社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2008－CSEC－42（44）
2008 / 7 / 25

頻出パータン木を利用した安全な相関ルール発見手法

蘇 春華† 櫻井 幸一††

† 九州大学大学院システム情報科学府
†† 九州大学大学院システム情報科学研究院
E-mail: †su@itslab.csce.kyushu-u.ac.jp, ††sakurai@csce.kyushu-u.ac.jp

A Secure Association Rules Mining Scheme Based on Frequent-Pattern Tree

Chunhua SU† and Kouichi SAKURAI††

† Department of Computer Science and Communication Engineering, Kyushu University
†† Department of Computer Science and Communication Engineering, Kyushu University
E-mail: †su@itslab.csce.kyushu-u.ac.jp, ††sakurai@csce.kyushu-u.ac.jp

Abstract Many government organizations and companies want to share their documents in a similar theme to get the joint benefits. Textual document clustering is a powerful data mining technique to analyze the large amount of documents and structure large sets of text or hypertext documents. While doing the document clustering in the distributed environment, it may involve the users' privacy of their own document. In this paper, we propose a framework to do the privacy-preserving text mining among the users under the distributed environment: multiple parties, each having their private documents, want to collaboratively execute agglomerative document clustering without disclosing their private contents to any other parties.

Key words association rules, privacy-preserving, cryptographic protocol

1. Introduction

Association rules mining techniques are generally applied to databases of transactions where each transaction consists of a set of items. In such a framework the problem is to discover all associations and correlations among data items where the presence of one set of items in a transaction implies (with a certain degree of confidence) the presence of other items. Association rules are statements of the form $X_1, X_2, ..., X_n \Rightarrow Y$, meaning that if we find all of $X_1, X_2, ..., X_n$ in the transactions, then we have a good chance of finding Y. The probability of finding Y for us to accept this rule is called the confidence of the rule. We normally would search only for rules that had confidence above a certain threshold. The problem is usually decomposed into two sub-problems. One is to find those itemsets whose occurrences exceed a predefined threshold in the database; those itemsets are called frequent or large itemsets. The second problem is to generate association rules from those large itemsets with the constraints of minimal confidence. Much data mining starts with the assumption that we only care about sets of items with high support, they appear together in many transactions. We then find association rules only involving a high-support set of items. That is to say that $X_1, X_2, ..., X_n \Rightarrow Y$ must appear in at least a certain percent of the transactions, called the support threshold. How to do the global support threshold counting with respecting clients' privacy is a major problem in privacy-preserving rules mining.

$support_{X \Rightarrow Y} = \frac{|T_{X \cup Y}|}{|DB|}$ means that the support is equal to the percentage of all transactions which contain both X and Y in the whole dataset. And then we can get that: $confident_{X \Rightarrow Y} = \frac{support_{X \Rightarrow Y}}{support_X}$

The problem of mining association rules is to find all rules whose support and confidence are higher than certain user specified minimum support and confidence.

Distributed mining can be applied to many applications which have their data sources located at different places. In this paper, we assume that there are n parties possess their private databases respectively. They want to get the common benefit for doing association rules analysis in the joint databases. For the privacy concerns, they need a private preserving system to execute the joint association rules mining. The concern is solely that values associated with an individual entity not being revealed.

1.1 Motivation and Our Contributions

In the recent research papers [5] [13] [15], some privacy-preserving association rules schemes are proposed. These papers are similar and developed a secure multi-party protocol based on homomorphic encryption. The related works we mentioned above have three problems. The first one is low efficiency by using the Apriori algorithm. As we know, the Apriori algorithm is not so efficient because of its candidates generation scan. The second one is the accuracy problem in [3] in which there is a trade-off between the accuracy and security. The third one is the security problem, the schemes proposed in the related works are not collusion-resistant. We propose a improved scheme to overcome these three problems.

- We apply frequent-pattern tree (FP-tree) structure to execute the association rules mining and extend it to distributed association rules mining framework. we can use FP-tree to compress a large database into a compact FP-tree structure to avoid costly database scans.
- We present a privacy-preserving protocol which can overcoming the accuracy problem causes randomization-based techniques and improve the efficiency compared to those cryptography-based scheme [5] [13] [15].
- Our privacy-preserving protocol provide a perfect security and collusion-resistant property. Our construction is based on the attribute

2. Preliminaries

2.1 Problem Definition

We assume that there are n parties want to do cooperation on the joint databases $D_1 \cup D_2 \cup ... \cup D_n$ without revealing the private information of database. And we assume the standard synchronous model of computation in which n parties communicate by sending messages via point-to-point channels. There are some distributed parties who want to get the global result from their data transactions over the internet. Every party P_i has their private transaction T_i^1. They all have serious concern about their privacy while they want to get the accurate result to help their following decision. No Party should be able to learn contents of a transaction of any other client. And we want to use some cryptographic toolkits to construct a secure multi-party computation protocol to perform this task. Let $I = \{a_1, a_2, ..., a_m\}$ be a set of items, and a transaction database $DB = \langle T_1, T_2, ..., T_n \rangle$, where $T_i (i \in [1...n])$ is a transaction which contains a set of items in I. The support (or occurrence frequency) of a pattern A, where A is a set of items, is the number of transactions containing A in DB. A pattern A is frequent if A's support is no less than a predefined minimum support threshold $MinSupp$.

2.2 Cryptographic Primitives

Public Key Encryption with Homomorphic Property: In modern terms, a public-key encryption scheme on a message space M consists of three algorithms (K, E, D):

(1) The key generation algorithm $K(1^k)$ outputs a random pair of private/public keys (sk, pk), relatively to a security parameter k.

(2) The encryption algorithm $E_{pk}(m; r)$ outputs a ciphertext c corresponding to the plaintext $m \in M$, using random value r.

(3) The decryption algorithm $D_{sk}(c)$ outputs the plaintext m associated to the ciphertext c. We will occasionally omit the random coins and write $E_{pk}(m)$ in place of $E_{pk}(m; r)$. Note that the decryption algorithm is deterministic.

In this paper we use Palliler encrytion as public key encryption. Paillier homomorphic encryption proposed by Pallier [11]. It is provably secure and one-way based on the Decisional Composite Residuosity Assumption and the Computational Composite Residuosity Assumption.

Attributes-based Encryption (ABE) The ABE scheme is developed from Identity based encryption(IBE) which introduced by Shamir [12], is a variant of encryption which allows users to use any string as their public key (for example, an email address). This means that the sender can send messages knowing only the recipient's identity (or email address), thus eliminating the need for a separate infrastructure to distribute public keys. In their scheme, there is one authority giving out secret keys for all of the attributes. Each encryptor then specifies a list of attributes such that any user with at least d of those attributes will be able to decrypt. They show that the scheme they present is secure.

2.3 Security Definition and Adversary Model

This paper considers both semi-honest and malicious adversaries. For Semi-honest adversaries, every party are assumed to act according to their prescribed actions in the protocol. The security definition is straightforward, particularly as in our case where only one party learns an output.

3. Secure Multi-party Protocol for Association Rules Mining based on FP-tree

3.1 Problem in Apriori-based Distributed Association Rules Mining

Most distributed association rules mining algorithms are adaptations of existing sequential (serial) algorithms. Generally speaking two strategies for distributing data for parallel computation can be identified:

(1) Data distribution: The data is apportioned amongst the processes, typically by "horizontally" segmenting the dataset into sets of records. Each process then mines its allocated segment (exchanging information on-route as necessary).

(2) Task distribution: Each process has access to the entire dataset but is responsible for some subset of the set of candidate itemsets.

The Apriori heuristic achieves good performance gained by (possibly significantly) reducing the size of candidate sets. However, in situations with a large number of frequent patterns, long patterns, or quite low minimum support thresholds, an Apriori-like algorithm may suffer from the following two nontrivial costs:

- It is costly to handle a huge number of candidate sets. For example, if there are 10^4 frequent 1-itemsets, the Apriori algorithm will need to generate more than 10^7 length-2 candidates and accumulate and test their occurrence frequencies. Moreover, to discover a frequent pattern of size 100, such as $\{a_1, ..., a_{100}\}$, it must generate $2^{100} - 2 \approx 10^{30}$ candidates in total. This is the inherent cost of candidate generation, no matter what implementation technique is applied.
- It is tedious to repeatedly scan the database and check a large set of candidates by pattern matching, which is especially true for mining long patterns.

3.2 The General Description of Our Proposal

our protocol construction is based on secure multi-party computation techniques. The history of the multi-party computation problem is extensive since it was introduced by Yao [14] and extended by Goldreich, Micali, and Wigderson [7]. Secure multi-party computation (MPC) protocols allow a set of n players to securely compute any agreed function on their private inputs, where the following properties must be satisfied: *privacy*, meaning that the corrupted players do not learn any information about the other players' inputs. and *correctness*, meaning that the protocol outputs the correct function value, even when the malicious players treat. In Secure Multi-party Computation, we always assume that *semi-honest model* exists.

(1) Every party executes FP-tree construction and prepare for the global FP-tree construction using attribute-based encryption scheme.

(2) Merge the conditional FP-trees to get using the private matching scheme.

(3) Support count among the common k-item sets privately using the scalar homomorphic encryption scheme.

(4) Secure global support count computation.

(5) Output the final result of association rules

Initiation: We assume that three secure (and sufficiently efficient) sub-protocols are available:

(1) Proving you know a plaintext: If P_i has created an encryption $E(a)$, he can give a zero-knowledge proof of knowledge that he knows a (or more accurately, that he knows a and a witness to he fact that the plaintext is a).

(2) Proving multiplications correct: Assume P_i is given an encryption $E(a)$, chooses a constant α, computes a random encryption $E(\alpha a)$ and broadcasts $E(\alpha)$, $E(\alpha a)$. He can then give a zero-knowledge proof that indeed $E(\alpha a)$ contains the product of the values contained in $E(\alpha)$ and $E(a)$.

(3) Threshold decryption: For the third sub-protocol, we have common input pk and an encryption $E(a)$, in addition every player also uses his share of the private key as input. The protocol computes securely a as output for everyone.

3.3 Distributed Association Mining With FP-tree

FP-growth is a divide-and-conquer methodology proposed by [8] which decomposes the association rules mining tasks into smaller ones. It only scans the database twice and does not generate candidate itemsets. The algorithm substantially reduces the search costs. At first, we let the parties build a global FP-tree together and then do the association rules mining on the global FP-tree. FP-growth, for mining the complete set of frequent patterns by pattern fragment growth. Efficiency of mining is achieved with three techniques: (1) a large database is compressed into a condensed, smaller data structure, FP-tree which avoids costly, repeated database scans, (2) our FP-tree-based mining adopts a pattern-fragment growth method to avoid the costly generation of a large number of candidate sets, and (3) a partitioning-based, divide-and-conquer method is used to decompose the mining task into a set of smaller tasks for mining confined patterns in conditional databases, which dramatically reduces the search space.

A frequent-pattern tree (or FP-tree in short) is a tree structure defined below:

(1) It consists of one root labeled as "null", a set of item-prefix sub-trees as the children of the root, and a frequent-item-header table.

(2) Each node in the item-prefix sub-tree consists of three fields: item-name, count, and node-link, where item-name registers which item this node represents, count regis-

ters the number of transactions represented by the portion of the path reaching this node, and node-link links to the next node in the FP-tree carrying the same item-name, or null if there is none.

(3) Each entry in the frequent-item-header table consists of two fields, (1) item-name and (2) head of node-link (a pointer pointing to the first node in the FP-tree carrying the item-name). Based on this definition, we have the following FP-tree construction algorithm.

4. The Details of Multi-Party Mining Scheme

In our scheme, we assume that the universe of attributes can be partitioned into K disjoint sets. Each will be monitored by a different authority. As mentioned above, we also have one trusted central authority who does not monitor any attributes.

4.1 Verifiable Secret Sharing

Secret-sharing schemes are used to divide a secret among a number of parties. The information given to a party is called the share (of the secret) for that party. It realizes some access structure that defines the sets of parties who should be able to reconstruct the secret by using their shares First, let's consider a very simplified scheme based on the Feldman Verifiable Secret Sharing scheme. Recall that, given d points $p(1), ..., p(d)$ on a $d-1$ degree polynomial, we can use Lagrange interpolation to compute $p(i)$ for any i. However, given only $d-1$ points, any other points are information theoretically hidden. According to the Lagrange formula, $p(i)$ can be computed as a linear combination of d known points. Let $\Delta_j(i)$ be the coefficient of $p(j)$ in the computation of $p(i)$. Then $p(i) = \sum_{j \in S} p(j)\Delta_j(i)$ where S is a set of any d known points and $\Delta_j(i) = \prod_{k \in S, j \neq k}(i-k)/(j-k)$. Note that any set of d random numbers defines a valid polynomial, and given these numbers we can find any other point on that polynomial.

4.2 Specifying Transaction's Attributes

If we take this approach, any user with any d attributes which are specified will be able to decrypt. But we want each encryptor to be able to give a specific subset of attributes such that at least d are necessary for decryption. In order to do this, we need an extra tool: bilinear maps, for bilinear map $e, g \in G_1$, and $a, b \in Zq$, $e(g^a, g^b) = e(g, g)^{ab}$. Now, suppose instead of giving each user $g^{p(i)}$ for each attribute i, we choose a random value t_i and give $g^{p(i)/t_i}$. If the user knew g^{t_i} for at least d of these attributes, he could compute $e(g, g)^{p(i)}$ for each i and then interpolate to find the secret $e(g, g)^{p(0)}$. Then if our encryption includes $e(g, g)^{p(0)}m$, the user would be able to find m. Thus, the encryptor can specify which attributes are relevant by providing g^{t_i} for each attribute i in the desired transaction set.

4.3 Multiple Encryptions

First, let's consider a very simplified scheme based on the Feldman Verifiable Secret Sharing scheme. Recall that, given d points $p(1), ..., p(d)$ on a $d-1$ degree polynomial, we can use Lagrange interpolation to compute $p(i)$ for any i. However, given only $d-1$ points, any other points are information theoretically hidden. According to the Lagrange formula, $p(i)$ can be computed as a linear combination of d known points. Let $\Delta_j(i)$ be the coefficient of $p(j)$ in the computation of $p(i)$. Then $p(i) = \sum_{j \in S} p(j)\Delta_j(i)$ where S is a set of any d known points and $\Delta_j(i) = \prod_{k \in S, j \neq k}(i-k)/(j-k)$. Note that any set of d random numbers defines a valid polynomial, and given these numbers we can find any other point on that polynomial.

Thus our first attempt Multi-Authority Scheme is as follows:

Preparing for the Global FP-tree Construction

Init First fix $y_1 ... y_k$, $\{t_{k,i}\}_{i=1...n, k=1...K} \leftarrow Z_q$. Let $y_0 = \sum_{k=1}^{K} y_k$. **System Public Key** $Y_0 = e(g, g)_{y_0}$.

Attribute Authority k

Authority Secret Key The SW secret key: $y_k, t_{k,1} ... t_{k,n}$.
Authority Public Key $T_{k,i}$ from the SW public key: $T_{k,1} ... T_{k,n}$ where $T_{k,i} = g^{t_{k,i}}$.
Secret Key for User u from authority k Choose random $d-1$ degree polynomial p with $p(0) = y_k$. Secret Key: $\left\{D_{k,i} = g^{p(i)/t_{k,i}}\right\}_{i \in A_u}$.

Encryption for attribute set A_C

Choose random $s \leftarrow Z_q$.
Encryption: $E = Y_0^s m, \left\{E_{k,i=T_{k,i}^s}\right\}_{i \in A_C^k, \forall k}$
Decryption: For each authority k, for d attributes $i \in A_C^k \cap A_u$, compute $e(E_{k,i}, D_{k,i}) = e(g, g)^{p(i)s}$. Interpolate to find $Y_k^s = e(g, g)^{p(0)s} = e(g, g)y_k^s$. Combine these values to obtain $\prod_{k=1}^{K} Y_k^s = Y_0^s$. Then $m = E/Y_0^s$.

4.4 Private FP-tree Matching

Here, we apply the secure matching protocol proposed by Freedman et al. citeFNP04 to merge the FP-tree between every two parties. We propose a framework whereby all parties participate to a secure aggregation mechanism without having access to the protected data.

Private Merging Protocol

Input: Party A's input is a set $T_A = \{T_A^1, ..., T_A^k\}$, party B's input is a set $T_B = \{T_B^1, ..., T_B^l\}$. The elements in the input sets are taken from a domain of size N.

(1) Party performs the following:

(a) He chooses the secret-key parameters for a semantically-secure homomorphic encryption scheme, and publishes its public keys and parameters. The plaintexts are in a field that contains representations of the N elements of the input domain, but is exponentially larger.

(b) She uses interpolation to compute the coefficients of the polynomial $P(y) = \sum_{i=0}^{k} \alpha_i T_B^i$ of degree k with roots $x_1, ..., x_k$.

(c) She encrypts each of the $(k+1)$ coefficients by the semantically-secure homomorphic encryption scheme and sends to party B the resulting set of ciphertexts, $Enc(\alpha_0), ..., Enc(\alpha_k)$.

(2) Party B performs the following for every $T_B^i \in T_B$:

(a) He uses the homomorphic properties to evaluate the encrypted polynomial at T_B^i. That is, he computes $Enc(P(y)) = Enc(\sum_{i=0}^{k} \alpha_i T_B^i)$.

(b) He chooses a random value r and computes $Enc(rP(y) + y)$. (One can also encrypt some additional payload data py by computing $Enc(rP(y) + (T_B^i | p_B))$. Party obtains p_B iff T_B^i is in the intersection.) He randomly permutes this set of kS ciphertexts and sends the result back to the client .

(3) Party A decrypts all l ciphertexts received. She locally outputs all values $x \in X$ for which there is a corresponding decrypted value .

In order to ensure end to end confidentiality, the framework uses additive homomorphic encryption algorithms. All the count of the conditional FP-tree is merged in this step.

5. Conclusions

The main contribution of this paper is proposing a general framework for privacy preserving association rules mining. For that the randomization methodologies are not good enough to attain the high accuracy and protect clients' information from privacy breach and the malicious attack, we show that how association rules mining can be done in this framework and prove that is secure enough to keep the clients' privacy. We also show that our protocols works with less communication complexity and communication complexity compared to other related schemes. In the future research, a common framework with more formal and reliable for privacy preservation will enable next generation data mining technology to make substantial advances in alleviating privacy concerns.

文　献

[1] R. Agrawal, T. Imielinski, and A. N. Swami. *Mining association rules between sets of items in large databases.* Proceedings of the ACM SIGMOD International Conference on Management of Data, 1993

[2] P. Feldman. *A practical scheme for non-interactive verifiable secret sharing.* In Proc. of FOCS pp.427-437 1987.

[3] T. Fukazawa, J. Wang, T. Takata, and M. Miyazak *An Effective Distributed Privacy-Preserving Data Mining Algorithm.* Fifth International Conference on Intelligent Data Engineering and Automated Learning, UK, 2004.

[4] Oded Goldreich. *Foundations of Cryptography Volume 2, Chapt.7,* Cambridge Univ. Press, 2004.

[5] O. Goldreich, S. Micali and A. Wigderson. emphHow to play any mental game. In Proceedings of the 19th annual ACM symposium on Theory of computing, 1987

[6] J. Han, J. Pei, Y. Yin, R. Mao, *"Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach"*, Data Mining and Knowledge Discovery, pp.53-87 (2004).

[7] Y. Lindell and B. Pinkas. *Privacy preserving data mining.* In Advances in Cryptology CRYPTO '00, volume 1880 of Lecture Notes in Computer Science, pp. 36-54. Springer-Verlag, 2000.

[8] Pascal Paillier. *Public-key cryptosystems based on composite degree residuosity classes.* In EUROCRYPT, Prague, Czech Republic,1999.

[9] Adi Shamir. Identity-based cryptosystems and signature schemes. In Proc. of CRYPTO 1984, volume 196, Springer LNCS, pp. 47-53, 1984.

[10] J.S. Vaidya and C. Clifton. *"Privacy Preserving Association Rule Mining in Vertically Partitioned Data"*. Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2002.

[11] A.C Yao, *Protocols for Secure Computation,* In 23rd FOCS, 1982

[12] J. Z. Zhan, S. Matwin, L. Chang: Privacy-Preserving Collaborative Association Rule Mining. Procceding of DBSec 2005, pp.153-165, 2005

[13] M. Freedman, K. Nissim and B. Pinkas. *Efficient Private Matching and Set Intersection,* Eurocrypt'04 Proceedings, LNCS 3027, Springer-Verlag, pp. 1-19, May 2004.