.

5.1Smart Splitting

To improve the utility-privacy tradeoff, we argue that long transactions should be split rather than truncated. That is, we transform the database by dividing long transactions into multiple subsets (i.e., sub-transactions), each of which meets the maximal length constraint. For example, assume itemsets
{a, b, c}and{d, e, f}are frequent and the maximal length constraint is 4. Given a transaction t={a, b, c, d, e, f}, if we simply truncate t to be{a, b, c, d},the support of itemset {d, e, f} and its subsets will all
decrease. Consequently, some itemsets which are frequent in the original database may become infrequent. Instead, if we divide t into t1={a, b, c}andt2={d, e, f}, the support of itemsets{a, b, c},{d, e,f} and their subsets will not be affected. However, Theorem4shows, if a transaction can be divided into at most k subsets, applying any ε-differentially private algorithm to the transformed database only ensures(k·ε)-differential privacy for the original database.

Theorem4: Let A be an ε-differentially private algorithm for the transformed database and f be a function that can divide one transaction into at most k subsets. Then, for any neighboring databases D and D’, and any subset of outputs S ⊆Range (A), we has:

$$\Pr(A(f(D)) = S) \le e^{k \cdot \varepsilon} \Pr(A(f(D')) = S).$$

Proof: Consider two neighboring databases D and D′.Let t denote the transaction in D′ but not in D(i.e., D′=D+t). Suppose the transformed database of D is ~D and t is divided intoksubsetst1, ...,tk. Since A is an ε
-differentially private algorithm for the transformed database ~D, based the definition of differential privacy, for any subset of outputs S ⊆Range(A), we have:

$$\Pr(A(\sim D) = S) \le e^{\varepsilon} \Pr(A(\sim\langle D, t1\rangle) = S).$$

Similarly, we can prove that:

$$\Pr(A(\sim D) = S) \le e^{k \cdot \varepsilon} \Pr(A(\langle \sim D, t1, ..., tk\rangle) = S).$$

Since ~D is the transformed database of D and t is divided into t1, ..., t,⟨D, t1, ..., tk⟩ can be considered as the transformed database of D′. The theorem then follows. To this end, we introduce the weighted splitting operation. When we divide a long transaction, we assign a weight to each generated subset. The weight of a subset indicates the change to the support of an itemset when adding (removing) this subset into (from) the database. It can be considered as a multiplier. For example, given a subset
ts={a,b,c,d} with weight 0.5, adding ts into the database will increase the support of its contained itemsets, e.g.,{a,b,c}, by 0.5. In the following, we formally define the weighted splitting operation.

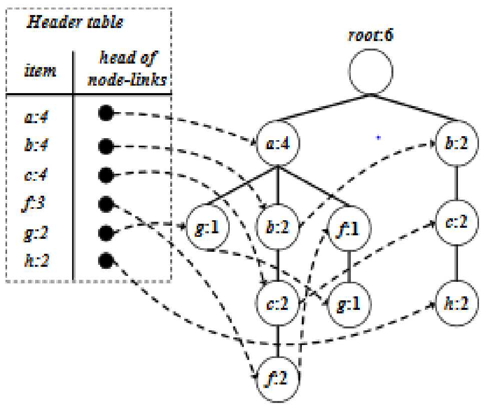

Figure 1 Header table and FP-tree for the given database

DOI: 10.18535/ijecs/v5i10.65

Definition 3:(Weighted Splitting Operation). Consider a transaction t whose length exceeds the maximal length constraint Lm. A function f divides t into multiple subsets t1, ...,tk, where ti is assigned a weight w_i and the length of ti is under the length constraint Lm. Then, function f is said to be a weighted splitting operation iff :(k∪i=1ti⊆t) and (k∑i=1wi≤1). In fact, transaction truncating can be seen as an extreme case of our weighted splitting operation. Suppose a transaction t is divided into subsets t1, ...,tk. If we assign weight1 to one subset ti and assign weights 0 to the other subsets, it is equivalent to truncate t and only keep the items in ti.

6 PFP-GROWTH ALGORITHM

6.1 Algorithm Description of PFP-Growth Algorithm: The PFP-growth algorithm consists of two phases. In particular, in preprocessing phase extract some statistical information from the original database and leverage the smart splitting method to transform the database. Notice that, for a given database, the preprocessing phase is performed only once. For a given threshold, privately find frequent itemsets in the mining phase. Run-time estimation and dynamic reduction methods are used in this phase to improve the quality of the results. Divide the total privacy budget ε into five portions: ε 1 is used to compute the maximal length constraint, ε2 is used to estimate the maximal length of frequent itemsets, ε3 is used to reveal the correlation of items within transactions, ε4 is used to compute μ-vectors of itemsets, and ε5is used for the support computations.

6.1.1Preprocessing Phase

```
Input:
Original database D; Percentage η; Privacy budgetε1,ε2 ,ε3;
Output:
Transformed database D′;
1: α= using ε1 it gets noisy number of transactions with different lengths;
2: Lm= based on α and η gets maximal length constraint Lm;
3: β= get noisy maximal support of itemsets of different lengths usingε2;
4: x= using the μ-vectors of itemsets computes a r×n matrix;
5:D1= enforce length constraint Lm on D by random truncating;
6:Set2= using ε3 compute the noisy support of all 2-itemsets in D1;
7: Create an undirected weighted graph G based onSet2;
8: CR-tree T=Louvain (G, Lm);
9:D′←∅;
10: for each transaction t in D do
11:if|t|> Lm then
12: Sub Transactions ST=Split One Transaction (t, T, Lm);
13: Add each subset in ST with weight 1/|ST| into D′;
14: else
15: Add transaction t into D′;
16: end if
17: end for
18: return D;
```

In the preprocessing phase, we also compute $\beta=\{\beta_1, ...,\beta_n\}$, where β_i is the maximal support of i-itemsets .This array β will be used to estimate the maximal length of frequent itemsets L_f in the mining phase. In practice, however, it is computationally infeasible to exactly compute every element in β. Instead, we select a relatively small threshold and run the FP-growth algorithm. Suppose the maximal length of discovered frequent itemsets is r. For i from 1 tor, we keep the maximal support of i-itemsets β_i. We assume the user-specific threshold is not smaller than this threshold. Since β is a property of the database, we add geometric noise $G(\varepsilon_2/\lceil\log\ n\rceil)$to each β_i .During computing β , we also generate a r×n matrix Z , where row$\{z_i\cdot\}$is the μ-vector of the i-itemset with the highest support. Preprocessing phase is used in run-time estimation method to quantify the information loss caused by transaction splitting.

7. CONCLUSION

Investigating the problem of designing a differentially private FIM algorithm is proposed which consists of a preprocessing phase and a mining phase. Preprocessing phase is used to improve the utility-privacy tradeoff, devise a smart splitting method to transform the database. Run-time estimation method is used in mining phase to offset the information loss incurred by transaction splitting. Moreover, by leveraging the downward closure property, dynamic reduction method is used to dynamically reduce the amount of noise added to guarantee privacy during the mining process. Formal privacy analysis and the results of extensive experiments on real datasets show that our PFP-growth algorithm is time-efficient and can achieve both good utility and good privacy.

REFERENCES

[1] C. Dwork, “Differential privacy,” in ICALP, 2006.
[2] L. Sweeney, “k-anonymity: A model for protecting privacy,” Int. J. Uncertain. Fuzziness Knowl.-Base Syst , 2002.
[3] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubra- maniam, “l-diversity: Privacy beyond k-anonymity,” in ICDE, 2006.
[4] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in VLDB, 1994.
[5] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation,” in SIGMOD, 2000.
[6] C. Zeng, J. F. Naughton, and J.-Y. Cai, “On differentially private frequent itemset mining,” in VLDB, 2012.
[7] Mohan, P., & Thangavel, R. “Resource Selection in Grid Environment based on Trust Evaluation using Feedback and Performance”, American Journal of Applied Sciences, 10(8), 924-930, 2013.
[8] J. Vaidya and C. Clifton, “Privacy preserving association rule mining in vertically partitioned data,” in KDD, 2002.
[9] Annamalai, R., J. Srikanth, and M. Prakash. "Accessing the Data Efficiently using Prediction of Dynamic Data Algorithm." International Journal of Computer Applications 116.22, 2015.
[10] M. Kantarcioglu and C. Clifton, “Privacy-preserving distributed mining of association rules on horizontally partitioned data,” TKDE, 2004.

Author Profile:

M.Jhansi Rani received B.Tech degree in Computer Science and Engineering from BITS-KNL affiliated to JNTUA in 2013 she is currently pursuing towards Masters degree in Computer Science and Engineering from School of Engineering and Technology, SPMVV.

K.Venkatagurunatha Naidu working at SPMVV as Assistant Professor.