Protocol Design for Privacy-Preserving Data Mining Using Partial Homomorphic Encryption

A THESIS Presented to

The Academic Faculty By

Yin-Ming Chang

In Partial Fulfillment

of the Requirements for the Degree of

Master in Computer Science

Institute of Computer Science

College of Computer Science

National Chiao Tung University

2013

Abstract

With the advance of computing power, data mining techniques can extract useful information from large amount of data. In 2012, 2.5 quintillion bytes of data (1 follow 18 zeroes) are created every day. Data privacy is of utmost concern for distributed data mining across multiple parties, which may be competitors. In this thesis, we focus on the privacy preserving techniques in distributed data mining algorithms. We propose two protocols — multi-party association rule mining (MP-ARM) and multi-party decision tree learning (MP-DTL). Both protocols use partial homomorphic encryption to perform secure data mining algorithms, which are more efficient than the existing work. With the aid from the third participant, two or more parties can securely perform large-scale data mining algorithms without revealing any additional information to the cloud servers.

Acknowledgements

I would like to appreciate the support from my parents. Also, I especially thank the suggestions from Professor Li-Chun Wang who discussed with me every week during these two years. I could not finish this work without his guidance and comments. I would also like to thank my laboratory mates in the cloud computing group. They discussed and excised with me. I can hardly go through these days without the company of them. Last but not the least, I give thanks to those who supported me during the completion of this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data mining is the process of knowledge discovery as shown in Fig. 1.1. The overall goal of data mining is to extract patterns and useful trends in large data sets, which involves the knowledge from different fields, including artificial intelligence, machine learning, statistics, and database systems. Data mining techniques [1] have been developed to make decisions precisely. For example, one Midwest chain used data mining technique to analyze local buying patterns [2]. They found that people tended to buy beer when they bought diapers on Saturday. On Thursdays, however, they only bought a few items. The retailer concluded that they needed to prepare more beers before the weekends. The grocery chain can use this newly discovered information in various ways to increase their revenue. For example, they can move the beer display closer to the diaper display, or sell beers and diapers at full price on Thursdays.

There are various types of existing data mining algorithms, these can mainly classified as: 1) classification; 2) clustering; 3) association.

Classification. Data mining classification is the process to group items based on certain key characteristics. There are several techniques used for data mining classification, including nearest neighbor classification [3], decision tree learning [4], and support vector machine [5].

Clustering. Data mining clustering is to find the groups of objects which are similar to each other, and differentiate with other groups.

Association. In data mining, association rule learning is a popular method to discover interesting relations between variables in large databases. It is intended to identify the strong rules discovered in databases using different measures of interests. For example, the rule $\{diaper\} \rightarrow \{beer\}$ means that while diaper appears in a transaction record, beer would also appears in this record with high probabilities.

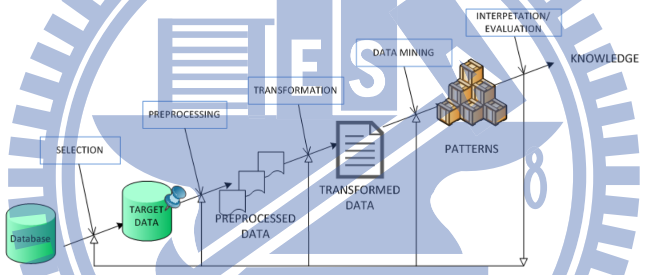

Figure 1.1: The process of knowledge discovery.

To perform a data mining algorithm, the traditional approach first collects all data into a centralized site, then run mining algorithms. However, a centralized database has the difficulty in sharing data among competitors due to the privacy concern. Data may be distributed among several sites, which are not allowed to transfer data directly to another site owing to the untrust network environment.

Taking medical research cooperation as an example. Suppose that Center for Disease Control wants to mine health records to find out the relationship between genders, ages, and health. Insurance companies and local hospitals hold data of patient diseases and prescriptions. Mining these data can help the Center of Disease Control to discover important rules such as *gender* & *age* $\Rightarrow$ *health status*. However, the problem is that insurance companies or hospitals will not share their data publicly. It is illegal to share patients' records without their permissions. Therefore, the protocol designed in this situation should ensure that all the sensitive information can not be known by outsiders.

In order to achieve the above scenario, data transfer between the Center of Disease Control and insurance companies or hospitals must be encrypted before sending to another site. This encryption behavior limits the capability of calculation. We propose the effective protocols for association rule mining and decision tree learning to reduce the time spent on collecting the intermediate values before obtaining the mining results.

In this thesis, we assumes that two or more participants want to perform data mining algorithms based on their joint databases. In the two-party case, we introduce a new participant Ted to help decide whether outcome is larger than the predetermined threshold or not. A special property should be addressed to emphasize the difficulty to preserve privacy when performing data mining algorithm under two-party case. For association rule mining, one party will know a rule is supported globally, but not supported in his site. This behavior shows that the other site supports this rule. Thus, lots of private information is revealed even under a secure protocol. Also, we assume no collusion in proposed protocols because collision may fail the security assurance.

We consider a scenario where two or more parties having private databases plan to compute a data mining algorithm based on the union of their databases. Due to confidential data, neither party will divulge any contents to the other.

We show how the involved data mining problem of decision tree learning can be efficiently computed, with no party learning anything other than the output itself. We demonstrate apriori algorithm and ID3 algorithm, both of which are well-known and influential algorithm for the task of data mining. We note that the extensions of apriori and ID3 are widely used in real market applications. Our proposed protocols are designed to securely perform two data mining algorithms as Fig. 1.2.

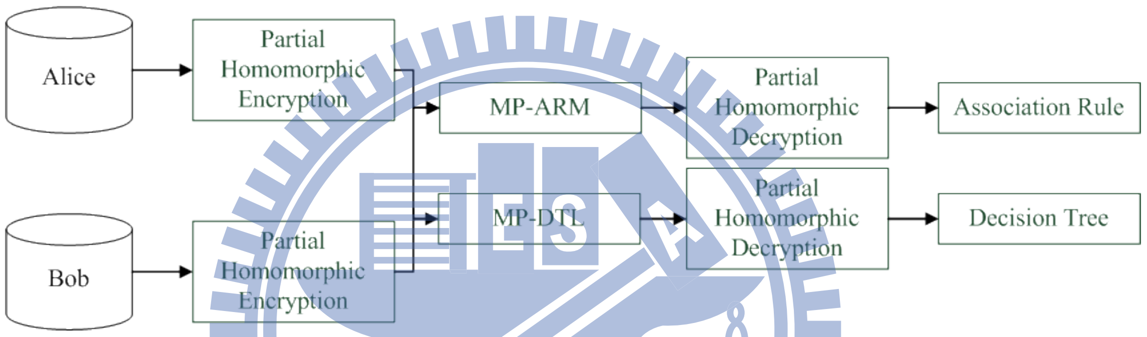
Alice
Bob
Partial Homomorphic Encryption
Partial Homomorphic Encryption
MP-ARM
MP-DTL
Partial Homomorphic Decryption
Partial Homomorphic Decryption
Association Rule
Decision Tree

Figure 1.2: The overview of proposed protocols in the process.

1.1 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 describes the background on the data mining algorithms and the cryptosystem forming the basis of our proposed protocols. Chapter 3 introduces the proposed protocol for securely perform distributed association rule mining on private. We present the other protocol for privacy-preserving decision tree learning on Chapter 4. Chapter 5 gives concluding remarks and outlines direction for future work.

Chapter 2

Background

2.1 Homomorphic Encryption

Homomorphic encryption is the scheme that allows computations to be carried out on ciphertext. The decryption of computation results match the outcome of operations performed on the plaintext. The concept of homomorphic encryption, or privacy homomorphism was first proposed to the scientific community in 1978 by Ronald Rivest, Leonard Adleman and Michael Dertouzos. A semantically secure homomorphic encryption scheme was developed and proposed by Shafi Goldwasser and Silvio Micali in 1982. In 2009, Craig Gentry proved that a completely homomorphic encryption scheme is possible.

Rivest, Aldeman and Dertouzos developed their theory based on the fact that the existing security and encryption systems severely limit the ability to manipulate data after it is encrypted and turned into ciphertext. Without the development of a homomorphic solution, “sending” and “receiving” data are the only function that can be accomplished with encrypted data. The biggest concern was the level of computing that processes the encrypted request on the encrypted data. This manipulation may reduce the security level of the encryption scheme.

With the advent and rapid expansion of cloud computing, a feasible homomorphic encryption method is crucial. Otherwise, the risk is too high to entrust sensitive data to a cloud computing service provider. If a service provider can access data in their decrypted form, the data can directly expose to malicious users. [6] proved that homomorphic encryption is viable, though the amount of computation time is a concern.

In [6], the author outlined how to create an encryption scheme that can allow data to be securely stored in a cloud environment where the owner can utilize the computational power of the cloud provider to manipulate the encrypted data. There are three main steps in [6]. An encryption scheme is constructed that is "bootstrappable". In this step, a somewhat homomorphic encryption scheme can work with its own decryption circuit. Next, an almost-bootstrappable public key encryption scheme is built using the idea of ideal lattices. Finally, the schemata are simplified, while maintaining the property of being bootstrappable.

Although [6] created a completely homomorphic encryption scheme, it remains impractical. Homomorphic encryption has evolved to be mostly secured against chosen plain-text attacks, but securing against chosen cipher-text attacks remains a problem. In addition to the security issue, the fully homomorphic schemes are so complex that the time factor has precluded their usage in many applications. Somewhat homomorphic encryption systems have been developed to address at least the time factor, using only the most efficient portions of a completely homomorphic encryption scheme.

In this thesis, we apply homomorphic encryption in realistic world. In other words, efficiency should be taken into considerations. We use partial homomorphic encryption as our mainly used encryption scheme and combine protocols design to realize the privacy-preserving data mining process. There are several efficient partial homomorphic cryptosystems:

2.1.1 Unpadded RSA

If the RSA public key is modulus m and exponent e, then the encryption of a message x is given by $E(x) = x^e \ mod \ m$. The homomorphic property is then

$$E(x_1) \cdot E(x_2) = x_1^e x_2^e \ mod \ m = (x_1 x_2)^e \ mod \ m = E(x_1 \cdot x_2).$$

2.1.2 ElGamal

In a group G, if the public key is (G, q, g, h), where $h = g^x$, and x is the secret key, then the encryption of a message m is $E(m) = (g^r, m \cdot h^r)$, for some random $r \in \{0, 1, \cdots, q-1\}$, the homomorphic property is then

$$E(x_1) \cdot E(x_2) = (g^{r_1}, x_1 \cdot h^{r_1})(g^{r_2}, x_2 \cdot h^{r_2}) = (g^{r_1+r_2}, (x_1 \cdot x_2)h^{r_1+r_2}).$$

2.1.3 Goldwasser-Micali

In Goldwasser-Micali cryptosystem, if the public key is the modulus m and quadratic non-residue x, then the encryption of a bit b is $E(b) = x^b r^2 \ mod \ m$, for some random $r \in \{0, 1, \cdots, m-1\}$. The homomorphic property is then

$$E(b_1) \cdot E(b_2) = x^{b_1} r_1^2 x^{b_2} r_2^2 = x^{b_1+b_2}(r_1 r_2)^2 = E(b_1 \oplus b_2).$$

2.1.4 Benaloh

If the public key is the modulus m and the base g with a blocksize of c, then the encryption of a message x is $E(x) = g^x r^c \ mod \ m$. for some random

$r \in \{0, 1, \cdots, m-1\}$. The homomorphic property is then

$$E(x_1) \cdot E(x_2) = (g^{x_1} r_1^c)(g^{x_2} r_2^c) = g^{x_1+x_2}(r_1 r_2)^c = E(x_1 + x_2 \; mod \; c).$$

2.1.5 Paillier Cryptosystem

The Paillier Cryptosystem [7] is a public key encryption scheme based on modular arithmetic, created by Pascal Paillier. The homomorphic property in Paillier cryptosystem is additive homomorphism as follow:

$$E_k(x) \times E_k(y) = E_k(x + y).$$

Encryption

To encrypt a message using the Paillier cryptosystem, a public key must be established first.

To construct the public key, one must choose two large primes, p and q, then calculate their product, $n = p \cdot q$. Then a semi-random, nonzero integer, g, in $\mathbb{Z}_{n^2}$, must be selected so that the order of g is a multiple of n in $\mathbb{Z}^*_{n^2}$. Thus, the public key is (n, g).

The steps of encryption is as follows:

1. Create a message, m, with $m \in \mathbb{Z}_n$.

2. Choose a random, nonzero integer, $r \in \mathbb{Z}^*_n$.

3. Compute $c \equiv g^m \cdot r^n \; mod \; n^2$.

Decryption

1. Define $L(u) = (u - 1)/n$.

2. Calculate $L(g^{\lambda(n)} \; mod \; n^2) = k$.

3. Compute $\mu \equiv k^{-1}$ mod n^2.

4. $m \equiv L(c^{\lambda(n)} mod\ n^2) \cdot \mu\ mod\ n$.

Our proposed protocols use additive homomorphic scheme to securely sum up the encrypted results, so we take Paillier cryptosystem as our encryption scheme. Also, Table 2.1 shows the key size recommended by NIST for security consideration, we implement our system with 1024-bit key size.

Table 2.1: NIST Recommended Key Size

Symmetric Key Size (bits)	RSA and Diffie-Hellman Key Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

2.2 Association Rule Mining

Association rule mining is a process that help find the confidential rules from a large amount of data. The problem can be defined as follows:

Let $I = \{i_1, i_2 ... i_n\}$ be a set of items. Let $T = \{t_1, t_2 ... t_n\}$ be a set of transactions, where each $t_i \subseteq I$. Given an itemset $X \subseteq I$, a transaction t_i contains X if and only if $X \subseteq t_i$. An association rule is an implication of the form $X \Rightarrow Y$ where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \emptyset$. The rule has support s in the transaction database DB if $s\%$ of transactions in D that contain $X \cup Y$. The association rule holds in the transaction database D with confidence c if

$c\%$ of transactions in D that contain X also contain Y. An itemset X with k items called k-itemset. The problem of mining association rules is to find all rules whose support and confidence are higher than the minimum support and confidence defined by user.

It has been shown that the problem of mining association rules can be reduced to two subproblems:

1. Find all large itemsets for a predetermined minimum support.

2. Generate the association rules from the large imtesets found.

The most crucial part affecting the performance of mining association rules is to solve the first problem efficiently.

2.2.1 Apriori Algorithm

The Apriori algorithm is an effective method for determining association rule [8]. The idea is to separate association rule mining problem into two stages:

1. The large itemsets are computed iteratively. In each iteration, the database is scanned once and all large itemsets of the same size are computed. The large itemsets are computed in the ascending order of their sizes.

2. In the first iteration, the size-1 large itemsets are computed by scanning the database once. Subsequently, in the kth iteration ($k > 1$), a set of candidate sets C_k is created by applying the candidate set generation function Apriori-gen on L_{k-1}, where L_{k-1} is the set of all large $(k-1)$-itemsets found in iteration $k-1$. Apriori-gen generates only those k-itemset whose every $(k-1)$-itemset subset is in L_{k-1}. The support

counts of the candidate itemsets in C_k are then computed by scanning the database once and the size-k large itemsets are extracted from the candidates.

In general, for both frequent itemsets XY and X, we can determine if the rule $X \rightarrow Y$ holds by computing the ratio r:

$$r = \mathit{support}(XY)/\mathit{support}(X).$$

The rule holds only if $r \geq$ minimum confidence. Note that this rule has the minimum support because XY is frequent.

2.3 Decision Tree Learning

Decision tree algorithms are the learning methods for approximating discrete-valued target function. These famous algorithms in inductive learning have been successfully applied in a broad range of tasks.

2.3.1 $ID3$ Algorithm

ID3 is a simple decision learning algorithm developed by J. Ross Quinlan in 1986 [9]. ID3 constructs a decision tree by employing a top-down greedy search through the given sets of training data to test each attribute at every node. It uses a statistics property called *information gain* to select which attribute to test at each node. Information gain measures how a given attribute separates the training examples according to their target classifications. The notation is listed as:

- R: the set of attributes;
- C: the class attribute;

- T: the set of transactions.

Entropy

It is a measure in the information theory, which characterizes the impurity of an arbitrary collection of examples. If the class attribute C takes on l different values, then the entropy $H_C(T)$ to this l-wise classification is defined as

$$H_C(T) = \sum_{i=1}^{l} -\frac{|T(c_i)|}{|T|} \log_2 \frac{|T(c_i)|}{|T|},$$

where $T(c_i)$ is the number of transactions that contain c_i. Logarithm is base 2 because entropy is a measure in terms of bits. For instance, if training data have 14 instances with 6 positive and 8 negative instances, the entropy is calculated as

$$H_C(T) = -(6/14)\log_2(6/14) - (8/14)\log_{(8/14)} = 0.985.$$

Note that the more the uniform the probability distribution, the greater the entropy. This implies that it is difficult to obtain clear information from the current distribution.

Information Gain

Let T be a set of transactions, C the class attribute with l different values, A the non-class attribute with m possibilities $(a_1, a_2, \cdots, a_m)$. $T(a_j)$ denotes the transactions containing a_j. Then the conditional entropy can be calculated as follows:

$$H_C(T|A) = \sum_{j=1}^{m} \frac{|T(a_j)|}{|T|} H_C(T(a_j)).$$

Information gain measures the expected reduction in entropy by partitioning the transaction set according to this attribute. The information gain, $Gain(A)$ of an attribute A, relative to the collection of transactions T, is defined as

$$Gain(A) = H_C(T) - H_C(T|A).$$

We can use this measure to rank attribute with the highest information gain among the attributes that have not yet been considered in the path from the root. By doing so recursively, we can acquire the decision tree in the end.

Example

Table 2.2: An original transaction database

Attribute					Classes
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	Normal	Strong	No

Assume that we have a transaction database as Table 2.2. The goal is to construct a decision tree using ID3 algorithm. We first compute the entropy which is

$$H_C(T) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940.$$

Then we test each non-class attribute A as root node of decision tree, and calculate the information gain of each non-class attribute. Here, we use "Outlook" as an example. We have three values in outlook field, including a_1(Sunny), a_2(Overcast), and a_3(Rain). Using Outlook as the attribute to partition the original transaction databases, Tables 2.3, 2.4, 2.5 can be acquired.

Table 2.3: The transaction table partitioned by outlook (Sunny)

Attribute		Classes
Day	Outlook	Play Tennis
D1	Sunny	No
D2	Sunny	No
D8	Sunny	No
D9	Sunny	Yes
D11	Sunny	Yes

Table 2.4: The transaction table partitioned by outlook (Overcast)

Attribute		Classes
Day	Outlook	Play Tennis
D3	Overcast	Yes
D7	Overcast	Yes
D12	Overcast	Yes
D13	Overcast	Yes

Table 2.5: The transaction table partitioned by outlook (Rain)

Attribute		Classes
Day	Outlook	Play Tennis
D4	Rain	Yes
D5	Rain	Yes
D6	Rain	No
D10	Rain	Yes
D14	Rain	No

For Outlook: the entropy now becomes

$$\begin{aligned} H_C(T|A(Outlook)) &= \frac{5}{15}(-(0.4)log_2(0.4) - (0.6)log_2(0.6)) \\ &+\frac{5}{15}(-(0.6)log_2(0.6) - (0.4)log_2(0.4)) \\ &= 0.647. \end{aligned}$$

$$Gain(A(Outlook)) = 0.940 - 0.647 = 0.293.$$

The information gain of all non-class attributes is listed at Table 2.6. "Outlook" has the most information gain, so we construct our decision tree

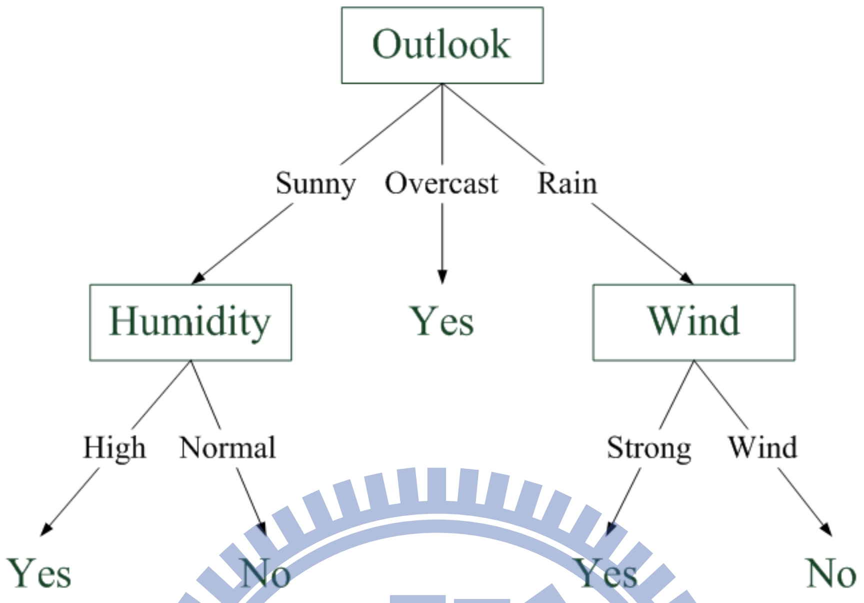

Figure 2.1: The example of ID3 decision tree.

with the root node "Outlook". By doing this recursively we can get the finally decision tree as Fig. 2.1.

Table 2.6: The calculation of information gain with different attribute

Attribute	Gain
Outlook	0.293
Temperature	0.074
Humidity	0.093
Wind	0.093

2.3.2 $ID3_\delta$ Approximation

The traditional $ID3$ algorithm chooses the best attribute that can maximize the information gain. For two attributes A_1 and A_2 with close information gain, we say that A_1 and A_2 have δ-close information gains if

$$|H_C(T|A_1) - H_C(T|A_2)| \geq \delta.$$

2.4 Secure Multi-party Computation

Substantial works have been done on secure multi-party computation. The key result is that a wide class of computations can be computed securely under reasonable assumptions. We give an overview on this work. The definitions were given by Goldreich [10]. For simplicity, we concentrate on the two-party case. Extending the definitions to the multi-party case is straightforward.

1. *Security in semi-honest model:* A semi-honest party follows the protocols using its correct input, but is free to later use what it sees during the execution of the protocol to compromise security. This is a realistic assumption because parties wanting to mine data for their mutual benefits will follow the protocols to get the correct results. Also, a protocol, being is buried in large and complex software, can not be easily altered.

2. *Yao's general two-party secure function evaluation:* Yao's general secure two-party evaluation is to express the function $f(x, y)$ as a circuit and encrypting the gates for secure evaluation [11]. Using this protocol, any two-party function can be evaluated securely in the semi-honest

model. However, the functions must have a small circuit representation. We will not detail this generic method, but adopt this generic for securely solving the millionaire problem. In comparison of any two integers securely, Yao's generic method is one of the most efficient methods, although other asymptotically equivalent but practically more efficient algorithms could be used as well [12].

Chapter 3

Privacy-Preserving Association Rule Mining

3.1 Problem Definition

Let $n = 2$ be the number of participants who perform association rule mining with their databases jointly. Each participant i has a private transaction database D_i. Given a support threshold s and confidence c, The goal is to discover all association rules which satisfy the threshold. To ensure the privacy of sensitive data owned by participants, it is required to limit the information leakage. Thus each participant knows nothing. Other participant's data can't be simulated by inputting its own data and the mining results.

We consider the following as sensitive information and should not be revealed in the progress of our protocol:

- The itemsets supported at each site;
- The local support count of an itemset at each site;

- The global support count of an itemset at each iteration k;
- Database size at each site.

3.2 Related Work

There are several privacy-preserving data mining approaches. Previous work in privacy-preserving data mining can be classified into two aspects. The first one aims at preserving participants' privacy through distorting data values [13]. The basic idea is that the distorted data doesn't reveal private information. Thus it is safe to use for mining. The distorted data and information can be used to generate an approximation to the original data distribution without revealing the original data values. The distribution can be used to provide mining results rather than mining the distorted data directly. This is primarily done by selecting split points to "bin" continuous data. Later refinement of this approach tightened the bounds on what private information is disclosed by showing that the ability to reconstruct the distribution can be used to tighten estimates of original values based on the distorted data [14].

More recently, the data distortion approach was used to boolean association rules [15], [16]. The key idea is to adjust the data values to make reconstructing the values for any individual transaction more difficult, but the rules learned from these distorted data are still available. One feature of this work is the flexibility to define privacy. The other approach is constructing decision trees, which will be discussed in Chapter 4.

The problem of privately computing association rules in vertically partitioned distributed database has also been addressed in [11]. The vertically partitioned problem occurs when each transaction is split across multiple sites, with each site having a different set of attributes for the entire set of transactions. With horizontal partitioning each site has a set of complete transactions. In relational terms, with horizontal partitioning the relation to be mined is the union of the relations at the sites. In vertical partitioning, the relations at the individual sites must be joined to get the relation to be mined. The change in the way the data is distributed makes this a much different problem form the one we address here, resulting in a very different solution.

Two-party association rule mining has been addressed owing to the hardness to preserve privacy when performing privacy-preserving association rule mining. If the database size of two databases are represented by d_1 and d_2, and the count of an itemset can be expressed as c_1 and c_2, then the inequality $\frac{c_1+c_2}{d_1+d_2} \geq s$ tests whether the itemset is frequent or not, where s is the minimum support count threshold. In [17], they used fully homomorphic encryption proposed by Gentry [6] instead of Yao's garbled circuit to test the inequality above. The main contribution of this work is the use of filly homomorphic encryption to solve the problem caused by using Yao's solution which has higher communication cause since the generated circuit can't be reused. But, when considering the efficiency to use this protocol in real system owing to the efficiency to perform fully homomorphic encryption

In order to realize the privacy preserving association rule mining in real system, we introduce a trusted participant who only helps us to decide whether the itemset is frequent or not. This participant can't learn extra information except the inequality stands or not.

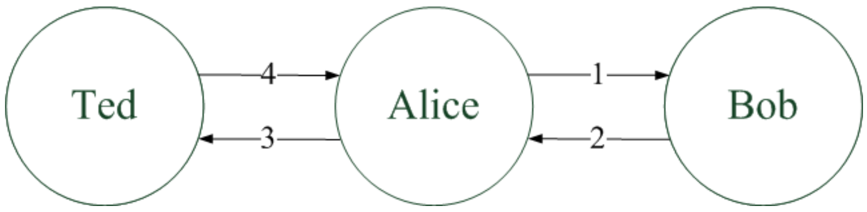

Figure 3.1: Proposed architecture (two-party case).

3.3 Proposed Protocol

We extend Apriori algorithm [8] to achieve privacy. The goal is to establish a secure version of the Apriori algorithm that minimize the information disclosure during the progress of the protocol. Our work is inspired by the method proposed in [17] for discovering association rules privately. Their protocol was based on the fully homomorphic encryption to discover the frequent rules.

Instead of using the actual support count to decide whether an itemsets is frequent or not, we use the excess support count of the itemset, i.e. by how much a support count at a participant's transaction database exceeds the threshold support s. For an itemset to be frequent, the following inequality must be true:

$$\sum_{i=1}^{n} X.sup_i \geq s \times \sum_{i=1}^{n} |D_i|,$$
$$\sum_{i=1}^{n} (X.sup_i - s \times |D_i|) \geq 0.$$

We denote $(X.sup_i - s \times |D_i|)$ as excess support count of participant i.

There are three participants in our protocol which includes Alice, Bob and Ted. Alice and Bob are two sites both hold large databases. Alice wants

Table 3.1: The notation table of MP-ARM

s	The minimum support count
c_i	The local support count at party i
$X.sup$	The global support count of an itemset X
$X.sup_i$	The local support count of X at site S_i
D	The number of transactions in DB
D_i	The number of transactions in DB_i
L_k	The set of global large k-itemsets
C_k	The set of candidate k-itemsets
pk	The public key of the first site S_1
sk	The private key of the first site S_1
$E_{pk}(m)$	Encrypt m with public key pk
$D_{sk}(c)$	Decrypt c with private key sk

to perform association rule mining with the union of both databases. All sites are assumed to be semi-honest. That is, all participants will follow the protocol honestly, but try to collect all the data received and figure out the secret as possible as they can. Unlike other protocols to implement association rule mining that uses homomorphic encryption to directly encrypt the excess support count, our protocols do not allow Alice who wants to perform the association rule mining with fake data to acquire the count of itemsets in Bob's database.

Our protocol is composed of two communication steps. First, Alice receives the encrypted results through cooperating with Bob. However, she doesn't know the encrypted support count. In the second step, Ted informs Alice whether the support count exceeds the minimum support count, helping Alice to realize that the current itemset is frequent itemset or not. By performing the protocol iteratively, Alice can get the complete mining results.

Alice can misbehave and send fake support count to Bob, or vice versa. Although the mining results will go wrong, each of them can't learn other's support count. We will discuss this situation in Section 4.6

We assume that $I = \{i_1, i_2 \cdots i_n\}$ be the set of items. Let D_{Alice} be a set of transactions owned by participant Alice. Each transaction T in D_{Alice} is an itemset that $T \subseteq I$. Alice want to check the support count in all transactions in D_{Alice} and D_{Bob} if

$$\sum_{i \in participants} (c_i \times 100 - s_i \times D_i) \geq 0,$$

if the equation before stands, that means the current itemset is frequent itemset. Fig. 3.1 represents the basic protocol used in our system model. These two communication channels are the one from Ted to Alice and from Bob to Alice. Instead of the role to perform mining association rule mining, Alice also acts as a relay point to help Bob send messages to Ted. The advantage of this approach is to hide Bob from Ted, thereby enhancing privacy in comparison of sending messages to Ted.

The protocol consists of two steps. First, let Alice collect the excess support count from all the other users. After collecting the necessary information, Alice doesn't know the exact mining results. She asks Ted for helping determine whether the current itemset is frequent or not. These protocols was detailed on algorithms 1 and 2.

3.4 Measurement

We implemented a software prototype to test the feasibility of our protocol. The prototype was executed on a machine with a 2.66 GHz Intel Core(TM)2 with 6 GB of memory, running the windows 8 operation system. We used Java as our programming language and Paillier cryptosystem implementation.

We separated the total execution time into three stages, which includes encryption, evaluation and decryption. The data or count value was encrypted by the Paillier cryptosystem with key size (security parameters) equals to 512, 1024 and 2048 which ensures the feasibility of our proposed protocol on large-scale association rule mining. The results are shown at Fig. 3.2. At the mean time, we compare our results with [17]. The results show that our protocol takes less time than previous work on encryption and evaluation stage but spend more time at decryption stage. Fig. 3.3 shows that even on larger key size, our proposed protocol can be completed in a short period (less than 1.5 sec).

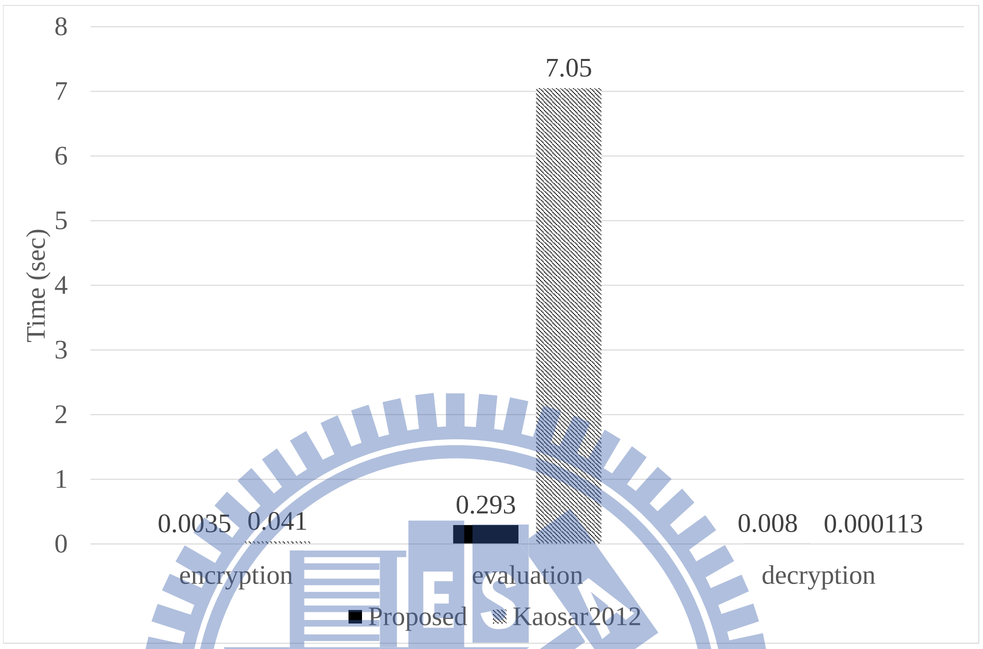

Figure 3.2: The execution time of MP-ARM and Koasar2012 [17].

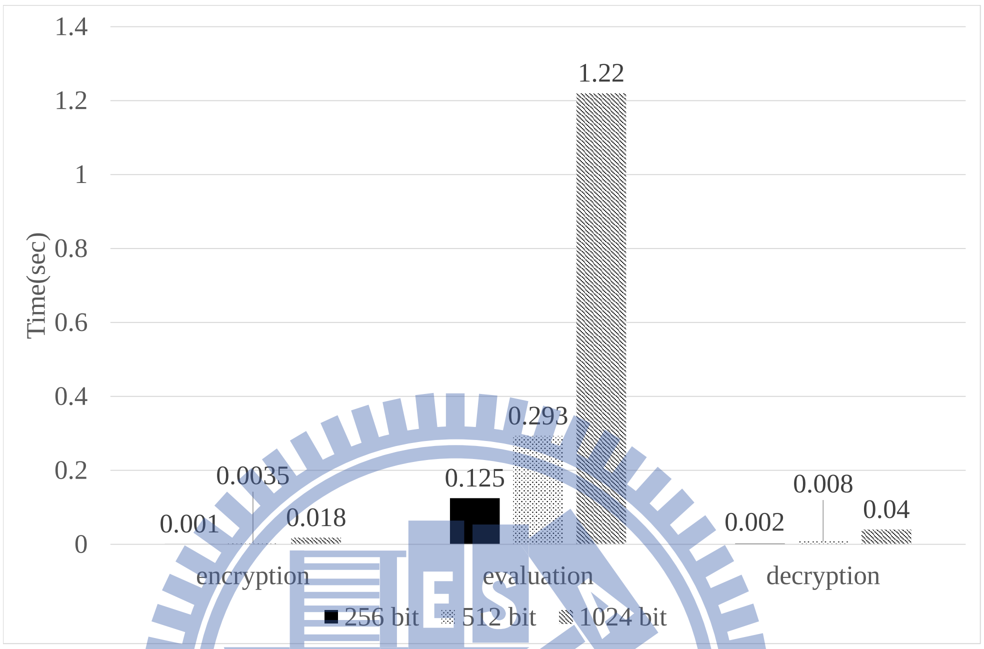

Figure 3.3: The execution time of MP-ARM with different key size.

3.5 Security Analysis

The protocol has no prevention from incorrect data usage as their input. Under this consideration, we assume that the protocol was executed under the semi-honest model to acquire the correct mining results. That is, all participants can gather all information in the progress of our protocolin the protocol should not deviate from protocol.

Owing to the semi-honest model we use in this protocol, it is assumed that all participants in the protocol can't deviate from protocol. By two secure channels that we use in this protocol, Ted doesn't know the existence of Bob. This improves privacy a little. Our protocol can prevent information leakage even if one or more participants using incorrect excess support count. We later analyze the situation if one of the participant misbehaves.

Alice misbehaves. If Alice try to use incorrect data to acquire the information from Bob. Owing to the random number k used when returning the calculation results d, the result may distribute over the field we use in the encryption scheme. Alice can't figure out the excess support count owned by Bob.

Bob misbehaves. If Bob adds the ciphertext with incorrect excess support count, Alice won't get the correct mining results from Ted. Nevertheless, Bob can't get the information about Alice's transaction database. Because the protocol mainly encrypted with Alice's public key. Each itemset at each site should be requested only once. That means if Bob receives the request from Alice who asks for the same itemset twice. He can doubt if Alice trying to resolve the information on his database. Then he can drop the protocol.

The additional participant, Ted, can't learn any information about Alice and Bob. The value d that Ted receives is the summation of excess support count of all participants. Moreover, Ted does not know what is the current itemsets in the middle of protocol.

Our protocol can provide partial collusion resistance. If either Alice or Bob colludes with Ted, then he/she can figure out the excess support count owned by the other participant. But if there are more than three participants, and two of them collude with each other can't get the information about the third participant owing to the random value k chose by the third participant.

In conclusion, If Alice or Bob use fake data in our protocol, although we can't get the correct mining results, but each of them can get additional information except the wrong mining result. Besides, Ted can't learn information from Alice or Bob without colluding. Because Ted only receive a number and judge whether the number is greater than zero or not.

3.6 Multi-Party Case

There are two approaches that can extend our proposed two-party case into multi-party case:

1. Choose one of the participants as the role Ted played in two party case.

2. Adding an additional participant as two party case.

We then give a brief discussion about the two different approaches, the first approach is the same as described in Section 4.6 except that more than two participants choosing random number to distort their true excess support count makes it more secure than two-party case. The second approach should be performed under the semi-honest model. That is, assumes Bob is the chosen one to execute the steps did by Ted in two-party case.

Additionally, this architecture uses only one key pairs to perform encryption and decryption. This can speed up the time spent on the protocol. Each participant can calculate his/her own encrypted excess support count multiplying with encrypted random value and send the encrypted results to Alice. This can be performed concurrently. As shown in Fig. 3.4, the first site can issue two parallel mining command, The first issue goes through site 2 and site 5, and the second one passes site 3 and site 4. After collecting the calculation results from both issues, site 1 can easily merge the final results directly using additive homomorphic properties on ciphertext. The above example can save almost half execution time than sequential calculation.

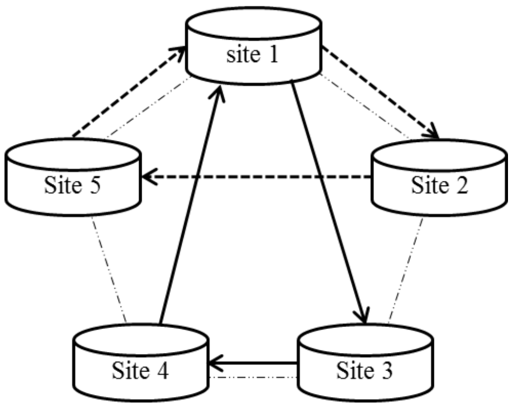

Figure 3.4: Proposed Architecture (multi-party case).

3.7 Large Scale Privacy-Preserving Association Rule Mining

In real databases, the number of itemsets and transactions are much more than the simulation in Section 3.4. In this section, we measure the relation between the execution time and the number of itemsets to test the feasibility of MP-ARM with large scale databases. The result is shown in Fig. 3.5, we observe that the execution time double after inserting a new itemset into the database. This is mainly caused by the increasing iterations needed for more itemsets.

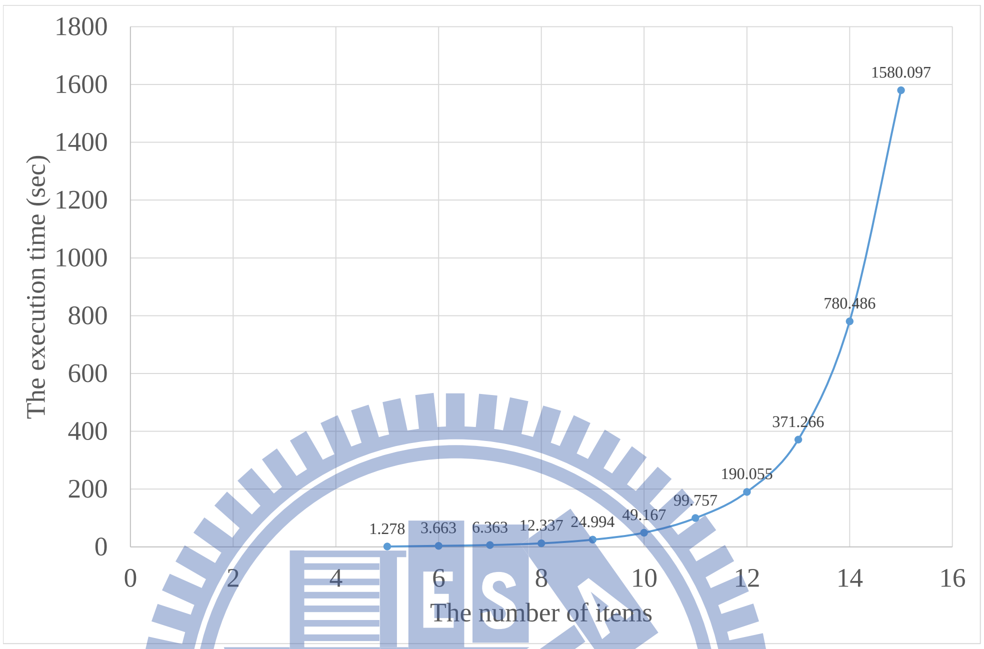

Figure 3.5: The execution time of MP-ARM with different itemsets.

Algorithm 1 Finding large k-itemsets

```
Input: $L_{k-1}$, $s$
Output: $L_k$

  Step 1 Candidate Sets Generation
  Alice generates $C_k = Apriori - gen(L_{k-1})$

  Step 2 Frequent itemsets Genration

  for $c_i \in C_k$ do
      Alice sends candidate and encrypted excess support count $E_A(T_A(c_i))$
  to Bob.
      Bob calculates $E_A(d+k)$ and sends to Alice, where $d = T_A(c_i) + T_B(c_i)$
  and $k$ is a random number in the predefined field.
      Alice decrypts $d+k$, and sends to Ted with $E_T(k)$
      Ted decrypts $k$ and calculate $d$,
      if $d \geq 0$ then
          Ted sends $E_T('YES')$ to Alice
      else
          Ted sends $E_T('NO')$ to Alice
      end if
      Alice decrypts the final result.
      if The result from Ted is 'YES' then
          Add itemset to $L_k$
      end if
  end for
```

Algorithm 2 Association Rule Generation

Input: L_k, c

Output: AR (sets of all association rules)

```
for Each frequent itemset $L_i \in L_k$ do
    Alice splits $L_i$ into all possible $i_1$ and $i_2$ such that, $L_i = \{i_1 \cup i_2\}$ and $\{i_1 \cap i_2 = \emptyset\}$
    Alice sends $\alpha_1 \leftarrow E_A(count(L_i))$ to Bob
    Bob sends $E_A(d' + k)$ and $E_T(k)$ to Alice
    Alice decrypts $d' + k$, and sends to T with $E_T(k)$
    Ted decrypts $k$ and calculate $d'$,
    if $d' \geq 0$ then
        Ted sends $E_T('YES')$ to Alice
    else
        Ted sends $E_T('NO')$ to Alice
    end if
    Alice decrypts the final result.
    if The result from Ted is 'YES' then
        $AR \leftarrow \{AR \cup i_1 \Rightarrow i_2\}$
    end if
end for
```

Chapter 4

Privacy-Preserving Decision Tree Learning

Big data are collections of large and complex datasets. It becomes difficult to process using traditional data processing algorithms. The main objective of analysis on big data is to classify large amount of raw data into meaningful cluster. In this chapter, we apply our techniques to implement a classification algorithm — ID3 decision tree learning.

4.1 Problem Definition

Consider two participants Alice and Bob possess two horizontally partitioned transaction databases T_1 and T_2 of size $|T_1|$ and $|T_2|$ respectively, where $T = \{T_1 \cup T_2\}$. First, we assume that two databases T_1 and T_2 have the same structure and the attributes name are public which is essential for joint computation. On the one hand, using universal name clearly would reduce the complexity of our protocol. On the other hand, even if the attribute names are sensitive information, it can be solved by using some distortion on naming at the preprocessing phase. Here, we simplify the protocol through using public attribute name. The goal is that two participants should jointly compute the decision tree using ID3 algorithm without additional information leakage.

4.2 Related Work

Several approaches have been proposed on privacy-preserving decision tree learning [19–27]. In [18], the goal is to securely build an ID3 decision tree where the training set is distributed between two parties. The key idea is that finding the attribute that maximizes information gain is equivalent to finding the attributes that maximizes the conditional entropy. The conditional entropy for an attribute in two-party case can be expressed as a sum of the expression of the form $(v_1+v_2)\times\log(v_1+v_2)$. In this paper, the author proposed a way to securely compute the value $(v_1 + v_2) \times \log(v_1 + v_2)$ and show how to take advantage of this function to securely build the $ID3_\delta$ decision tree. This approach treats privacy preserving data mining as a special case of secure multi-party computation [10] and not only preserve individual privacy but also tries to forbid leakage of any information.

Although we can implement this algorithm by previous method, the efficiency is still a bottleneck if we have large amount of data (big data). In the following section, we first describe how this problem can be solved by homomorphic encryption and then use the similar protocol that mentioned in the Chapter 3. In order to increase the efficiency, we can sometimes share some non-sensitive information to improve the highly efficient manipulation.

Table 4.1: The notation table of MP-DTL

R	The set of attributes
C	The class attribute
T	The set of transactions
A	The non-class attributes
$H_C(T)$	The entropy of transaction database T
$T(c_i)$	The transaction count which has class attribute c_i
$T(a_j)$	The transaction count which has non-class attribute a_j

4.3 Proposed Protocol

To perform privacy preserving decision tree learning, we can take advantage of the protocol described before with a little modification. We first analyze why we can separate the jointed calculation. The root node can be decided by calculating information gain under all non-class attributes. Assume we have n non-class attributes $A = \{a_1, a_2 \cdots, a_n\}$. The impurity of the original database is

$$H_C(T) = \sum_{i=1}^{l} -\frac{|T(c_i)|}{|T|} \log \frac{|T(c_i)|}{|T|}.$$

For a given non-class attribute A, Let A have m possible values $a_1, a_2, \cdots, a_m$ and let the class attribute C have l possible values $c_1, c_2, \cdots, c_l$.

$$\begin{aligned} H_C(T|A) &= \sum_{j=1}^{m} \frac{|T(a_j)|}{|T|} H_C(T(a(j)), \\ &= \frac{1}{|T|} \sum_{i=1}^{l} -\frac{|T(a_j, c_i)|}{T(a_j)} \cdot \log\left(\frac{|T(a_j, c_i)|}{|T(a_j)|}\right), \\ &= \frac{1}{|T|}(-\sum_{j=1}^{m}\sum_{i=1}^{l} |T(a_j, c_i)| \log(|T(a_j, c_i)|) + \sum_{j=1}^{m} |T(a_j)| \log(|T(a_j)|)). \end{aligned}$$

Therefore, showing that the conditional entropy can be jointly calculated by collecting $T(a_j, c_i)$ and $T(a_j)$. here,

$$|T(a_j)| = |T_1(a_j)| + |T_2(a_j)|,$$

$$|T(a_j, c_i)| = |T_1(a_j, c_i)| + |T_2(a_j, c_i)|,$$

where $T_b(a_j)$ is from party b and here $b = 1, 2$. This means we can jointly compute the entropy and information gain by collecting the data through different partitioned databases.

This protocols contains two steps: 1) Jointly calculate $H_C(T)$; 2) Jointly calculate $H_C(T|A)$. After executing these two steps. Ted can figure out the exact attribute that can maximize the information gain. The detail of protocol steps is listed at algorithm 3.

4.4 Measurement

The communication and computational complexity for each party is analyzed as follow (Recall that R is the set of non-class attribute and T the set of transactions):

1. The proposed protocol is repeated $m \cdot l$ times for each testing attribute where m and l are the number of non-class attribute and class attribute as mentioned before. For all $|R|$ attributes, we have $O(m \cdot l \cdot |R|)$.

2. The number of rounds needed to communicate between parties is constant. To construct the decision tree, and we have $|R|$ non-class attributes, so the total communication complexity is $O(m \cdot l \cdot |R|^2)$. Compare with the non-private distributed ID3, our proposed protocol double the communication by inserting the special participant Ted.

4.5 Analysis

4.6 Security Analysis

The protocol has no prevention from incorrect data usage as their input. Under this consideration, we assume that the protocol was executed under the semi-honest model to acquire the correct mining results. That is, all participants can gather all information in the progress of our protocolin the protocol should not deviate from protocol.

Owing to the semi-honest model we use in this protocol, it is assumed that all participants in the protocol can't deviate from protocol. By two secure channels that we use in this protocol, Ted doesn't know the existence of Bob. This improves privacy a little. Our protocol can prevent information leakage even if one or more participants using incorrect excess support count. We later analyze the situation if one of the participant misbehaves.

Alice misbehaves. If Alice try to use incorrect data to acquire the information from Bob. Owing to the random number k used when returning the calculation results d, the result may distribute over the field we use in the encryption scheme. Alice can't figure out the excess support count owned by Bob.

Bob misbehaves. If Bob adds the ciphertext with incorrect excess support count, Alice won't get the correct mining results from Ted. Nevertheless, Bob can't get the information about Alice's transaction database. Because the protocol mainly encrypted with Alice's public key. Each itemset at each site should be requested only once. That means if Bob receives the request from Alice who asks for the same itemset twice. He can doubt if Alice trying to resolve the information on his database. Then he can drop the protocol.

The additional participant, Ted, can't learn any information about Alice and Bob. The value d that Ted receives is the summation of excess support count of all participants. Moreover, Ted does not know what is the current itemsets in the middle of protocol.

Our protocol can provide partial collusion resistance. If either Alice or Bob colludes with Ted, then he/she can figure out the excess support count owned by the other participant. But if there are more than three participants, and two of them collude with each other can't get the information about the third participant owing to the random value k chose by the third participant.

In conclusion, If Alice or Bob use fake data in our protocol, although we can't get the correct mining results, but each of them can get additional information except the wrong mining result. Besides, Ted can't learn information from Alice or Bob without colluding. Because Ted only receive a number and judge whether the number is greater than zero or not.

Algorithm 3 ID3 Decision Tree Learning

```
Input: Partitioned Database $T_a(Alice), T_b(Bob)$
Output: Decision Tree
  Step 1 Jointly calculate $H_C(T)$
  for each class value $c_i \in C$ do
     1. Alice calculates the count of each $T_A(c_i)$
     2. Alice sends the encrypted count $E_A(T_A(c_i))$ to Bob
     3. Bob sends $E_A(d_i + k_i)$ and $E_T(k_i)$ to Alice, where $d_i = T(c_i) = T_A(c_i) + T_B(c_i)$
     4. Alice decrypts $d_i + k_i$, and sends to Ted with $E_T(k_i)$
     5. Ted decrypts $k_i$ and calculate $d_i$.
  end for
  Ted calculate entropy $H_C(T) = \sum_{i=1}^{l} -d_i \log d_i$

  Step 2 Jointly calculate $H_C(T|A)$
  for Each attribute $a_j \in A$ do
     1. Alice calculates her own $T_A(a_j)$ and $T_A(a_j, c_i)$
     2. Alice sends the encrypted results to Bob
     3. Bob calculates his $T_B(a_j)$ and $T_B(a_j, c_i)$
     4. Bob encrypts the result and sends $E_A(d'_i + k_i)$, $E_A(d''_j + k_j)$ and $E_T(k_j)$ to Alice, where $d'_j = T_A(a_j)+T_B(a_j)$ and $d''_j = T_A(a_j, c_i)+T_B(a_j, c_i)$
     5. Alice decrypts $d'_j + k_j$ and $d''_j + k_j$, and sends to Ted with $E_T(k_j)$
     6. Ted decrypts $k_j$, and calculate $d'_i$ and $d''_i$.
     7. Ted calculate conditional entropy $H_C(T|A)$
  end for
  Ted finds the largest information gain and sends back to Alice.

  Recursively construct the decision tree.
```

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we have introduced two privacy-preserving protocols. One for association rule mining, the other for decision tree learning. Our results show that the execution time of the protocol is much shorter than previous work.

After adding a special participant in these protocols, we improves the efficiency than previous work. The protocol was shown to be secure under the semi-honest model of multi-party computation. The security analysis is based on the mathematical hardness assumption.

The main contribution was to design protocols to reach a balance between efficiency and privacy preserving. Previous work either use Yao's garbled circuit or combine fully homomorphic encryption. The former approach spends too much time on complex circuit and the latter one can't fit into large scale distributed databases.

5.2 Future Work

Although these protocols have higher efficiency, they can not assure each participant using correct data to jointly compute the mining results. Using signature to promise the data correction may be a possible way to solve this problem.

Bibliography

[1] P. N. Tan, M. Steinbac, and V. Kumar, "Introduction to Data Mining". Addison-Wesley, 2006.

[2] T. Fawcett, "diapers and beer", 2000.

[3] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 18, no. 6, pp. 607–616, JUN 1996.

[4] S. K. Murthy, S. Kasif, and S. Salzberg, "A system for induction of oblique decision trees," *Journal of Artificial Intelligence Research*, vol. 2, pp. 1–32, 1994.

[5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[6] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices," in *STOC'09: PROCEEDINGS OF THE 2009 ACM SYMPOSIUM ON THEORY OF COMPUTING*, ser. Annual ACM Symposium on Theory of Computing, SIGACT; ACM. : ASSOC COMPUTING MACHINERY, 2009, Proceedings Paper, pp. 169–178, 41st Annual ACM

Symposium on Theory of Computing, Bethesda, MD, MAY 31-JUN 02, 2009.

[7] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *Advances in Cryptology EUROCRYPT 99*, ser. Lecture Notes in Computer Science, J. Stern, Ed. Springer Berlin Heidelberg, 1999, vol. 1592, pp. 223–238.

[8] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” *Proc. 20th Int. Conf. Very Large Data*, 1994.

[9] J. R. Quinlan, “Induction of decision trees,” *Mach. Learn*, pp. 81–106, 1986.

[10] O. Goldreich and A. Warning, “Secure multi-party computation,” 1998.

[11] A. C. C. Yao, “How to generate and exchange secrets,” in *Foundations of Computer Science, 1986., 27th Annual Symposium on*, 1986, pp. 162–167.

[12] I. Ioannidis and A. Grama, “An efficient protocol for yao’s millionaires’ problem,” in *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, 2003, pp. 6 pp.–.

[13] R. Agrawal and R. Srikant, “Privacy-preserving data mining,” *SIGMOD RECORD*, vol. 29, no. 2, pp. 439–450, JUN 2000, International Conference on Management of Data, DALLAS, TEXAS, MAY 16-18, 2000.

[14] D. Agrawal and C. C. Aggarwal, “On the design and quantification of privacy preserving data mining algorithms,” in *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles*

of database systems, ser. PODS '01. New York, NY, USA: ACM, 2001, pp. 247–255.

[15] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy preserving mining of association rules," *INFORMATION SYSTEMS*, vol. 29, no. 4, pp. 343–364, JUN 2004, 8th International Conference on Knowledge Discovery and Data Mining (KDD 2002), EDMONTON, CANADA, JUL 23-26, 2002.

[16] S. Rizvi and J. Haritsa, "Maintaining data privacy in association rule mining," *the 28th international conference on Very Large Data*, 2002.

[17] M. G. Kaosar, R. Paulet, and X. Yi, "Fully homomorphic encryption based two-party association rule mining," *Data and Knowledge Engineering*, vol. 76-78, pp. 1–15, June 2012.

[18] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '00. London, UK, UK: Springer-Verlag, 2000, pp. 36–54.

[19] W. Du and Z. Zhan, "Using randomized response techniques for privacy-preserving data mining," *the conference on Knowledge discovery and data mining*, pp. 505–510, 2003.

[20] Z. Zhong and R. Wright, "Privacy-preserving classification of customer data without loss of accuracy," *SIAM International Conference on Data Mining*, pp. 1–11, 2005.

[21] R. Wright and Z. Yang, "Privacy-preserving Bayesian network structure computation on distributed heterogeneous data," *Proceedings of the*

2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04, p. 713, 2004.

[22] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," *Knowledge and Data Engineering*, pp. 2–13, 2004.

[23] D. Cheung, V. Ng, and a.W. Fu, "Efficient mining of association rules in distributed databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 911–922, 1996.

[24] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *Journal of Cryptology*, vol. 15, no. 3, pp. 177–206, June 2002.

[25] J. Vaidya and C. Clifton, "Privacy preserving association rule mining in vertically partitioned data," *ACM SIGKDD Conference on Knowledge discovery and data mining*, 2002.

[26] I. Saleh and A. Mokhtar, "P3ARM: privacy-preserving protocol for association rule mining," *Information Assurance Workshop, 2006 IEEE*, pp. 76–83, 2006.

[27] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, vol. 16, no. 9, pp. 1026–1037, SEP 2004.

Vita

Yin-Ming Chang

He was born in Taiwan, R. O. C. in 1985. He received a B.S. in Chemical Engineering from National Taiwan University in 2007. From June 2011 to July 2013, he worked his Master degree in the Mobile Communications and Cloud Computing Lab in the Department of Computer Science at National Chiao-Tung University. His research interests are in the field of security issue on cloud environment.