Open access Journal International Journal of Emerging Trends in Science and Technology

Secure Mining of Generalized Association Rules from Horizontally Distributed Databases

Authors

Mintu Thomas[1], Neena Joseph[2]

[1]Student, Department of Computer Science and Engineering,
Mangalam College of Engineering, Ettumanoor, Kerala, India
Email: *theresamintu@gmail.com*

[2]Assistant Professor, Department of Computer Science and Engineering,
Mangalam College of Engineering, Ettumanoor, Kerala, India
Email: *nestofneena@gmail.com*

Abstract

We consider the problem of secure mining of generalized association rules from horizontally distributed databases. Given a large horizontally distributed database of transactions, where each transaction consists of a set of items and taxonomy on the items, we find associations between items at any level of the taxonomy. Generalized association rule mining technique has discussed in many papers. But, in this paper we discuss about secure mining of generalized association rules from horizontally distributed databases or homogeneous databases. For that purpose, we use the same privacy preserving distributed mining concepts discussed in paper [1] with the generalized association rule mining technique called 'cumulate' algorithm discussed in paper [2]. The main privacy preserving parts of the protocol in paper [1] are two secure multi-party algorithms called UNIFI and SETINC. Our proposed protocol is based on Fast Distributed Mining (FDM) algorithm. FDM algorithm is an unsecured distributed version of Apriori algorithm. It offers enhanced privacy, simplicity and efficiency.

Keywords: *Privacy Preserving Data Mining, Horizontally Distributed Databases, Generalized Association ruldes, Frequent Itemsets.*

1. Introduction

Secure mining of generalized association rules from horizontally partitioned databases is an important problem in the area of data mining. Secure mining of association rule means mining association rules from different sites (or players) without sharing their individual information. Here, we consider the case of horizontally distributed databases. That is, here the databases are homogeneous. Homogeneous databases mean databases that share the same schema but hold information on different entities. If a database *D* is partitioned between *M* players, then that setting is called distributed databases. If database *D* is horizontally partitioned between *M* players, then that setting is called horizontally partitioned (or horizontally distributed) databases. Where, each player holds one partial database. For example, let *D* be a transaction database, where each row represents a transaction and each column represents one of the items in the transaction. Then a horizontal partitioning of that database leads to the generation of different partial databases. In each of these partial databases, the columns are same but transaction rows are different.

Our goal is to find all generalized association rules with support at least *s* and confidence at least *c* that hold in the unified database (unified database means union of all partial databases). That is, given a set of transactions and taxonomy, we want to find association rules where the items may be from any

level of the taxonomy. That goal defines the problem of secure multi-party computation. For solving such problems, if there existed a trusted third party, the players could surrender to him their inputs and then he would perform the required computations and return to them the resulting output. But, in the absence of such a trusted third party, we need a protocol that the players can run on their own and which provides the required output at the end of computation. That protocol is said to be perfectly secure, if the players could not extract, from their view of the protocol, information on other databases. Yao [13] was the first to propose a generic solution for this problem in the case of two players. Other generic solution for multi-party case was discussed in [5].

Association rule shows relation among different items or itemsets. An association rule is an implication of the form $A ==> B$. Where, $A \subseteq I$, $B \subseteq I$, $A \cap B = \emptyset$ and I is the set of all items. Which hold in the transaction database D with support s and confidence c. Support of an association rule, $A ==> B$ is the percentage of transactions in D, that contain both A and B. It is also defined as the probability, $P(A \cup B)$. Confidence of an association rule, $A ==> B$ is the percentage of transactions in D, that contain A, also contains B.

2. Related Works

T. Tassa, in [1] proposed a protocol for secure mining of association rules from horizontally distributed databases. This is the privacy enhanced version of the Kantarcioglu and Clifton protocol discussed in [3]. Both are based on FDM algorithm. Within the protocol of Kantarcioglu and Clifton, there occurs a problem of excess information leakage at the time of computation of union of locally frequent private subsets held by different players. So, in order to overcome that problem, Tamir Tassa in [1] proposed that protocol. Its main ingredients are two secure multi-party algorithms called UNIFI and SETINC. Where, UNIFI computes the union of locally frequent private subsets held by different players and SETINC tests the inclusion of an item held by one player in a subset held by another player.

The main part of Kantarcioglu and Clifton protocol [3] is a sub-protocol for the secure computation of the union of private subsets that are held by the different players. That is the most costly part of the protocol and its implementation requires cryptographic techniques such as commutative encryption, oblivious transfer and hash functions. That is, here, the main idea is that each site encrypts the locally supported itemsets, along with enough "fake" itemsets to hide the actual number supported. Each site then encrypts the itemsets from other sites. This is the phase 1 of that protocol. Then, in phases 2 and 3, the encrypted itemsets are merged. Phase 4 decrypts the merged frequent itemsets. This is also the only part in the protocol in which the players may extract from their view of the protocol excess information on other databases. This method assumes three or more parties. But, [1] works in the case of any number of parties and it offers improved simplicity, efficiency and privacy.

The most basic algorithms for mining generalized association rules are those proposed by R. Srikant and R. Agrawal in [2]. Given a large database of transactions, where each transaction consists of a set of items, and taxonomy on the items, then the proposed algorithm find association between items at any level of the taxonomy. An obvious solution to the above problem is to add all ancestors of each item in a transaction to the transaction, and then run any of the algorithms for mining association rules on these new transactions. This is the Basic algorithm. But, it is not very fast. So, this paper presents two other algorithms called Cumulate and Est Merge, which run 2 to 5 times faster than Basic. This paper also presents a new interest-measure for rules which uses information in the taxonomy. That is, given a user-specified "minimum-interest-level", this measure prunes a large number of redundant rules.

The most basic and fast algorithms for mining association rules are Apriori, Apriori Tid and Apriori Hybrid [4]. The databases involved in these applications are very large. In all of these algorithms we first find all itemsets that have transaction support above minimum support. Itemsets with minimum support are called large itemsets and all others are called small itemsets. Then, in the next

step we use these large itemsets to generate the desired rules. In the Apriori algorithm, the first pass simply counts item occurrences to determine the large 1-itemsets. Any subsequent pass, say pass k consists of two phases. In the first phase the large itemsets found in the (k-1)th pass are used to generate the candidate itemsets C_k. In the second phase, the database is scanned and the support of candidates in C_k is counted. The Apriori Tid algorithm is same as Apriori but has the additional property that the database is not scanned after the first pass for counting support of candidate itemsets. Rather, an encoding of the candidate itemsets used in the previous pass is employed for this purpose. Best features of Apriori and Apriori Tid can be combined into a hybrid algorithm, called Apriori Hybrid. It has excellent scale-up properties.

For mining association rules from distributed databases, we require distributed mining algorithms. Most basic of such algorithms is FDM algorithm [5]. Within this algorithm, we can see that some interesting relationships between locally large and globally large itemsets. It generates a small number of candidate sets and reduces the number of messages to be passed at the time of mining association rules. Here, after the candidate sets have been generated, two pruning techniques called local pruning and global pruning are developed to prune away some candidate sets at each individual site. Three versions of FDM such as FDM-LP, FDM-LUP and FDM-LPP are also discussed in this paper.

The privacy preserving association rule mining technique discussed in [6] contains a framework for mining association rules. This framework mine rules from transactions consisting of categorical items where the data has been randomized to preserve privacy of individual transactions. That is, this paper proposes a class of randomization operators for limiting privacy breaches found with uniform randomization and thus preserving privacy. But, J. Vaidya and C. Clifton in [7] propose a privacy preserving association rule mining technique for the case of vertically partitioned databases. The key privacy-preserving part of that technique is the component scalar product protocol.

3. Proposed Method

3.1 Preliminaries

A generalized association rule is an implication of the form $A ==> B$, where $A \subseteq I$, $B \subseteq I$, $A \cap B = \emptyset$, and no item in B is an ancestor of any item in A. Finding generalized association rule is necessary because rules that found at lower levels of the taxonomy may not have minimum support count. This will leads to the missing of many significant associations among items at the leaves of the taxonomy. Another reason for finding generalized association rule is that taxonomies can be used to prune uninteresting and redundant rules.

For the mining of generalized association rules, we consider a set of transactions of items (database D) and taxonomy on those items. Here, each transaction is denoted by T and taxonomy is denoted by T_x. Fig. 1 shows an example of taxonomy. Taxonomies are represented by using a directed acyclic graph (DAG). In these taxonomies, an edge represents an *is-a* relationship and a node represents one of the items in the given item set, I. If there exist an edge from p to c, then we call p is the parent and c is the child. Where, p is also called as the generalization of c. Let, x denote an item in the taxonomy, we call x^* is the ancestor of x if there exist an edge from x^* to x. But, a node is not an ancestor of itself. We say that a transaction T supports an item $x \in I$ if x is in T or x is an ancestor of some item in T.

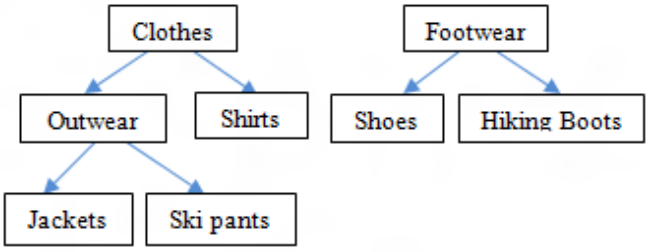

Fig. 1: Example of taxonomy

Here, we mine generalized association rules from horizontally distributed databases. So, first we view the database D as a binary matrix of N rows and L columns. Where each row represents a transaction over some set of items, $I = \{a_1, a_2, \ldots, a_L\}$, and each column represents one of the items in I. That is, the (i, j)th entry of the database D is equal to 1 if transaction i includes the item a_j and 0 otherwise.

Then, D is partitioned horizontally between M players, denoted as P_1, P_2,…..P_M. Now player P_m holds the partial database D_m, where $1 \leq m \leq M$ and the unified database is $D = D_1 \cup ..... \cup D_M$. Let, s denotes the support threshold of itemsets where $0 \leq s \leq 1$, then, an itemset A is called s-frequent if $supp(A) \geq sN$. It is called locally s-frequent at partial database D_m if $supp_m(A) \geq sN_m$.

3.2 Problem Definition

Given a transaction database D that is horizontally distributed across M players and a set of taxonomies T_x, our goal is to find all generalized association rules from the unified database D. These rules should satisfy the minimum support threshold s and the minimum confidence threshold c.

Let k be a real number between 1 and L, then F^k_s denote the set of all s-frquent$k$-itemsets and $F^{k,m}_s$ denote the set of all locally s-frequent k-itemsets at D_m. So, our main computational goal is to find the set $G^{k,m}_s$ which is the resulting set that obtained after applying the concept of 'cumulate' algorithm on locally frequent itemsets of each players. Then we can find $G^k_s = \cup^M_{m=1} G^{k,m}_s$. From this, we can find F^k_s and F_s. Finally, we can derive generalized association rules from this F_s.

3.3 Mining generalized association rules from horizontally distributed databases

Our proposed protocol is based on FDM algorithm. That is, here the mining procedure is goes through the steps of FDM algorithm [5]. FDM is an unsecured distributed version of Apriori algorithm. The main idea of FDM is that any s-frequent itemset must be also locally s-frequent in at least one of the sites (or players). So, in order to find out the globally s-frequent itemsets, each player must find out his locally s-frequent itemsets first and then check whether it is globally s-frequent. The union of these locally s-frequent itemsets gives us the globally s-frequent itemsts. In this procedure, to preserve privacy at the time of union computation we apply UNIFI and SETINC algorithms [1] proposed by T. Tassa. But, in order to derive generalized association rules from globally s-frequent itemsets at the end, we need to apply 'cumulate' algorithm on locally s-frequent itemsets of each player. For that purpose, here, we add an additional step to the FDM algorithm. This procedure proceeds as follows.

Step 1: This is the initialization step. In this step, we assume that each player jointly computes the set F^{k-1}_s where, k denotes the pass number. But, our goal is to calculate F^k_s.

Step 2: This is the step where candidate itemsets are generated. For that purpose each player P_m computes the set of all (k-1) itemsets which are both locally and globally s-frequent. They then applies Apriori algorithm on that sets and thus generate k-itemsets called $B^{k,m}_s$.

Step 3: In this step, each player applies local pruning on their candidate itemsets. That is, for each $A \in B^{k,m}_s$, P_m computes support count of A within it. After that each player maintains only those item sets which are locally s-frequent. We denote these item sets by $C^{k,m}_s$.

Step 4: This is the most significant step in our proposed solution. In this step we derive locally s-frequent generalized itemsets called $G^{k,m}_s$ from the candidate itemsets $C^{k,m}_s$, generated in the previous step. For that purpose, the concept of one of the generalized association rule mining algorithm called 'cumulate' algorithm [2] is used. In our solution, this algorithm is modified as follows.

Algorithm 1: cumulate-m

Compute T^a_x, the set of ancestors of each item from T_x.
$G^{1,m}_s = C^{1,m}_s$ { frequent 1-itemsets };
while ($G^{k-1,m}_s \neq \emptyset$) do
begin
$C^{k,m}_s$ = candidates generated from $G^{k-1,m}_s$.
if(k = 2) then
Delete itemsets in $C^{2,m}_s$ that consists of an item and its ancestor.
Delete ancestors in T^a_x, that not present in any of the itemsets in $C^{k,m}_s$.

for all transactions $T \in D_m$ do
begin
for each item $a \in T$ do
Add all ancestors of a in T^a_x to T.
Remove duplicates from T.
Increment the count of all items in $C^{k,m}_s$, that contained in T.
end
$G^{k,m}_s$ = all items in $C^{k,m}_s$ with minimum support.
end

This algorithm is used by each player to derive their locally s-frequent generalized k-itemsets, $G^{k,m}_s$. Since it is a modified version, we call it as 'cumulate-m'.

Step 5: During this step, candidate itemsets held by different players are unified. For unifying the candidate itemsets, each player broadcasts his $G^{k,m}_s$ and then all players compute $G^k_s = \cup^M_{m=1} G^{k,m}_s$. This is, one of the steps in FDM algorithm where, privacy is violated. So, in order to preserve privacy efficiently and thus to enable secure mining of generalized association rules we can apply the same privacy preserving concepts discussed by T. Tassa in [1]. Where, two secure multi-party algorithms called UNIFI and SETINC are used for that purpose. In UNIFI, the main idea is that each player encodes his subset $G^{k,m}_s$ as a binary vector b_m of length $n_k = | Ap(F^{k-1}_s) |$. Where, $Ap(F^{k-1}_s)$ means the resulting set obtained after applying the Apriori algorithm on set F^{k-1}_s. This binary vector is then broadcasted to all other players instead of original itemsets and then, each player computes the union of these binary vectors. But, the SETINC algorithm is used to test the inclusion of an item held by one player in a subset held by another player.

Step 6: In this step, all players compute the local support counts of all itemsets in G^k_s.

Step 7: During this step, each player broadcasts his local support counts. From this, every player can compute the global support counts of all itemsets in G^k_s. This is another step in FDM algorithm where privacy is violated. So, in order to overcome this disadvantage we can use the same privacy preserving concepts discussed by Kantarcioglu and Clifton in [3]. Where, some fake itemsets are added to the original itemset to hide the actual number supported. In this step, finally we compute all globally s-frequent k-itemsets called F^k_s, which is the subset of G^k_s.

The above procedure proceeds iteratively until it finds the longest globally s-frequent itemsets. If the length of such itemsets is k, then the $(k+1)$th pass will not find any $(k+1)$- itemsets. Finally, we can easily derive all generalized (s,c)-association rules from these globally s-frequent generalized itemsets.

4. Performance Evaluation

Here, we compare the performance of our distributed generalized association rule mining algorithm with FDM algorithm and with Kantarcioglu and Clifton algorithm. For that purpose, we denote our algorithm by FDM-G and Kantarcioglu and Clifton algorithm by FDM-KC. In addition to this, we consider two parameters such as N and M to denote the number of transactions in the unified database and the number of players respectively. Experiments show that total computation time of FDM-G is much higher than FDM and lower than FDM-KC (see Fig. 1 and Fig. 2). We can also see that computation time of all the three algorithms increases with the number of transactions and number of players.

Total computation time

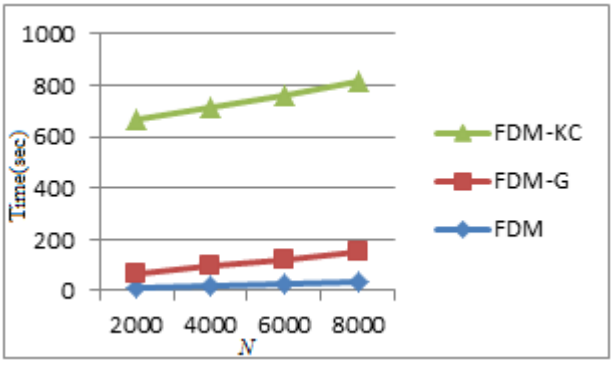

Fig 2: Computation costs versus the number of transactions N

Total computation time

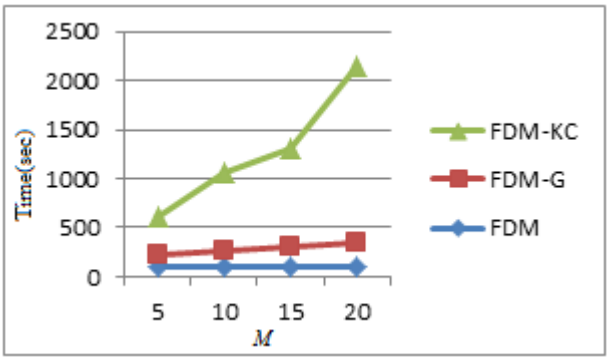

Fig. 3: Computation costs versus the number of players *M*

5. Conclusion

We proposed a protocol for secure mining of generalized association rules from horizontally distributed databases. This protocol is based on FDM algorithm. That is, our protocol proceeds through the steps of FDM algorithm. Here, for mining generalized association rules, we add an additional step to the FDM algorithm. This step works by using the concept of 'cumulate' algorithm [2] for generalized association rule mining. But, for preserving privacy, we apply UNIFI and SETINC algorithms [1] on the next step. The concept of Kantarcioglu and Clifton protocol [3] is also used for that purpose. Therefore, the computational cost is much higher in our solution. But, one of the advantages of our protocol is that it offers enhanced privacy.

References

1. T. Tassa, "Secure mining of association rules in horizontally distributed databases", IEEE Transactions on Knowledge and Data Engineering, April 2014.
2. R. Srikant and R. Agrawal, "Mining generalized association rules", InVLDB, pages 407–419, 1995.
3. M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data", IEEE Transactions on Knowledge and Data Engineering, 16:1026–1037, 2004.
4. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases", In VLDB, pages 487–499, 1994.
5. D.W.L. Cheung, J. Han, V.T.Y. Ng, A.W.C. Fu, and Y. Fu, "A fast distributed algorithm for mining association rules", In PDIS, pages 31–42, 1996.
6. A.V. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy preserving mining of association rules", In KDD, pages 217–228, 2002.
7. J. Vaidya and C. Clifton, "Privacy preserving association rule mining in vertically partitioned data", In KDD, pages 639–644, 2002.
8. J. Zhan, S. Matwin, and L. Chang, " Privacy preserving collaborative association rule mining", In Data and Applications Security, pages 153–165, 2005.
9. R. Agrawal and R. Srikant, "Privacy-preserving data mining", In SIGMOD Conference, pages 439–450, 2000.
10. M. Kantarcioglu, R. Nix, and J. Vaidya, " An efficient approximate protocol for privacy-preserving association rule mining", In PAKDD, pages 515–524, 2009.
11. J.S. Park, M.S. Chen, and P.S. Yu, "An effective hash based algorithm for mining association rules", In SIGMOD Conference, pages 175–186, 1995.
12. A. Schuster, R. Wolff, and B. Gilburd, "Privacy-preserving association rule mining in large-scale distributed systems", In CCGRID, pages 411– 418, 2004.
13. A.C. Yao, "Protocols for secure computation", In FOCS, pages 160–164, 1982.
14. A. Ben-David, N. Nisan, and B. Pinkas, "FairplayMP - A system for secure multi-party computation", In CCS, pages 257–266, 2008.
15. H. Grosskreutz, B. Lemmen, and S. R¨uping, "Secure distributed subgroup discovery in horizontally partitioned data", Transactions on Data Privacy, 4:147–165, 2011.