IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology
Vol. 3, Issue 6, June 2016

Differentially Private Frequent Item Set Mining Using Smart Splitting and OCCT

Miss. Varsha V. Dabhole[1], Prof. V.S. Nandedkar[2]

PG Student, Computer Engg, PVPIT, Pune, India [1]

Assistant Professor, Computer Engg, PVPIT, Pune, India [2]

Abstract: Frequent item set mining (FIM) is one of the most fundamental problems in data mining. In this paper, a differentially private FIM algorithm is given which can not only achieve high data utility and a high degree of privacy, but also offer high time efficiency. A differentially private FIM algorithm is based on the FP-growth algorithm, which is referred to as PFP-growth. The PFP-growth algorithm consists of two main phases, preprocessing phase and mining phase. In the preprocessing phase, a smart splitting method is used to transform the database. In the mining phase, to cover the information loss caused by transaction splitting, we used a run-time estimation method to estimate the actual support of item sets in the original database. Through formal privacy analysis, we show that our PFP-growth algorithm is differentially private. After this one-to-many data linkage method is used to link different item sets and this method is based on One Class Clustering Tree (OCCT).

Keywords: Frequent Item set Mining, Differential Privacy, Smart Splitting, OCCT, Data Linkage, and Clustering Tree.

I. INTRODUCTION

Frequent Item set Mining (FIM) is one of the most fundamental issues in data mining. It is used in variety of application areas decision support systems, web usage mining etc..

For a given database contains multiple transactions in which each transaction contains a set of items, FIM finds item sets that occur in transactions more frequently than a given threshold value. For the sensitive database such as web browsing history, medical records, releasing the discovered frequent item sets might pose considerable threats to individual privacy.

To solve this problem a new term Differential Privacy has been proposed. Differential privacy gives strong guarantees on the privacy of released data without making assumptions about an attacker's background knowledge. By adding a carefully chosen amount of noise, differential privacy assures that the output is insensitive to changes in any individual's record, and thus restricting privacy leaks through the results.

There are many algorithms for mining frequent item sets. The Apriori and FP-growth [12] are the two most prominent algorithms. Apriori algorithm is a breadth- first search, candidate set generation-and-test. It needs l database scans if the maximal length of frequent item sets is l. In contrast, FP-growth is a depth-first search algorithm, which does not require candidate generation. Compared with Apriori algorithm, FP-growth algorithm performs only two database scans, which makes FP-growth algorithm more faster than Apriori algorithm. An interesting features of FP-growth algorithm motivate us to design a differentially private FIM algorithm based on the FP-growth algorithm.. Differentially private FIM algorithm should not only achieve high data utility and a high degree of privacy, but also offer high time efficiency. The utility-privacy trade off can be improved by limiting the length of transactions. If Apriority-based differentially private FIM algorithm is used, it enforces the limit by truncating transaction method, if a transaction contains more items than the limit, it deletes items until its length is under the limit. In each database scan, to preserve more frequency information, it leverages discovered frequent item sets to re-truncate transactions. Instead, FP-growth performs only two database scans.

There is no way to re-truncate transactions during the mining process. So, the transaction truncating approach proposed in [6] is not suitable for FP-growth. In addition, to avoid privacy breach, we add noise to the support count of item sets. Given an i-item set S (i.e., S contains i items), to satisfy differential privacy, the amount of noise added to the support of i-item set S depends on the number of support computations of i-item sets. Unlike Apriori, FP-growth is a depth-first search algorithm. It is hard to obtain the exact number of support computations of i-item sets during the mining process. A naive approach for computing the noisy support of i-item set X is to use the number of all possible i-item sets. However, it will definitely produce invalid results. To address these challenges, we present our private FP-growth (PFP-growth) algorithm, which consists of a pre-processing phase and a mining phase. In the pre-processing phase, we transform the database to limit the length of transactions. The pre-processing phase is needs to be performed only once for a given database. To limit long transactions, transaction should be split rather than truncated. That is, if a transaction has more items than the limit, we divide it into multiple sub-transactions and each subset is under the limit. A smart splitting method is used to transform the database.

II. LITERATURE SURVEY

Zhi-Hong Deng, Sheng-Long Lv, "PrePost+: An efficient N-lists-based algorithm for mining frequent item sets via Children–Parent Equivalence pruning"[1].
This paper represents PrePost+ a high-performance algorithm for mining frequent item sets. PrePost+ uses N-list to represent item sets and directly detects frequent item sets using a set-enumeration search tree. N-List is a novel data structure proposed in recent years. It is very efficient technique for mining frequent item sets. High performance frequent item set mining algorithm is proposed in this paper. It employs N-List to represent item set and directly discovers frequent item sets using a set enumeration search tree. It uses efficient pruning strategy that is parent-child equivalent pruning to greatly reduce the search space.

Shen and Yu, "Mining frequent graph patterns with differential privacy"[3].
A differentially private frequent graph pattern mining algorithm is proposed in this paper, which does not based on the output of a non-private mining algorithm. Discovering frequent graph patterns in a graph database offers valuable information in a variety of applications. However, if the graph dataset contains sensitive data of individuals such as mobile phone-call graphs and web-click graphs, releasing discovered frequent patterns may present a threat to the privacy of individuals. Differential privacy has recently emerged as the de facto standard for private data analysis due to its provable privacy guarantee. In this paper the first differentially private algorithm for mining frequent graph patterns is proposed. This paper first show that previous techniques on differentially private discovery of frequent item sets cannot apply in mining frequent graph patterns due to the inherent complexity of handling structural information in graphs. Then address this challenge by proposing a Markov Chain Monte Carlo (MCMC) sampling based algorithm.
"PrivBasis: Frequent Item set Mining with Differential privacy" [4].
Privacy to meet the challenge of high dimensionality of transactional database, Liet al. proposed the PrivBasis algorithm which projects the input database onto several sets of dimensions for differentially private top-k frequent item sets mining. The discovery of frequent item sets can serve valuable economic and research purposes. Releasing discovered frequent item sets, however, presents privacy challenges. In this paper, we study the problem of how to perform frequent item set mining on transaction databases while satisfying differential privacy. We propose an approach, called PrivBasis, which leverages a novel notion called basis sets. A θ-basis set has the property that any item set with frequency higher than θ is a subset of some basis. We introduce algorithms for privately constructing a basis set and then using it to find the most frequent item sets.
Zeng C, Naughton JF, Cai J-Y, "On differentially private frequent item set mining"[5].
Zeng C, Naughton JF, Cai J-Y proposed a transaction truncating approach where items in long transactions are deleted until the transactions cardinality is under a specified number. Based on the transaction truncating approach, they present a differentially private FIM algorithm. We consider differentially private frequent item set mining. We begin by exploring the theoretical difficulty of simultaneously providing good utility and good privacy in this task. While our analysis proves that in general this is very difficult, it leaves a glimmer of hope in that our proof of difficulty relies on the existence of long transactions (that is, transactions containing many items). Accordingly, we investigate an approach that begins by truncating long transactions, trading of errors introduced by the truncation with those introduced by the noise added to guarantee privacy. Experimental results over standard benchmark databases show that truncating is indeed effective. Our algorithm solves the "classical" frequent item set mining problem, in which the goal is to find all item sets whose support exceeds a threshold. Related work has proposed differentially private algorithms for the top-k item set mining problem ("find the k most frequent item sets").

III. KEY METHODS AND PROPOSED SYSTEM

A. Differential Privacy

Differential privacy [4] is the standard notion of privacy in data analysis. For two databases D and D', they are neighbouring databases if they differ by at most one record. Formally, the differential privacy is defined as follows.
A private algorithm A satisfies ϵ-differential privacy iff for any two neighbouring databases D and D', and any subset of outputs $S \subseteq$ Range (A),
$\Pr[A(D) \in S] \leq e\, \epsilon \times \Pr[A(D') \in S]$,
where the probability is taken over the randomness of A.

B. Smart Splitting

To improve the utility-privacy trade off, long transactions should be split rather than truncated. For this we transform the database by dividing long transactions into multiple sub-transactions, each of which meets the maximal length constraint. Following algorithm is used for transaction splitting [12].

```
Algorithm 1 : Algorithm For Transaction Splitting
input: Transaction t of length p, CR-Tree CR, Maximum Length Constraint Lm
output: q = |p/Lm| subset.
1. R← Φ;
2. Construct initial nose set Nl;
3. for i from 1 to q do
4. ti ← Φ;
5. Select nose nl with highest number of items from Nl;
6. Add the items from nl to ti, remove nl from Nl;
7. Sort the remaining nodes in Nl;
8. for each node nl' in Nl do 9. if |ti| + |nl'|≤ Lm then 10. Add items in nl' into ti , remove nl' from Nl;
11. End if
12. End for
13. Add ti to R
14. End for
```

```
15. for each nr in Nl do
16. Randomly add items in nr in to the subsets in R;
17. End for
18. return R;
```

C. Run-Time Estimation

Runtime estimation method is used to quantify the information loss caused by transaction splitting. Such information loss comes from two aspects. Suppose a transaction t={a, b, c, d} is divided into t1={a, b} and t2={c, d} with weight w1, w2 respectively. On the one hand, assigning weights makes the support of item sets {a, b} and {c, d} decrease from 1 to w1 and w2. On the other hand, splitting t causes the support of some item sets, such as item set {a, c}, decreases from 1 to 0. To offset the information loss caused by transaction splitting, inspired by the double standards method in [7], propose the run-time estimation method. The method consists of two steps: based on the noisy support of an item set in the transformed database, 1) we first estimate its actual support in the transformed database, and 2) then we further compute its actual support in the original database. For an item set X, let ω denote its noisy support in the transformed database and ω′ denote its actual support in the transformed database. Based on the Bayesian rule, we have Pr (ω′|ω) = Pr (ω|ω′) • Pr (ω′) / Pr (ω). For the information loss caused by transaction splitting, it also depends on how the items in a transaction are partitioned into subsets. Our smart splitting method utilizes the CR-tree to guide the splitting process. However, due to the privacy requirement, we cannot use the CR-tree to quantify the information loss. Our run-time estimation method only depends on differentially private information. In particular, in the first step, to get the probability distribution of ω′, we only need the noisy support ω. In our PFP-growth algorithm, we use the run-time estimation method in the following manner. Suppose the conditional pattern base of item set Y, CPB, is currently being mined.

When we obtain the noisy support of an item i in CPB, we first estimate the average support of i in CPB (i.e., the average support of item set {Y ∪ i}). If this average support exceeds the threshold, we output item set {Y ∪i} as a frequent item set. Then, we further estimate the maximal support of i in CPB (i.e., the maximal support of item set {Y ∪ i}). If this maximal support exceeds the threshold, we insert item i into Y's header table and generate the conditional pattern base of item set {Y ∪ i}.

D. Dynamic Reduction

It is performed in the mining process[12],we should ensure the method would not incur much computational overhead. Our main idea is to leverage the downward closure property (i.e., the supersets of an infrequent item set are infrequent), and dynamically reduce the sensitivity of support computations by decreasing the upper bound on the number of support computations. Then, based on the obtained noisy support, by using our run-time estimation method, we estimate the "maximal" support of item set {Y ∪ i}. If the estimated "maximal" support is smaller than the threshold, we regard i as infrequent items in CPB. Next, we decrease the upper bounds based on the infrequent items found in CPB. Let S2 denote the infrequent items found in CPB. For each item j ∈ S2, item set Z = {Y ∪ j} is infrequent. Based on the downward closure property, it is unnecessary to compute the support of the item sets which are the concatenations of Z with any subsets of {S1 − j}.

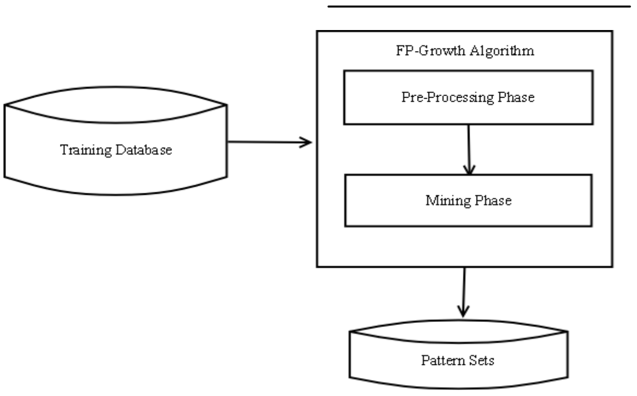
E. Frequent Item set Mining(FIM)

Given a transaction database D[5], is a set of transactions T = {t1, . . ., tm} where each transaction ti is a subset of the alphabet I = {1, . . . , n}. Each subset of the alphabet I is called an item set. If the number of transactions containing an item set exceeds a predefined threshold value, then that item set is called a frequent item set. For any item set X, the support of X is the number of transactions containing X. If that number exceeds a predefined threshold λ, then X is called a frequent item set with respect to the threshold λ. For given transaction database and threshold value, the main goal of FIM if to find complete set of frequent item sets.

F. Differentially Private Frequent Item set Mining Using Smart splitting And OCCT

A workflow of system is shown in Fig. 1.

Phase 1



Phase 2

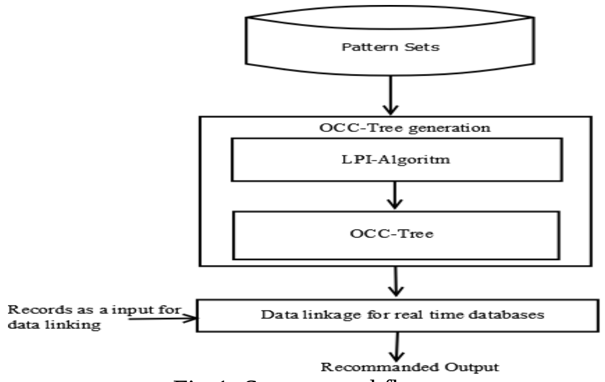

Pattern Sets
OCC-Tree generation
LPI-Algoritm
OCC-Tree
Records as a input for data linking
Data linkage for real time databases
Recommanded Output

Fig 1: System workflow

The system consists of two phases[12]. In the pre-processing phase, we extract some statistical information from the original database and the smart splitting method is used to transform the database. For a given database, the pre-processing phase is performed only once. In the mining phase, for a given threshold, we privately find

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 3, Issue 6, June 2016

frequent item sets. The run-time estimation and dynamic reduction methods are used in this phase to improve the quality of the results. We divide the total privacy budget ϵ into five portions: ϵ1 is used to compute the maximal length constraint, ϵ2 is used to estimate the maximal length of frequent item sets, ϵ3 is used to reveal the correlation of items within transactions, ϵ4 is used to compute μ-vectors of item sets, and ϵ5 is used for the support computations.

Pre processing Phase: In the pre-processing phase, we transform the database to limit the length of transactions. The pre-processing phase is irrelevant to user specified thresholds and needs to be performed only once for a given database. To enforce such a limit, long transactions should be split rather than truncated. That is, if a transaction has more items than the limit, we divide it into multiple subsets.

Mining Phase: In the mining phase, given the transformed database and a user-specified threshold, we privately discover frequent item sets. During the mining process, we dynamically estimate the number of support computations, so that we can gradually reduce the amount of noise required by differential privacy. In the mining phase, to offset the information loss caused by transaction splitting, we devise a run-time estimation method to estimate the actual support of item sets in the original database. Runtime estimation method is used to quantify the information loss caused by transaction splitting.

OCC-tree generation phase [13]: inputs to this phase are pattern sets generated by Phase-1 and records for data linkage and output is the relevancy between items. The phase-II consists of following stages:

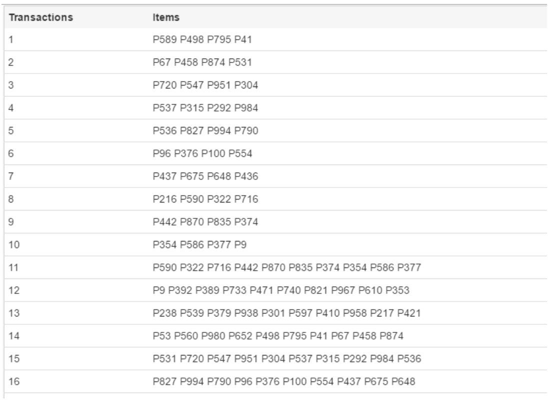
LPI Algorithm: Least probable intersections (LPI) algorithm proposes a distinct combination of attributes as a unique identifier of an entity. The goal is to find a splitting attribute for which there is the least amount of identifiers that are shared, in comparison to a random split of the same size.

OCC-Tree: OCC-Tree consists of attributes which can be derived from the LPI algorithm in tree like structure.
Data linkage for Real Time Databases**:** During the linkage phase, each possible pair of test records is tested against the linkage model in order to determine if the pair is a match. This process produces a score representing the probability of the record pair being a true match.

G. Mathematical Model
Given:
(a) A set B = {i} of items, the item base,
(b) A tuple T = (t1, . . . , i1m, . . . , t) of transactions over B, the transaction database,
(c) A number smin ∈ IN, 0 < s= n, or (equivalently)a number smin ∈ IR, 0 < smin<= 1, the minimum support.
(d) The set of frequent item sets, that is, the set FT (smin) = {I ⊆ B |sT(I) = s} or (equivalently) the set FT(smin)={I⊆B|s
Desired:
(a) The set of frequent item sets, that is, the set FT(smin) = {I ⊆ B |sT(I) = s} or (b) (equivalently) the set FT(smin) = {I ⊆B|s}
Where,
(1)Let B = {i1, . . . , im} be a set of items. This set is called the item base. Items may be products, special equipment items, service options etc. Any subset I ⊆ B is called an item set.
(2)An item set may be any set of products that can be bought (together).
(3) Let T = (t1. . . tn) with ∀k, 1 = k = n : t⊆ B be a tuple of transactions over B. This tuple is called the transaction database.
(4)Let I ⊆ B be an item set and T a transaction database over B.
(5)A transaction t ∈ T covers the item set I or the item set I is contained in a transaction t ∈ T if I ⊆ t.
(6) The set KT (I) = {k ∈ {1, . . . , n}|I ⊆ t} is called the cover of I w.r.t. T.
The value sT (I) = |K(I) |is called the (absolute) support of I w.r.t. T.
The value sT(I) =1nT|K(I) |is called the relative support of I w.r.t. T.

IV. RESULTS

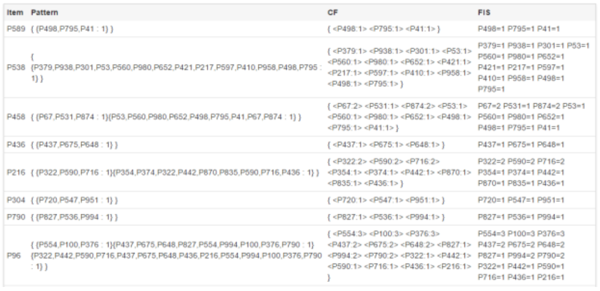
Input data for mining consisting of multiple transactions performed by different users in which each transaction contains multiple products purchased by that user as shown in fig 2.



Transactions	Items
1	P589 P498 P795 P41
2	P67 P458 P874 P531
3	P720 P547 P951 P304
4	P537 P315 P292 P984
5	P536 P827 P994 P790
6	P96 P376 P100 P554
7	P437 P675 P648 P436
8	P216 P590 P322 P716
9	P442 P870 P835 P374
10	P354 P586 P377 P9
11	P590 P322 P716 P442 P870 P835 P374 P354 P586 P377
12	P9 P392 P389 P733 P471 P740 P821 P967 P610 P353
13	P238 P539 P379 P938 P301 P597 P410 P958 P217 P421
14	P53 P560 P980 P652 P498 P795 P41 P67 P458 P874
15	P531 P720 P547 P951 P304 P537 P315 P292 P984 P536
16	P827 P994 P790 P96 P376 P100 P554 P437 P675 P648

Fig 2: Transaction database

After preprocessing and mining phase the system secretly identifies frequent item sets as shown in fig 3.



Item	Pattern	CF	FIS
P589	{ {P498,P795,P41 : 1} }	{ <P498:1> <P795:1> <P41:1> }	P498=1 P795=1 P41=1
P538	{ {P379,P938,P301,P53,P560,P980,P652,P421,P217,P597,P410,P958,P498,P795 : 1} }	{ <P379:1> <P938:1> <P301:1> <P53:1> <P560:1> <P980:1> <P652:1> <P421:1> <P217:1> <P597:1> <P410:1> <P958:1> <P498:1> <P795:1> }	P379=1 P938=1 P301=1 P53=1 P560=1 P980=1 P652=1 P421=1 P217=1 P597=1 P410=1 P958=1 P498=1 P795=1
P458	{ {P67,P531,P874 : 1}{P53,P560,P980,P652,P498,P795,P41,P67,P874 : 1} }	{ <P67:2> <P531:1> <P874:2> <P53:1> <P560:1> <P980:1> <P652:1> <P498:1> <P795:1> <P41:1> }	P67=2 P531=1 P874=2 P53=1 P560=1 P980=1 P652=1 P498=1 P795=1 P41=1
P436	{ {P437,P675,P648 : 1} }	{ <P437:1> <P675:1> <P648:1> }	P437=1 P675=1 P648=1
P216	{ {P322,P590,P716 : 1}{P354,P374,P322,P442,P870,P835,P590,P716,P436 : 1} }	{ <P322:2> <P590:2> <P716:2> <P354:1> <P374:1> <P442:1> <P870:1> <P835:1> <P436:1> }	P322=2 P590=2 P716=2 P354=1 P374=1 P442=1 P870=1 P835=1 P436=1
P304	{ {P720,P547,P951 : 1} }	{ <P720:1> <P547:1> <P951:1> }	P720=1 P547=1 P951=1
P790	{ {P827,P536,P994 : 1} }	{ <P827:1> <P536:1> <P994:1> }	P827=1 P536=1 P994=1
P96	{ {P554,P100,P376 : 1}{P437,P675,P648,P827,P554,P994,P100,P376,P790 : 1}{P322,P442,P590,P716,P437,P675,P648,P436,P216,P554,P994,P100,P376,P790 : 1} }	{ <P554:3> <P100:3> <P376:3> <P437:2> <P675:2> <P648:2> <P827:1> <P994:2> <P790:2> <P322:1> <P442:1> <P590:1> <P716:1> <P436:1> <P216:1> }	P554=3 P100=3 P376=3 P437=2 P675=2 P648=2 P827=1 P994=2 P790=2 P322=1 P442=1 P590=1 P716=1 P436=1 P216=1

Fig 3: Frequent Item sets

Based on the relation count the most relevant items with the relation with other items are shown in fig 4.

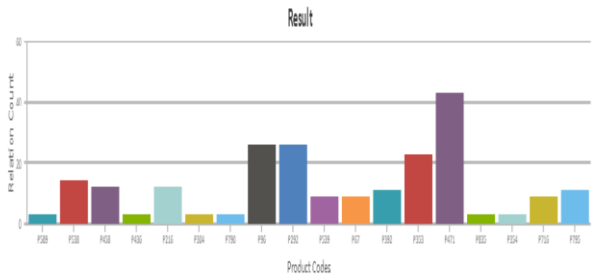
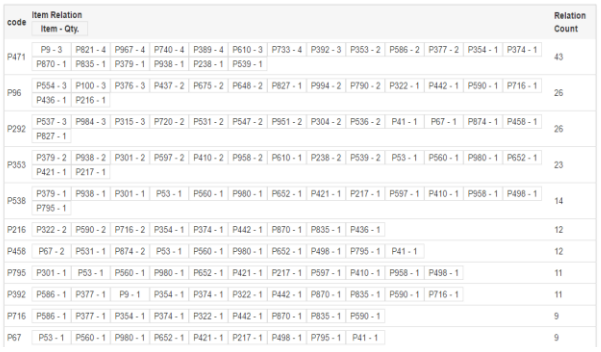


Fig 4: Relevant Items

Based on the relation count the most relevant items with the relation with other items using OCCT are shown in fig 5.



code	Item Relation Item - Qty.	Relation Count
P471	P9 - 3, P821 - 4, P967 - 4, P740 - 4, P389 - 4, P610 - 3, P733 - 4, P392 - 3, P353 - 2, P586 - 2, P377 - 2, P354 - 1, P374 - 1, P870 - 1, P835 - 1, P379 - 1, P938 - 1, P238 - 1, P539 - 1	43
P96	P554 - 3, P100 - 3, P376 - 3, P437 - 2, P675 - 2, P648 - 2, P827 - 1, P994 - 2, P790 - 2, P322 - 1, P442 - 1, P590 - 1, P716 - 1, P436 - 1, P216 - 1	26
P292	P537 - 3, P984 - 3, P315 - 3, P720 - 2, P531 - 2, P547 - 2, P951 - 2, P304 - 2, P536 - 2, P41 - 1, P67 - 1, P874 - 1, P458 - 1, P827 - 1	26
P353	P379 - 2, P938 - 2, P301 - 2, P597 - 2, P410 - 2, P958 - 2, P610 - 1, P238 - 2, P539 - 2, P53 - 1, P560 - 1, P980 - 1, P652 - 1, P421 - 1, P217 - 1	23
P538	P379 - 1, P938 - 1, P301 - 1, P53 - 1, P560 - 1, P980 - 1, P652 - 1, P421 - 1, P217 - 1, P597 - 1, P410 - 1, P958 - 1, P498 - 1, P795 - 1	14
P216	P322 - 2, P590 - 2, P716 - 2, P354 - 1, P374 - 1, P442 - 1, P870 - 1, P835 - 1, P436 - 1	12
P458	P67 - 2, P531 - 1, P874 - 2, P53 - 1, P560 - 1, P980 - 1, P652 - 1, P498 - 1, P795 - 1, P41 - 1	12
P795	P301 - 1, P53 - 1, P560 - 1, P980 - 1, P652 - 1, P421 - 1, P217 - 1, P597 - 1, P410 - 1, P958 - 1, P498 - 1	11
P392	P586 - 1, P377 - 1, P9 - 1, P354 - 1, P374 - 1, P322 - 1, P442 - 1, P870 - 1, P835 - 1, P590 - 1, P716 - 1	11
P716	P586 - 1, P377 - 1, P354 - 1, P374 - 1, P322 - 1, P442 - 1, P870 - 1, P835 - 1, P590 - 1	9
P67	P53 - 1, P560 - 1, P980 - 1, P652 - 1, P421 - 1, P217 - 1, P498 - 1, P795 - 1, P41 - 1	9

Fig 5: Relevancy between items

V. CONCLUSION

A novel technique for the differentially private mining of frequent patterns is studied. FP Growth algorithm is used which consists of pre-processing phase and mining phase to discover frequent item sets. A one class clustering tree method for data linkage is used which contains, a decision tree in which the inner nodes consist only of features describing the first set of entities, while the leaves of the tree represent the features of their matching entities from second database to find most relevant items.

ACKNOWLEDGMENT

I take this golden opportunity to owe our deep sense of gratitude to my project guide **Prof. V. S. Nandedkar,** for her instinct help and valuable guidance with a lot of encouragement throughout this paper work, right from selection of topic work up to its completion. My sincere thanks to the Head of the Department of Computer Engineering **Dr. B. K. Sarkar** sir and **Prof. V.S. Nandedkar** mam, who continuously motivated and guided me for completion of this paper. I am also thankful to all teaching and non-teaching staff members, for their valuable suggestions and valuable co-operation for partially completion of this work. I specially thank to those who helped us directly-indirectly in completion of this work successfully.

REFERENCES

[1] Zhi-Hong Deng, Sheng-Long Lv "PrePost+: An efficient N-lists-based algorithm for mining frequent itemsets via Children–Parent Equivalence pruning", ScienceDirect, 2015.
[2] Toan C. Ong, Michael V. Mannino, Lisa M. Schilling, Michael G. Kahn "Improving record linkage performance in the presence of missing linkage data" ScienceDirect, 2014.
[3] Entong Shen, Ting Yu "Mining Frequent Graph Patterns with Differential Privacy", 2013, p.545-553
[4] Li N, Qardaji WH, Su D, Cao J. "Privbasis: frequent itemset mining with differential privacy". PVLDB 2012; 5(11):1340-51.
[5] Zeng C, Naughton JF, Cai J-Y. "On differentially private frequent itemset mining." PVLDB 2012; 6(1):25-36.
[6] 2015-IEEE, kiran chavan, priyanka kulkarni, pooja ghodekar," Frequent Itemset Mining for Big Data".
[7] 2015-IEEE, kiran chavan, priyanka kulkarni, pooja ghodekar," Frequent Itemset Mining for Big Data".
[8] 2015-IEEE, Patel Tushar S.,"Performance Analysis of Frequent Itemset Finding Techniques using Sparse Datasets".
[9] 2015-IEEE, iyer chandrashekharan, p.k.baruah, ravi mukkamala, "Privacy-Preserving Frequent Itemset Mining in Outsourced Transaction Databases".
[10] Zhi-Hong Deng ,"DiffNodesets: An Efficient Structure for Fast Mining Frequent Itemsets".
[11] A. Friedman and A. Schuster. Data mining with differential privacy. In KDD, pages 493–502, 2010.
[12] A. Gershman et al., "A Decision Tree Based Recommender System," in Proc. the 10th Int. Conf. on Innovative Internet Community Services, pp.170-179, Trondheim, Norway, 2010.
[13] Sen Su, Shengzhi Xu, Xiang Cheng, Zhengyi Li, and Fangchun Yang "Differentially Private Frequent Itemset Mining via Transaction Splitting", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 27, NO. 7, JULY 2015.
[14] Ma'ayan Dror, Asaf Shabtai, Lior Rokach, Yuval Elovici "OCCT: A One-Class Clustering Tree for Implementing One-to-Many Data Linkage" IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, vol 26., march, 2014.

BIOGRAPHIES

Miss. Varsha V. Dabhole Completed his B.E. from Shivaji University, Kolhapur. M.E. perusing from Dept. of Computer Engineering, Padmabhooshan Vasantdada Patil Institute of Technology, Bavdhan, Pune, Maharashtra, India.

Prof. V.S. Nandedkar Assistant Professor, Dept. of Computer Engineering, Padmabhooshan Vasantdada Patil Institute of Technology, Bavdhan, Pune, Maharashtra, India.