Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 1, January 2015, pg.94 – 101

RESEARCH ARTICLE

Privacy Preserving and Secure Mining of Association Rules in Distributed Data Base

N.Kowsalya[#1], M. Saraswathi[*2]

[#1]Assistent Professor, Dept of Computer Science & Applications
[*2]M.Phil Full Time Research scholar, Department of Computer Science
Vivekanandha College of Arts and Sciences for Women (Autonomous), Namakkal, TamilNadu, India
[2] saraswathi.yms@gmail.com

Abstract- Association rule mining is an active data mining research area and most ARM algorithms cater to a centralized environment. Centralized data mining to discover useful patterns in distributed databases isn't always feasible because merging data sets from different sites incurs huge network communication costs. In this paper, an improved algorithm based on good performance level for data mining is being proposed. Local Site also finds a centre site to manage every message exchanged to obtain all globally frequent item sets. It also reduces the time of scan of partition database. The problem of computing efficient anonymizations of partitioned databases. Given a database that is partitioned between several sites, either horizontally or vertically, we devise secure distributed algorithms that allow the different sites to obtain a k-anonymized and ℓ-diverse view of the union of their databases, without disclosing sensitive information. Without leaking any information about their inputs except that revealed by the algorithm's output. Working in the standard secure multi-party computation paradigm, we present new algorithms for privacy-preserving computation of APSD (all pairs shortest distance) and SSSD (single source shortest distance), as well as two new algorithms for privacy-preserving set union. We prove that our algorithms are secure provided the participants are "honest, but curious."

Keywords: Secure Multiparty Computation, privacy-preserving, databases partitioning

I. INTRODUCTION

Most existing parallel and distributed ARM algorithms are based on a kernel that employs the well-known Apriori algorithm [1]. Directly adapting an Apriori algorithm will not significantly improve performance over frequent item sets generation or overall distributed ARM performance. In distributed mining, synchronization is implicit in message passing, so the goal becomes communication optimization. Data decomposition is very important for distributed memory[2]. Therefore, the main challenge for obtaining good performance on distributed mining is to

find a good data decomposition among the nodes for good load balancing, and to minimize communication. Protecting the privacy of the individuals whose private data appear in those repositories is of paramount importance. Al- though identifying attributes such as names and ID numbers are always removed before releasing the table for data mining purposes, sensitive information might still leak due to *linking attacks*; such attacks may join the public attributes, a.k.a *quasi-identifiers*, of the published table with a publicly accessible table like the voters registry, and thus disclose private Information of specific individuals. Privacy-preserving data mining [3] has been proposed as a paradigm of exercising data mining while protecting the privacy of individuals. One of the well-studied models of privacy preserving data mining is *k*-anonymization [4,5]. Trusted third party, each site could surrender to that third party his part of the database and trust the third party to compute an anonymization of the unified database. Without such a trusted third party, the goal is to devise distributed protocols, for the horizontal and vertical settings, that allow the data holders to simulate the operation of a trusted third party and obtain a k-anonymized and ℓ-diverse view of the union of their databases, without disclosing unnecessary information to any of the other parties, or to any eavesdropping adversary. In this paper, we construct privacy-preserving versions of classic graph algorithms for APSD (all pairs shortest distance) and SSSD (single source shortest distance). Our algorithm for APSD is new, while the SSSD algorithm is a privacy preserving transformation of the standard Dijkstra's algorithm. We also show that minimum spanning trees can be easily computed in a privacy-preserving manner.

II. RELATED WORKS

This paper follows a long tradition of research on privacy-preserving algorithms in the so called secure multiparty computation (SMC) paradigm. Informally, security of a protocol in the SMC paradigm is defined as computational indistinguishability from some ideal functionality, in which a trusted third party accepts the parties' inputs and carries out the computation. The ideal functionality is thus secure by definition. The actual protocol is secure if the adversary's view in any protocol execution can be simulated by an efficient simulator who has access only to the ideal functionality, i.e., the actual protocol does not leak any information beyond what is given out by the ideal functionality. In this paper, we aim to follow the SMC tradition and provide provable cryptographic guarantees of security for our constructions. Another line of research has focused on statistical privacy in databases, typically achieved by randomly perturbing individual data entries while preserving some global properties. A survey can be found in The proofs of security in this framework are statistical rather than cryptographic in nature, and typically permit some leakage of information, while supporting more efficient constructions. In this paradigm, Clifton et al. have also investigated various data mining problems, while Du et al. researched special-purpose constructions for problems such as privacy-preserving collaborative scientific analysis. Recent work by Chawla et al. aims to bridge the gap between the two frameworks and provide rigorous cryptographic definitions of statistical privacy in the SMC paradigm. Another line of cryptographic research on privacy focuses on private information retrieval (PIR), but the problems and techniques in PIR are substantially different from this paper.

III. DEFINITION OF PRIVACY

We use a simplified form of the standard definition of security in the static semi-honest model due to Goldreich (this is the same definition as used, for example, by Lindell and Pinkas).

Definition 1. Protocol π securely computes deterministic functionality f in the presence of static semi-honest adversaries if there exist probabilistic polynomial time simulators S1 and S2 such that

$\{S1(x, f(x, y))\}_{x,y \in \{0,1\}} \equiv c$

$\{view_{\pi 1}(x, y)\}_{x,y \in \{0,1\}}$

$\{S2(y, f(x, y))\}_{x,y \in \{0,1\}} \equiv c$

$\{view_{\pi 2}(x, y)\}_{x,y \in \{0,1\}}$

where $|x| = |y|$.

EXISTING SYSTEM

Kantarcioglu and Clifton studied that problems and devised a protocol for its solution. The main part of the protocol is a sub-protocol for the secure computation of the union of private subsets that are held by the different players. (The private subset of a given player, as we explain below, includes the item sets that are s-frequent in his partial database. That is the most costly part of the protocol and its implementation relies upon cryptographic primitives such as commutative encryption, oblivious transfer, and hash functions. This is also the only part in the protocol in which the players may extract from their view of the protocol information on other databases, beyond what is implied by the final output and their own input. While such leakage of information renders the protocol not perfectly secure, the perimeter of the excess information is explicitly bounded and it is argued there that such information leakage is innocuous, whence acceptable from a practical point of view.

Disadvantages of Existing System

Insufficient security, simplicity and efficiency are not well in the databases, not sure in privacy in an existing system.

While our solution is still not perfectly secure, it leaks excess information only to a small number (three) of possible coalitions, unlike the protocol of that discloses information also to some single players. Our protocol may leak is less sensitive than the excess information leaked by the protocol.

PROPOSED SYSTEM

The protocol that we propose here computes a parameterized family of functions, which we call threshold functions, in which the two extreme cases correspond to the problems of computing the union and intersection of private subsets. Those are in fact general-purpose protocols that can be used in other contexts as well. Another problem of secure multiparty computation that we solve here as part of our discussion is the set inclusion problem; namely, the problem where Alice holds a private subset of some ground set, and Bob holds an element in the ground set, and they wish to determine whether Bob's element is within Alice's subset, without revealing to either of them information about the other party's input beyond the above described inclusion.

Mining Association Rules Efficient algorithms for mining frequent itemsets are crucial for mining association rules as well as for many other data mining tasks. Methods for mining frequent itemsets have been implemented using a prefix-tree structure, known as an FP-tree, for storing compressed information about frequent itemsets. Numerous experimental results have demonstrated that these algorithms perform extremely well. In this paper, we present a novel FP-array technique that greatly reduces the need to traverse FP-trees, thus obtaining significantly improved performance for FP-tree-based algorithms. Our technique works especially well for sparse data sets. Furthermore, we present new algorithms for mining all, maximal, and closed frequent itemsets. The results show that our methods are the fastest for many cases. Even though the algorithms consume much memory when the data sets are sparse, they are still the fastest ones when the minimum support is low. The L-Matrix Algorithm Algorithm L-Matrix minimizes the communication overhead. Our solution also reduces the size of average transactions and datasets that leads to reduction of scan time. It minimizes the number of candidate sets and exchange messages by local and global pruning. Reduces the time of scan partition databases to get support counts by using a compressed matrix-L-Matrix, which is very effective in increasing the performance. Finds a centre site to manage every the message exchanges to obtain all globally frequent item sets, only O(n) messages are needed for support count exchange.

It has superior running efficiency, lower communication cost and stronger scalability that direct application of a sequential algorithm in distributed databases. This new algorithm LMatrix is used to achieve maximum efficiency of algorithms. The transaction database is first created to develop the L-Matrix. A LMatrix is an object-by-variable compressed structure. Transaction database is a binary matrix where the rows represent transactions and columns represent alarms. The partitioned databases need to be scanned only once to convert each of them to the local LMatrix. The local LMatrix is read to find support counts instead of scanning the partition databases time after time, which will save a lot of memory. The proposed algorithm can be applied to the mining of association rules in a large centralized database by partitioning the database to the nodes of a distributed system. This is particularly useful if the data set is too large for sequential mining. Informally, this definition says that each party's view of the protocol can be efficiently simulated given only its private input and the output of the algorithm that is being computed (and, therefore, the protocol leaks no information to a semi-honest adversary beyond that revealed by the output of the algorithm). Generality. Our approach applies to both horizontal and vertical partitionings, whereas the approaches in are restricted to only one of those settings. Our approach applies to any number of data sites, where the entailed communication cost depends linearly on the number of sites. The sequential algorithm, on which our

protocols are based, may be applied with any utility measure and it applies to any generalization technique (unlike which works with generalization by suppression only, that work only with global recoding generalizations). Our protocols support additional privacy measures such as ℓ-diversity and ℓ-site diversity. • Simplicity. While previous solutions invoke costly cryptographic primitives such as homomorphic encryptions, oblivious transfers, and group exponentiations, the only cryptographic primitives that we need are an SMC protocol for computing sums, and a secure hash function. • Efficiency. Our protocols are practical and efficient. We analyze the communication complexity of our protocols and then conduct experiments that illustrate the dependence of the communication costs on different parameters in several datasets, both in the horizontal and vertical settings. • Privacy. We analyze the privacy of our distributed protocols and show that, even though they are not perfectly secure in the cryptographic sense, they leak very little and benign information. Such a compromise is widely acceptable when the information leakage is deemed innocuous and the gain in utility, efficiency, and practicality is significant. • Utility. Perhaps the most important advantage offered by our protocols is the utility of the resulting anonymizations. The sequential algorithm is currently one of the leading *k*-anonymization algorithms in terms of the utility of its output. In particular, it offers anonymizations with significantly smaller information losses than those offered by the algorithms on which the distributed solutions in [23, 24, 34, 50] are based. Other contributions of this study are two novel generic SMC protocols. The first one is a simple SMC protocol for the computation of the AND (or the OR) of private bits held by the different players. That protocol gives rise to a simple protocol for secure computation of set intersections and unions. The second SMC protocol computes the least common ancestor of private nodes in a tree. To the best of our knowledge, that problem was not studied before, and it may be of interest in other applications as well.

Advantages of Proposed System:

We proposed a protocol for secure mining of association rules in horizontally distributed databases that improves significantly upon the current leading protocol in terms of privacy and efficiency. The main ingredient in our proposed protocol is a novel secure multi-party protocol for computing the union (or intersection) of private subsets that each of the interacting players holds.

IV. ANONYMIZATION BY GENERALIZATION

Consider a database that holds information on individuals in some population. Each record in the database has several attributes, and we distinguish between identifiers, quasi-identifiers, and sensitive attributes. Identifiers are attributes that uniquely identify the individual, e.g. name or id. Quasi-identifiers are attributes, such as age or zip code that appear also in publicly-accessible databases and may be used in order to identify a person. The sensitive attributes are those that carry private information like a medical diagnosis or the salary of the person. *k*-Anonymity is a model that was proposed in order to prevent the disclosure of sensitive attributes for the purpose of protecting the privacy of individuals that are represented in the database. we view the database records as elements in $A1 \times \cdots \times Ad \times Ad{+}1$, where Aj is the set of possible values for the jth attribute; say, if the jth attribute is gender then $Aj = \{M, F\}$. Hereinafter, D denotes the projection of the database on the set of d quasi-identifiers and the records of D are denoted Ri, $1 \leq i \leq n$; namely, $Ri \in A1 \times \cdots \times Ad$. We denote the jth component of the record Ri by $Ri(j)$. Also, for any set A we let $P(A)$ denote its power set. Next, we define the notion of generalization. Definition 2.1. *Let* Aj , $1 \leq j \leq d$, *be finite sets and let* $Aj \subseteq P(Aj)$ *be a collection of subsets of* Aj . *A mapping* $g : A1 \times \cdots \times Ad \rightarrow A1 \times \cdots \times Ad$ *is called a generalization if for every* $(b1, \ldots, bd) \in A1 \times \cdots \times Ad$ *and* $(B1, \ldots, Bd) = g(b1, \ldots, bd)$, *it holds that* $bj \in Bj$, $1 \leq j \leq d$. As an example, consider a database D with two attributes, age ($A1$) and zipcode ($A2$). A valid generalization of the record Ri = (34, 98003) can be $g(34, 98023) = (\{30, \ldots, 39\}, \{98000, \ldots, 98099\})$. We assume here that each of the collections Aj is a generalization hierarchy tree for Aj , $1 \leq j \leq d$. Such a tree has $|Aj|$ leaves – one for each singleton subset of Aj ; the root corresponds to the whole set; and the subset of each node is the union of the subsets that correspond to the direct descendants of that node. Definition 2.1 refers to generalizations of single records. We now define generalizations of an entire database.

A. Private Single Source Shortest Distance (SSSD)

The Single Source Shortest Distance (SSSD) problem is to find the shortest path instances from a source vertex *s* to all other vertices [11]. An algorithm to solve APSD also provides the solution to SSSD, but leaks additional information beyond that of the SSSD solution and cannot be considered a private algorithm for SSSD. Therefore, this problem warrants its own investigation. Similar to the protocol of section 5.1, the SSSD protocol on the minimum joint graph adds edges in order from smallest to largest. This protocol is very similar to Dijkstra's algorithm, but is modified to take two graphs as input.

1. Set $w(0)\ 1 = w1$ and $w(0)\ 2 = w2$. Color all edges incident on the source s blue by putting all edges esi into the set $B(0)$. Set the iteration count k to 1.
2. Both parties privately compute the minimum length of blue edges in their
graphs. $m(k)\ 1 = \min_{esi \in B(k-1)} w(k-1)\ 1\ (esi)$, $m(k)\ 2 = \min_{esi \in B(k-1)} w(k-1)\ 2\ (esi)$ 3. Using the privacy-preserving minimum protocol, compute
$m(k) = \min(m(k)\ 1, m(k)\ 2)$.
4. Each party finds the set of blue edges in its graph with length $m(k)$.
$S(k)\ 1 = \{esi | w(k-1)\ 1\ (esi) = m(k)\}$, and $S(k)\ 2 = \{esi | w(k-1)\ 2\ (esi) = m(k)\}$
5. Using the privacy-preserving set union protocol, compute
$S(k) = S(k)\ 1 \cup S(k)2$.

B. The horizontal setting

The only interaction between the players in the horizontal setting is for computing the size and closure of clusters (as described in Section 4), and computing the distribution of the sensitive values in each cluster (Section 6.1). During the protocol, the players may learn information on records held by other players, which is not implied by their own input and the final output. Therefore, the protocol is not perfectly secure in the cryptographic sense. Such a compromise is widely acceptable since, as written in, "allowing innocuous information leakage allows an algorithm that is sufficiently secure with much lower cost than a fully secure approach". Indeed, many distributed protocols accept innocuous information leakage for gaining efficiency, utility and practicality. We proceed to characterize herein the excessive information that is leaked, compare it to information leakage in other protocols, and argue that such a leakage of information is benign from practical point of view. We separate our discussion to three types of information that the players may learn on the private data of other players. Assume that the different players are hospitals and the partial database of Hospital i, $1 \leq i \leq m$, holds information on the patients in that hospital. One of the participating hospitals may be interested to know whether a particular individual, Alice, was hospitalized in one of the other hospitals. Using Alice's publicly accessible quasi-identifier values, which hospital may try to examine his view of the protocol in order to deduce the answer? More generally, the hospital may wish to learn how many people from a given age range and location took part in the other databases. In Section 8.2.1 we explain why such inferences are hard and sometimes even impossible to extract from the protocol's views. Alternatively, it is possible that one hospital knows that Alice was hospitalized in another participating hospital, but it wishes to know her sensitive value. In Section 8.2.2 we explain why it is impossible to extract such information beyond what is implied by the final k-anonymized and ℓ-diversified anonymization. Finally, it is possible that hospitals will aim at learning information on the number of patients in the other hospitals. we explain how to hide also that information. (To the best of our knowledge, no other study dealt with the question of hiding the size of the partial databases.) Information on the quasi-identifiers of records of other players discuss possible inferences that the players may make on the quasi-identifier values of records of other players. In the first part of this section we show that any attempt to infer information about the inclusion of a given quasi-identifier record, $R = (R(1), \ldots, R(d))$, in the unified database D is useless. Then, we proceed to characterize the significantly weaker type of information leakage on the quasi-identifier values of records in D that does occur. We conclude this section by comparing the information leakage on quasi-identifiers in our protocol to other protocols in the horizontal setting

Example 1. Some specific record R. Hence, those records are connected by an hyperedge if they could all be the generalized view of the same original record in D. Example 1. Consider the table D in Table 2 that has $d = 3$ quasi-identifier attributes, $A1 = \{a, b\}$, $A2 = \{x, y\}$ and $A3 = \{1, 2\}$. Assume that during the distributed protocol, the players constructed $p = 3$ anonymized views of D as shown in Table 2. The corresponding hypergraph GD is shown in Figure 1. It has four yperedges: The three hyperedges that correspond to the three real records in D, and a fourth artifact hyperedge. The first hyperedge is $\{R1, R21, R31\}$, since all those generalized records generalize the record $R1$

D	$\overline{D}_1$	$\overline{D}_2$	$\overline{D}_3$
$R_1 = (a, x, 1)$	$\overline{R}_1^1 = (a, x, *)$	$\overline{R}_1^2 = (*, x, 1)$	$\overline{R}_1^3 = (a, *, 1)$
$R_2 = (b, x, 2)$	$\overline{R}_2^1 = (b, x, *)$	$\overline{R}_2^2 = (*, x, 2)$	$\overline{R}_2^3 = (b, *, 2)$
$R_3 = (a, y, 2)$	$\overline{R}_3^1 = (a, y, *)$	$\overline{R}_3^2 = (*, y, 2)$	$\overline{R}_3^3 = (a, *, 2)$

Table 2. A table D and three anonymized views

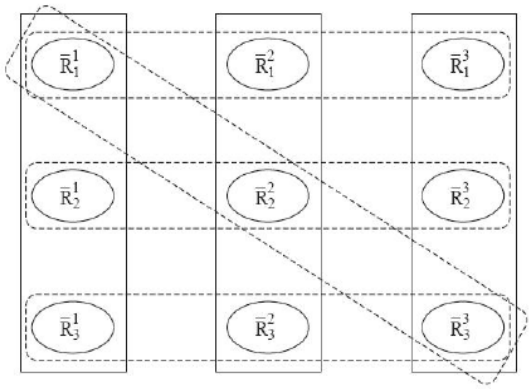

Figure 1. The hypergraph corresponding to the three anonymized views in Table 2

Conclusions

In this paper, we presented privacy-preserving protocols that enable two honest but curious parties to compute APSD and SSSD on their *joint graph*. A related problem is how to construct privacy-preserving protocols for graph *comparison*. Many of these problems (*e.g.*, comparison of the graphs' respective maximum flow values) reduce to the problem of privacy-preserving comparison of two values, and thus have reasonably efficient generic solutions. For other problems, such as graph isomorphism, there are no known polynomial-time algorithms even if privacy is not a concern. Investigation of other interesting graph algorithms that can be computed in a privacy-preserving manner is a topic of future research. In conclusion, we presented a general approach to secure distributed computations of anonymized views of shared databases.

The presented algorithms are highly efficient and simple, as they rely on very basic and few cryptographic primitives. Even though we focused here on distributed versions of one particular algorithm (sequential clustering) and one particular goal (anonymization), the ideas and techniques that were presented here are suitable for any other algorithm that reorganizes clusters (like simulated annealing or *k*-means) and could be applicable also for other distributed data mining problems. $\in$ *D*. The sets *{R*1 2,*R*2 2,*R*3 2 *}* and *{R*13,*R*23,*R*33*}* are two additional hyperedges, corresponding to *R*2,*R*3 $\in$ *D*. The fourth hyperedge is *{R*11,*R*22,*R*33*}*. All three records in that hyperedge indeed generalize the same record — (*a*, *x*, 2). However, as opposed to the first three hyperedges (which generalize a true record in *D*), that latter record is an artifact one that does not appear in *D*.

For each algorithm considered in this paper, we calculate the number of rounds, the total communication complexity, and the computational complexity, and compare them with the generic method. Using Yao's method on a circuit with *m* gates and *n* inputs requires *O*(1) rounds, *O*(*m*) communication, and *O*(*m*+*n*) computational overhead. Lindell and Pinkas note in [28] that the computational overhead of the *n* oblivious transfers in each invocation of Yao's protocol typically dominates the computational overhead for the *m* gates, but for correct asymptotic analysis we must still consider the gates. *Complexity of privacy-preserving APSD.* For our analysis we will assume that the edge set *E* has size *n*, and that the maximum edge length is *l*. The generic approach to this problem would be to apply Yao's Method to a circuit that takes as input the length of every edge in *G*1 and *G*2, and returns as output *G* = APSD(gmin(*G*1,*G*2)). Clearly, such a circuit will have 2*n* log *l* input bits. To count the number of gates, note that a circuit to implement Floyd-Warshall requires *O*(*n*3/2) minimums and *O*(*n*3/2) additions. For

integers represented with log l bits, both of these functionalities require log l gates, so we conclude that Floyd-Warhsall requires $O(n3/2 \log l)$ gates. To compute gmin requires $O(n \log l)$ gates, but this term is dominated by the gate requirement for Floyd-Warshall. We conclude that the generic approach requires $O(1)$ rounds, $O(n3/2 \log l)$ communication, and $O(n3/2 \log l)$ computational overhead. The complexity of our approach depends on the number of protocol iterations k, which is equal to the number of different edge lengths that appear in the solution graph. In iteration i, we take the minimum of two (lg l)-bit integers, and compute a set union of size si. Because each edge in the graph appears in exactly one of the set unions, we also know that $\sum_{i=1}^{k} si = n$. First we will determine the contribution to the total complexity made by the integer minimum calculations. If we use Yao's protocol, then each integer minimum requires a constant number of communication rounds, $O(\lg l)$ inputs, and $O(\lg l)$ gates, so the k calculations together contribute $O(k)$ rounds, $O(k \lg l)$ communication complexity, and $O(k \lg l)$ computational complexity. Complexity contribution of the set union subprotocols depends on whether we use the iterative method or the tree pruning method as described in section 4. If the iterative method is used, then the k invocations of set union require a total of $O(n)$ rounds, $O(k \lg n)$ communication complexity, and $O(k \lg n)$ computational complexity. If the tree-pruning method is used, then $O(k \lg n)$ rounds are required, but the communication and computational complexity remains the same. The asymptotically better performance of the iterative method hides the fact that each of the k rounds requires $O(\lg n)$ oblivious transfers, which are considerably more expensive than the $O(|si|)$ private Bit-Or computations performed in each of the lg u rounds of the tree-pruning method.

Using the iterative method for set union, and noting that $k = O(n)$, we conclude that our APSD protocol requires $O(n)$ communication rounds, $O(n \log n + n \log l)$ communication complexity, and $O(n \log n + n \log l)$ computational complexity. As compared to the generic approach, we have traded more rounds for better overall complexity. *Complexity of privacy-preserving SSSD.* Complexity of SSSD is similar to that of APSD, except that the number of rounds is $k = O(v)$ and the total number of set union operations is v, where v is the number of vertices ($O(e1/2)$). We conclude that our protocol requires $O(v)$ rounds, $O(v(\log v + \log l))$ oblivious transfers, and $O(v(\log v + \log e))$ gates. A generic solution, on the other hand, would require $O(v2 \log l)$ oblivious transfers.

FDM generates fewer candidates than CD, and use effective pruning techniques to minimize the messages for the support exchange step. In each site, FDM finds the local support counts and prunes all infrequent local support counts[7]. After completing local pruning, instead of broadcasting the local counts of all candidates as in CD, they send the local counts to polling site. FDM's main advantage over CD is that it reduces the communication overhead to O (|Cp|*n), where |Cp| and n are potentially frequent candidate item sets and the number of sites, respectively[8]. When different sites have nonhomogeneous data sets, the number of disjoint candidate itemsets among them is frequent, and FDM generates fewer candidate itemsets compared to CD. Mining Association Rules Efficient algorithms for mining frequent itemsets are crucial for mining association rules as well as for many other data mining tasks. Methods for mining frequent itemsets have been implemented using a prefix-tree structure, known as an FP-tree, for storing compressed information about frequent itemsets. Numerous experimental results have demonstrated that these algorithms perform extremely well. In this paper, we present a novel FP-array technique that greatly reduces the need to traverse FP-trees, thus obtaining significantly improved performance for FP-tree-based algorithms. Our technique works especially well for sparse data sets. Furthermore, we present new algorithms for mining all, maximal, and closed frequent itemsets. The results show that our methods are the fastest for many cases. Even though the algorithms consume much memory when the data sets are sparse, they are still the fastest ones when the minimum support is low

V. CONCLUSION

Previous work in privacy preserving data mining has considered two related settings. One, in which the data owner and the data miner are two different entities, and another, in which the data is distributed among several parties who aim to jointly perform data mining on the unified corpus of data that they hold. In the first setting, the goal is to protect the data records from the data miner. Hence, the data owner aims at anonym zing the data prior to its release. The main approach in this context is to apply data perturbation. The idea is that Computation and communication costs versus the number of transactions N the perturbed data can be used to infer general trends in the data, without revealing original record information.

In the second setting, the goal is to perform data mining while protecting the data records of each of the data owners from the other data owners. This is a problem of secure multiparty computation. The usual approach here is cryptographic rather than probabilistic. Lindell and Pinkas showed how to securely build an ID3 decision tree when the training set is distributed horizontally. Lin et al. discussed secure clustering using the EM algorithm over horizontally distributed data. The problem of distributed association rule mining was studied in in the vertical setting, where each party holds a different set of attributes, and in in the horizontal setting. Also the work of considered this problem in the horizontal setting, but they considered large-scale systems We proposed a

protocol for secure mining of association rules in horizontally distributed databases that improves significantly upon the current leading protocol in terms of privacy and efficiency. One of the main ingredients in our proposed protocol is a novel secure multi-party protocol for computing the union (or intersection) of private subsets that each of the interacting players hold. Another ingredient is a protocol that tests the inclusion of an element held by one player in a subset held by another. Those protocols exploit the fact that the underlying problem is of interest only when the number of players is greater than two. One research problem that this study suggests was described in Section 3; namely, to devise an efficient protocol for inequality verifications that uses the existence of a semihonest third party. Such a protocol might enable to further improve upon the communication and computational costs of the second and third stages of the protocol of , as described in Sections 3 and 4. Other research problems that this study suggests is the implementation of the techniques presented here to the problem of distributed association rule mining in the vertical setting , the problem of mining generalized association rules , and the problem of subgroup discovery in horizontally

ACKNOWLEDGEMENT

We express our sincere thanks to my guide N.KOWSALYA for their whole hearted and kind cooperation. We extend our thanks to all the faculties of the department of Computer Science and Applications, who were behind throughout the course of study.

REFERENCES

[1]T. Tassa and E. Gudes. Secure distributed computation of anonymized Views of shared databases. *Transactions on Database Systems*, 37,Article 11, 2012.

[2]J. Brickell and V. Shmatikov. Privacy-preserving graph algorithms in the semi-honest model. In *ASIACRYPT*, pages 236–252, 2005.

[3] D.W.L Cheung, V.T.Y. Ng, A.W.C. Fu, and Y. Fu. Efficient mining of association rules in distributed databases. IEEE Trans. Knowl. Data Eng., 8(6):911–922, 1996.

[4] G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the *k*th-ranked element. In *EUROCRYPT*, 2004.

[5] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *ACM-SIGMOD Conference on Management of Data*, pages 439–450, May 2000.

[6] R. Bayardo and R. Agrawal. Data privacy through optimal *k*-anonymization. In *ICDE*, 2005.

[7] D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *STOC*, pages 503–513, 1990.

[8] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee. Towards privacy in public databases. In *Proc. 2nd Theory of Cryptography Conference (TCC)*, volume 3378 of *LNCS*, pages 363–385. Springer-Verlag, 2005.

[9] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval.*J. ACM*, 45(6):965–981, 1998.

[10] D.W. Cheung, et al., "A Fast Distributed Algorithm for Mining Association Rules," Proc. Parallel and CS Press, 1996,pp. 31-42;

[11] M.J. Zaki and Y. Pin, "Introduction: Recent Developments in Parallel and Distributed Data Mining," J. Distributed and Parallel Databases, vol. 11, no. 2, 2002, pp. 123-127.

