Privacy-Preserving Mining of Association Rules on Distributed Databases

Chin-Chen Chang[†,††], Jieh-Shan Yeh[†††], and Yu-Chiang Li[††]

[†]Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan
[††]Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan
[†††]Department of Computer Science and Information Management, Providence University, Taichung, Taiwan

Summary
Data mining techniques can extract hidden but useful information from large databases. Most efficient approaches for mining distributed databases suppose that all of the data at each site can be shared. However, source transaction databases usually include very sensitive information. In order to obtain an accurate mining result on distributed databases and to preserve the private data that is accessed, Kantarcioglu and Clifton proposed a scheme to mine association rules on horizontally partitioned data. This study proposes an Enhanced Kantarcioglu and Clifton Scheme's (EKCS), which is a two-phase, privacy-preserving, distributed data mining scheme. It is based on the Kantarcioglu and Clifton's Scheme (KCS) and reduces the quantities of global candidates that are encrypted and reduces the transmission load without raising the risk of itemsets leak in the first phase. Moreover, to increase the security against collusion in the second phase, this study proposes two protocols to be applied in the communication environment with or without a trusted authority, respectively.

Key words:
data mining, frequent itemset, privacy-preserving, security

1. Introduction

The goal of recent advances in data mining techniques is to efficiently discover valuable and non-obvious knowledge from large databases [9, 16]. The mining of association rules plays an important role in various data mining fields, such as financial analysis, the retail industry and business decision-making [9].

Modern organisations have their own databases, located in different places. Most mining techniques assume that the data is centralised or the distributed amounts of data can efficiently move to a central site to become a single model. However, organisations may be willing to share only their mining models, not their data. These centralised techniques have a high risk of unexpected information leaks when data is released [5]. Organisations urgently require evaluation to decrease the risk of disclosing information. Privacy-Preserving Data Mining (PPDM) can run a data mining algorithm to obtain mutually beneficial global mining objectives without exposing private data [27]. Therefore, PPDM has become an important issue in many data mining applications.

A simple method of PPDM in distributed databases is to perturb the original data. The procedure of transforming the original database into a new one that hides some sensitive association rules is called the *sanitisation process* [5, 12]. Performing a mining process on the sanitised database can reduce the risk of revealing the sensitive information [5, 12, 19, 20, 24, 26]. However, the mining result on the sanitised database is less precise than that of the original database.

In some business environments, the data mining may need to be processed among databases. Nevertheless, data may be distributed among several sites, but none of the sites is allowed to expose its database to another site.

Consider the following scenario: Some insurance companies have their own databases that record their insured's information. For mutual benefit, these companies decide to cooperate for insurance fraud detection by distributed data mining. The data mining model must be high accurate to detect fraud, because a mistake results in great loss of revenue or great amounts of pay. Moreover, insurance companies cannot share data about their customers with other companies, owing to the restriction laws (and having a high competitive edge). They may share knowledge about fraudulent insurance records, but not their data. Each company attempts to share their "block-box" models to discover more interesting rules on the whole shared information than that on their own database, and can protect the exclusive records that other companies may find [25].

Secure Multiparty Computation (SMC) [6, 7] employs distributed algorithms in a secure manner. SMC not only preserves individual privacy, but also aims to preserve leakage of any information other than the final result. However, traditional SMC methods require a high communication overhead. They do not scale well with the

database size. Therefore, this study focuses on the problem of privacy-preserving mining frequent itemsets in multiple distributed databases, in which each transaction entirely belongs to only one site, with a low communication requirement and without perturbing the original data.

Kantarcioglu and Clifton proposed a two-phase scheme for privacy-preserving distributed mining of association rules on horizontally partitioned data [15]. This scheme transmits and encrypts large amounts of candidates in the first phase. In the second phase, the Kantarcioglu and Clifton's scheme has a high risk of collusion between sites. Therefore, this study proposes the Enhanced Kantarcioglu and Clifton's Scheme (EKCS) to speed up the process of the first phase and reduce the security risk in the second phase.

The rest of this paper is organised as follows: Section 2 presents the background and an overview of the current methods for solving the problem of PPDM. Section 3 explains the proposed Enhanced Kantarcioglu and Clifton's Scheme (EKCS). Finally, we conclude in Section 4 with a summary of our work.

2. Background and Related Work

2.1 Association rule mining

Agrawal et al. first introduced the problem of association rule mining over a market-basket transaction database in [3]. An example of a rule is as follows: 50% of transactions that purchase an 21″ LCD monitor also purchase a video game. Such rules can provide valuable information on the customer buying behavior. The formal statement of association rule mining is as follows [3, 4]:

Let $I = \{i_0, i_1, \ldots, i_{n-1}\}$ be the set of items. Let *DB* be a transaction database, where each transaction *T* in DB is a set of items, that is, $T \subseteq I$. A set of items *X* is also referred as an *itemset*. An itemset that contains *k* items is called a *k*-itemset. A transaction *T* supports an itemset *X*, if $X \subseteq T$. An association rule is denoted as the form $X \Rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$ and $X \bigcap Y = \phi$ (For example, $I = \{A, B, C, D, E\}$, $X = \{A, C\}$ and $Y = \{B, E\}$). A rule $X \Rightarrow Y$ includes two important attribute values, *support* and *confidence*, denoted as $Sup(X \Rightarrow Y)$ and $Conf(X \Rightarrow Y)$, respectively. Given two user pre-specified minimum support (*minSup*) and minimum confidence (*minConf*) thresholds, a rule $X \Rightarrow Y$ holds in *DB* if and only if $Sup(X \Rightarrow Y) \geq minSup$ and $Conf(X \Rightarrow Y) \geq minConf$. The support value *s*% of $X \Rightarrow Y$ means that *s*% of transactions in *DB* contain $X \bigcup Y$. The confidence value *c*% of $X \Rightarrow Y$ means that the transactions contain *X* in *DB* in which *c*% of them also contain *Y*. The itemset $X \bigcup Y$ with length *k* is called a frequent *k*-itemset if $Sup(X \Rightarrow Y) \geq minSup$.

The process of association rule mining includes two main sub-problems: the first is to discover all frequent itemsets; the second is to use these discovered frequent itemsets to generate association rules. Since each association rule can easily be derived from the corresponding frequent itemsets, the overall performance of the association rule mining is determined by the first sub-problem. Therefore, researchers usually focus on efficiently discovering frequent itemsets. Agrawal et al. presented the Apriori algorithm to efficiently identify frequent itemsets [4]. Apriori is a level-by-level algorithm including multiple passes. In each pass, Apriori generates a candidate set of frequent *k*-itemsets (frequent itemsets with length *k*). Each frequent k-itemset is combined from two arbitrary frequent (*k*-1)-itemsets, in which the first *k*-2 items are identical. Then, Apriori scans the entire transaction database to determine the frequent *k*-itemsets. The process is repeated for the next pass until no candidate can be generated. Apriori employs the downward closure property to efficiently generate candidates in each pass. The property indicates that no subset of a frequent itemset is infrequent; otherwise the itemset is infrequent. The property can be used to eliminate useless candidates to speed up the mining process. Other methods have been proposed to efficiently discover frequent itemsets, such as level-wise algorithms [3, 4, 8, 22] and pattern-growth methods [2, 13, 14, 17].

2.2 Distributed association rule mining

Association rule mining in a very large database may require substantial processing power or be operated on a distributed system. Moreover, many large databases are distributed in nature [10]. Therefore, several algorithms for parallel mining of association rules have been proposed [1, 10]. Assume that a distributed system has *n* sites $S_0, S_1, \ldots, S_{n-1}$ and the transaction database *DB* is horizontally divided into *n* non-overlapping partitions $db_0, db_1, \ldots, db_{n-1}$, where $DB = db_0 \cup db_1 \cup \ldots \cup db_{n-1}$, $db_i \cap db_j = \phi$, $0 \leq i \neq j \leq n-1$. Each partition db_i is assigned to site S_i, and *DB* is horizontally distributed. Clearly, $|DB| = |db_0| + |db_1| + \ldots + |db_{n-1}|$.

$X.sup_i$ is the local support count of itemset *X* at site S_i, for $0 \leq i \leq n-1$. The global support count of *X* in *DB* is given as $X.sup = \sum_{i=0}^{n-1} X.\sup_i$. *X* is globally frequent if $X.sup \geq minSup \times |DB|$. Similarly, *X* is locally frequent if $X.sup_i \geq minSup \times |db_i|$. Let F_k be the set of all global

frequent k-itemsets and $LF_{k(i)}$ be the set of all local frequent k-itemsets at site S_i, for $0 \le i \le n-1$. If DB is divided into n partitions, then any global frequent itemset must appear as a frequent itemset in at least one of the n partitions [23]. Therefore, we get $F_k \subseteq \bigcup_{i=0}^{n-1} LF_{k(i)}$; the characteristic can be used to efficiently discover global frequent itemsets. Cheung et al. proposed a fast algorithm, Fast Distributed Mining (FDM) of association rules, for distributed association rule mining [10]. The FDM algorithm is briefly described as follows:

1. In each site, like Apriori, FDM discovers local frequent k-itemsets in each pass.
2. In each pass, each site broadcasts $LF_{k(i)}$ and calculates the local support value of itemsets in $\bigcup_{i=0}^{n-1} LF_{k(i)}$.
3. In each pass, each site broadcasts the local support values of itemsets in $\bigcup_{i=0}^{n-1} LF_{k(i)}$.

Therefore, each site can determine the global frequent k-itemsets F_k.

2.3 Secure Multiparty Computation

A Secure Multiparty Computation (SMC) problem is defined a situation in which some information can be exchanged by ideal functions without a leak of knowledge other than the final result [6, 7] among multiparties. The generic techniques for SMC have high polynomial-time complexity, resulting in it sometimes being impractical [6, 7]. Several studies focus on finding efficient privacy-preserving algorithms for specific problems, such as privacy-preserving computation of decision trees [18], and mining of vertically partitioned databases [11].

2.4 Privacy-preserving mining on horizontally partitioned databases

Distributed association rule mining techniques can discover association rules among multiple sites [1, 10]. They do not require that each site discloses the individual database, but each site is required to exchange all global candidate itemsets and the corresponding support counts with each other. If the support count for each global candidate itemset in each individual site is sensitive, the above approach reveals such sensitive information to other competition companies. Therefore, to enhance the security of distributed mining and reduce the computation complexity of SMC, Kantarcioglu and Clifton proposed a secure scheme for privacy-preserving association rule mining on horizontally partitioned databases [15].For the simplicity, we refer Kantarcioglu and Clifton's Scheme as KCS for the rest of the paper. For this issue, Veloso et al. also proposed an efficient method to speed up the global candidate generation and concern the privacy-preserving for discovering frequent itemsets on distributed databases [25].

For a two-party case, no doubt, a site obtains the global support count of an itemset X, and the local support count of X in another site must be revealed by a simple subtraction operation. Therefore, KCS preserves the privacy of individual site results for three or more parties [15].

Let n be the number of sites, where $n \ge 3$. Each site maintains a private transaction database db_i, where $0 \le i \le n-1$. Users assign the *minSup* and *minConf* values for the global database $DB = db_0 \cup db_1 \cup \ldots \cup db_{n-1}$, where $db_i \cap db_j = \phi$ and $0 \le i \ne j \le n-1$. KCS discovers all association rules satisfying the two thresholds in the disclosure restriction, in which no site can access the data and know the support count for any global candidate itemset of any other site except for its own data and the final result. The KCS process includes two phases: securely generating candidate frequent itemsets and finding global frequent itemsets without revealing support count. KCS is described as follows:

In the first phase, KCS applies commutative encryption [21] to preserve the global candidate itemsets in each site. Each site encrypts its own local frequent itemsets and some fake itemsets, then sends the encrypted itemsets to the next site until all sites have encrypted all itemsets. Then, KCS merges all encrypted itemsets to S_0 by eliminating duplicates. All encrypted itemsets are then decrypted site by site. The site S_{n-1} obtains the all global candidate itemsets and broadcasts them to each site. As shown in Figure 1, $\{A, B\}$, $\{A, C\}$ and $\{A, D\}$ are the local frequent 2-itemsets in S_0, S_1 and S_2, respectively. The top of Figure 1 shows that S_0 has merged five encrypted itemsets by three sites including three local frequent itemsets and two fake itemsets $\{\{A, E\}, \{A, F\}\}$. S_0 decrypts the five three-scale encrypted itemsets and sends them to S_1. Similarly, S_1 decrypts the five two-scale encrypted itemsets and sends them to S_2. Finally, S_2 obtains the five global candidate itemsets $\{\{A, B\}, \{A, C\}, \{A, D\}, \{A, E\}, \{A, F\}\}$ and broadcasts them to each site.

After all global candidate itemsets are generated, KCS performs the second phase to test whether each itemsets is frequent. Start with an initial site S_i, $0 \le i \le n-1$, for any candidate itemset X, S_i selects a distinct random number R_i and adds R_i with an excess support count of X ($X.sup_i$ - $minSup \times |db_i|$) then sends the result to next site. Except for the initial site S_i, each site S_j does not select a random number and only adds the calculating result of the previous site of X with $X.sup_i$ - $minSup \times |db_i|$. The final site S_j obtains the result to determine whether X is globally

frequent, $j \equiv n+i-1 \pmod{n}$. In Figure 2, KCS selects the random number R_0 = 19 for 2-itemset {*A*, *B*} in S_0, meanwhile $\{A, B\}.sup_0$ =18, $|db_0|$ = 100 and *minSup* = 10%. KCS sends the result, $r_0 = 27$ (19+18-10%×100), to site S_1. Site S_1 adds 27 with the excess support count -3 (7-10%×100) and sends the result 24 to site S_2. Site S_2 adds 24 with the excess support count -8 (12-10%×200) and obtains the final result, $r_2 = 16$. Since the final result is less than the random number R_0, itemset {*A*, *B*} is infrequent.

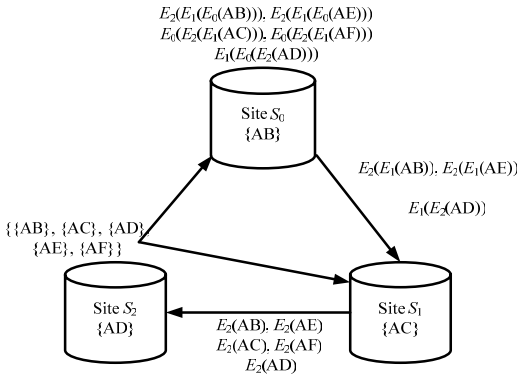

Fig. 1. Generating global candidate 2-itemsets

3. Enhanced Kantarcioglu and Clifton's Scheme (EKCS)

Like KCS, the proposed Enhanced Kantarcioglu and Clifton's Scheme (EKCS) is also a two-phase method. The useful notations of the proposed scheme are described as follows:

DB: The global database
minSup : The pre-specified minimum support threshold
X: An itemset
S_i: A local sites, for $0 \le i \le n-1$
db_i : A local database on site S_i
$X.sup_i$: The local support count of *X* at site S_i
X.sup: The global support count of *X*, where $X.sup = \sum_{i=0}^{n-1} X.sup_i$
F: The set of all global frequent itemsets in *DB*
$LF_{(i)}$: All local frequent itemset at site S_i

3.1 First phase of EKCS

This study proposes EKCS to reduce the communication overhead of KCS in the first phase. KCS requires *k* round of communication to transmit all local frequent itemsets during the distributed mining operation. According to the downward closure property of Apriori, a frequent *k*-itemset contains 2^k-1 frequent sub-itemsets. A frequent itemset has no frequent superset is called a maximum frequent itemset (MFI). In database *DB*, the set of all MFIs by deleting the redundant frequent sub-itemsets from *F* can represent all frequent itemsets. In other words, once all maximum frequent itemsets are found, it is straightforward to obtain all frequent itemsets. Veloso et al. [25] have proposed another scheme, which applied the set of all MFIs to reduce the number of transmitted itemsets, and to speed up the first phase of KCS. However, this scheme requires a trusted authority to combine all maximum frequent itemsets among sites.

In the first phase, EKCS combines the advantages of Veloso et al. and KCS schemes to efficiently unite all global candidate itemsets. After each site S_i discovers all local frequent itemsets, $LF_{(i)}$, EKCS selects local frequent itemsets from longest *k*-itemsets to 2-itemsets to delete their frequent subsets from $LF_{(i)}$ level-by-level. Meanwhile, EKCS adds some fake itemsets into $LF_{(i)}$ and deletes their sub-itemsets from $LF_{(i)}$. In each site, the final $LF_{(i)}$, which only contains maximum frequent itemsets and some fake itemsets, is denoted as $LMF_{(i)}$. Then, EKCS applies commutative encryption to encrypt each itemset in $LMF_{(i)}$. Each site S_i encrypts itemsets in $LMF_{(i)}$ and sends the encrypted itemsets to the next site S_j ($j = i+1 \bmod n$), then S_j encrypts them and sends them to next, until itemsets have be encrypted by all sites. The mergence and decryption steps are similar to KCS. After the site, S_{n-1}, obtains all local maximum frequent itemsets *LMF* (including fake itemsets), EKCS merges and deletes the redundant itemsets (the redundant itemset is a subset of another arbitrary itemset in *LMF*) and broadcasts them to every other site. These fake itemsets must be globally infrequent and do not alter the counting result of global frequent itemsets in the second phase.
EKCS and KCS utilize the same encryption technique in this phase. However, EKCS requires only one circular round of encryption processes. Clearly, EKCS transmits less itemsets than KCS. Therefore, EKCS is a more efficient method than KCS without increasing the security risk.

3.2 Second phase of EKCS

Each site receives *LMF* from the first phase result of EKCS. All global candidates can be straightforwardly derived from *LMF*. The second phase determines whether each global candidate itemset is globally frequent. A global frequent itemset, *X*, satisfies the inequality $X.sup \ge minSup \times |DB|$. Therefore,

$$X.sup - minSup \times |DB|$$

$$= \sum_{i=0}^{n-1} X.sup_i - minSup \times \sum_{i=0}^{n-1} |db_i|$$

$$= \sum_{i=0}^{n-1} (X.sup_i - minSup \times |db_i|) \geq 0 . \quad (1)$$

Equation 1 can be used to determine whether a candidate itemset is globally frequent or not, without revealing $X.sup_i$ or $|db_i|$. The value, $X.sup_i - minSup \times |db_i|$, is called the excess support count at site S_i.

In the second phase, KCS cannot resist the collusion attack. Moreover, if an attacker intercepts the input and output data from site S_i, each intercepted itemset can easily be derived whether it is locally frequent by a simple subtraction operation. For example, in Figure 2, an attacker acquires the r_0 and r_1 values of itemset $\{A, B\}$ by monitoring S_1. $\{A, B\}$ is locally infrequent in S_1 because the excess support count $r_1 - r_0 = 24 - 27 = -3 < 0$. If $|db_1|$ has also been leaked out, the local support count $(10\% \times 100 - 3)$ of $\{A, B\}$ appears.

Therefore, this study proposes two protocols, which have the resilience against collusions, to be applied in a communication environment with or without a trusted authority, respectively.

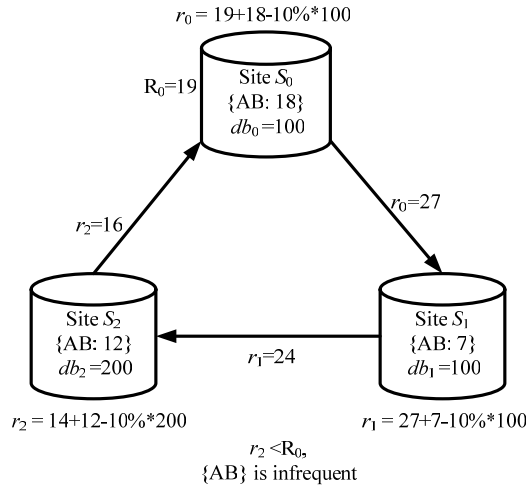

Fig. 2. Determining if the candidate 2-itemset $\{A, B\}$ is globally frequent

Protocol A (With a trusted authority)

Instead of generating a random number only at an initial site for each global candidate itemset, the trusted authority (TA) generates an individual random number R_i for each site S_i. Then, each site S_i computes the locally resulting value r_i, where $r_i = X.sup_i - minSup \times |db_i| + R_i$. That is, the resulting value is the sum of the locally excess support value and the corresponding random number. Let $Rsum = \sum_{i=0}^{n-1} R_i$. The globally excess support value of a global candidate itemset X, GE, can be calculated as follows:

$$GE = X.sup - minSup \times |DB|$$

$$= \sum_{i=0}^{n-1} (X.sup_i - minSup \times |db_i|)$$

$$= \sum_{i=0}^{n-1} X.sup_i - minSup \times \sum_{i=0}^{n-1} |db_i| + (Rsum - Rsum)$$

$$= \sum_{i=0}^{n-1} X.sup_i - minSup \times \sum_{i=0}^{n-1} |db_i| + \sum_{i=0}^{n-1} R_i - Rsum$$

$$= \sum_{i=0}^{n-1} (X.sup_i - minSup \times |db_i| + R_i) - Rsum$$

$$= \sum_{i=0}^{n-1} r_i - Rsum$$

If $GE \geq 0$, X is a frequent itemset; otherwise, X is infrequent. In a communication environment with a trusted authority, the steps of the protocol are as follows:

Step1. For each global candidate itemset, a trusted authority (TA) generates n random number $\{R_0, R_1, \ldots, R_{n-1}\}$; then, TA calculates $Rsum = \sum_{i=0}^{n-1} R_i$.

Step 2. For each global candidate itemset, TA distributes the number pair $(R_i, Rsum)$ to site S_i over a secured channel, then each site S_i calculates the locally resulting value r_i, where $r_i = X.sup_i - minSup \times |db_i| + R_i$.

Step 3. Each site S_i broadcasts the locally resulting value r_i of each global candidate itemset X to every other site.

Step 4. Each site S_i computes the globally excess support count, $GE = \sum_{i=0}^{n-1} r_i - Rsum$, of each global candidate itemset. Then, it determines whether each global candidate itemset is globally frequent or not.

Example 3.1:

Consider three local sites S_0, S_1 and S_2. The transaction numbers of db_0, db_1 and db_2 are 100, 100 and 200, respectively. Let $X = \{A, B\}$ be a global candidate itemset. As shown in Figure 3, the support counts of X in db_0, db_1 and db_2 are 18, 7 and 12, respectively. Let the minimum support threshold $minSup$ be 10%. The following steps determine if X is globally frequent.

Step 1. TA randomly selects $R_0 = 9$, $R_1 = 12$ and $R_2 = 7$; $Rsum = 28$.

Step 2. Over a secured channel, TA sends (9, 28), (12, 28) and (7, 28) to S_0, S_1 and S_2, respectively. S_0, S_1 and S_2 independently calculate $r_0 = 18 - 10\% \times 100 + 9 = 17$, $r_1 = 7 - 10\% \times 100 + 12 = 9$ and $r_2 = 12 - 10\% \times 200 + 7 = -1$, respectively.

Step 3. Sites S_0, S_1 and S_2 broadcast r_0, r_1 and r_2 to every other site, respectively.

Step 4. Each site computes $GE = 17 + 9 - 1 - 28 = -3 < 0$. Therefore, X is globally infrequent.

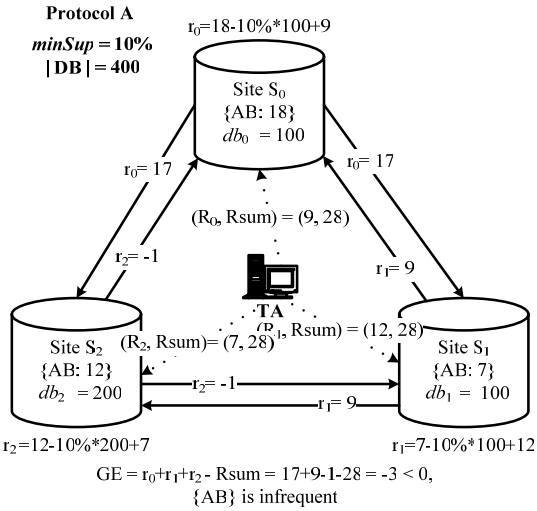

Fig. 3. Determining if the global candidate 2-itemset {*A*, *B*} is globally frequent

Protocol B (Without a trusted authority)

Instead of generating a random number for each global candidate itemset only at an initial site, EKCS generates an individual random number R_i for each site S_i. Then, each site S_i sends R_i to the next site R_j, where $j \equiv i+1 \pmod n$. Then, the initial site S_i computes the locally resulting value r_i, where $r_i = X.sup_i - minSup \times |db_i| + R_i - R_j$ and $j \equiv n+i-1 \pmod n$. Except for the initial site S_i, each site $S_{i'}$ computes the locally resulting value $r_{i'}$, where $r_{i'} = r_j + X.sup_{i'} - minSup \times |db_{i'}| + R_{i'} - R_j$ and $j \equiv n+i'-1 \pmod n$. Let S_0 be the initial site. The globally excess support value of a global candidate itemset X, GE, can be calculated as follows:

$$GE = X.sup - minSup \times |DB|$$
$$= \sum_{i=0}^{n-1} X.sup_i - minSup \times \sum_{i=0}^{n-1} |db_i| + \sum_{i=0}^{n-1} R_i - \sum_{i=0}^{n-1} R_i$$
$$= \sum_{i=0}^{n-1} (X.sup_i - minSup \times |db_i| + R_i - R_j) \text{ , where } j \equiv n+i-1 \pmod n$$
$$= r_0 + \sum_{i=1}^{n-1} (X.sup_i - minSup \times |db_i| + R_i - R_j)$$
$$= r_1 + \sum_{i=2}^{n-1} (X.sup_i - minSup \times |db_i| + R_i - R_j)$$
$$= \ldots$$
$$= r_{n-2} + \sum_{i=n-1}^{n-1} (X.sup_i - minSup \times |db_i| + R_i - R_j)$$
$$= r_{n-1}$$

If $GE = r_{n-1} \geq 0$, X is a frequent itemset; otherwise, X is infrequent. In a communication environment without a trusted authority, the steps of the protocol are as follows:

Step1. Each site S_i randomly selects R_i, and sends R_i to site S_j via a secured channel, where $j \equiv i+1 \pmod n$.

Step 2. Select an initial site S_i, then calculate $r_i = X.sup_i - minSup \times |db_i| + R_i - R_j$ to site $S_{i'}$, where $j \equiv n+i-1 \pmod n$ and $i' \equiv i+1 \pmod n$.

Step 3. For each non-initial site S_i, EKCS calculates $r_i = r_j + X.sup_i - minSup \times |db_i| + R_i - R_j$ and sends r_i to the next site $S_{i'}$, where $j \equiv n+i-1 \pmod n$ and $i' \equiv i+1 \pmod n$. Then, it determines whether each global candidate itemset is globally frequent or not at the final site.

Step 4. The final site S_i broadcasts the determined result of each global candidate itemset (X is frequent or not) to every other site.

Example 3.2:

Consider the example as shown in Figure 4. Let X = {*A*, *B*} be a global candidate frequent itemset. The transaction number of each site, the local support count of each site, and the minimum support threshold are the same as given in Example 3.1. The following steps determine if X is globally frequent.

Step 1. Sites S_0, S_1 and S_2 randomly select $R_0 = 9$, $R_1 = 12$ and $R_2 = 7$, respectively. S_0 sends 9 to S_1; S_1 sends 12 to S_2; S_2 sends 7 to S_0.

Step 2. Let S_0 be the initial site, EKCS calculates $r_0 = 18 - 10\% \times 100 + 9 - 7 = 10$ and sends r_0 to S_1.

Step 3. Site S_1 calculates $r_1 = r_0 + (7 - 10\% \times 100 + 12 - 9) = 10$. Then, S_1 sends r_1 to S_2. Finally, site S_2 computes $r_2 = r_1 + (12 - 10\% \times 200 + 7 - 12) = -3$. Therefore, X is globally infrequent.

Step 4. Site S_2 broadcasts that X is globally infrequent to sites S_0 and S_1.

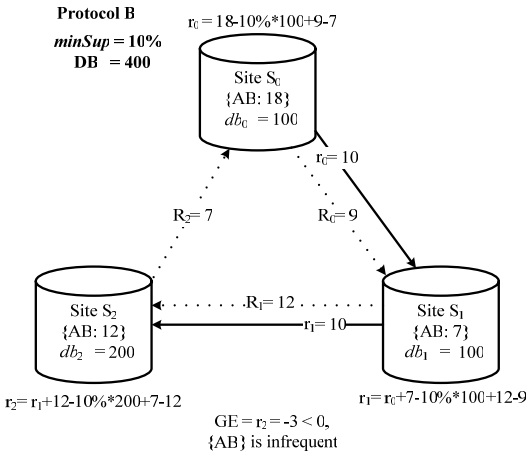

Fig. 4. Determining if the global candidate 2-itemset {*A*, *B*} is globally frequent

4. Conclusions

Data mining techniques are very useful in extracting interesting information from databases. In this competitive but also cooperative business environment, companies need to share information with others, but not sharing the data. The research of privacy-preserving data mining on distributed databases has become an important issue. This study proposes an Enhanced Kantarcioglu and Clifton Scheme (EKCS) based on the two-phase method of the Kantarcioglu and Clifton Scheme (KCS). In the first phase, EKCS reduces the number of itemsets to be encrypted and transmitted without increasing the security risk. Furthermore, in the second phase, this study introduces two protocols for enhancing security against collusion.

Now, we are investigating the development of superior privacy-preserving algorithms to further reduce computation complexity and increase the security without sharing the data in distributed database environments.

References

[1] Agrawal, R. and Shafer, J.C. (1996) "Parallel mining of association rules," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 8, No. 6, pp.929-969.

[2] Agarwal, R.C., Aggarwal, C.C. and Prasad, V.V.V. (2001) "A tree projection algorithm for generation of frequent itemsets," *Journal of Parallel and Distributed Computing*, Vol. 61, No. 3, pp.350-371.

[3] Agrawal, R., Imielinski, T. and Swami, A. (1993) "Mining association rules between sets of items in large databases," *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, Washington, D.C., May, pp.207-216.

[4] Agrawal, R. and Srikant, R. (1994) "Fast algorithms for mining association rules," *Proceedings of the 20th International Conference on Very Large Data Bases*, Santiago, Chile, September, pp.487-499.

[5] Atallah, M., Bertino, E., Elmagarmid, A., Ibrahim, M. and Verykios V. (1999) "Disclosure limitation of sensitive rules," *Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange*, Chicage, IL, November, pp.45-52.

[6] Brickell, J. and Shmatikov, V. (2005) "Privacy-preserving graph algorithm in the semi-honest model," in Roy, B. (Ed.): *Lecture Notes in Compute Science*, Vol. 3788, Springer-Verlag, pp.236-252.

[7] Canetti, R. (2000) "Security and composition of multiparty cryptographic protocols," *Journal of Cryptology* Vol. 13, No. 1, pp.143-202.

[8] Chang, C.-C. and Lin, C.-Y. (2005) "Perfect hashing schemes for mining association Rules," *The Computer Journal*, Vol. 48, No. 2, pp.168-179.

[9] Chen, M.-S., Han, J. and Yu, P.S. (1996) "Data mining: an overview from a database perspective," *IEEE Transactions on Knowledge Data Engineering*, Vol. 8. No. 6, pp.866-883.

[10] Cheung, D.W.-L., Han, J., Ng, V.T.Y., Fu, A.W.-C. and Fu, Y. (1996) "A fast distributed algorithm for mining association rules," *Proceedings of the 1996 International Conference on Parallel and Distributed Information Systems*, Miami Beach, Florida, December, pp.21-42.

[11] Dwork, C. and Nissim, K. (2004) "Privacy-preserving data mining on vertically partitioned databases," in Franklin, M.K. (Ed.): *Lecture Notes in Computer Science*, Vol. 3152, Springer-Verlag, pp.528-544.

[12] Evfimievski, A., Srikant, R., Agrawal, R. and Gehrke, J. (2002) "Privacy preserving mining of association rules," *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Alberta, Canada, July, pp.217-228.

[13] Grahne, G. and Zhu, J. (2005) "Fast algorithms for frequent itemset mining using FP-trees," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 10, pp.1347-1362.

[14] Han, J., Pei, J., Yin, Y. and Mao, R. (2004) "Mining frequent pattern without candidate generation: a frequent pattern tree approach," *Data Mining and Knowledge Discovery*, Vol. 8, No. 1, pp.53-87.

[15] Kantarcioglu, M. and Clifton, C. (2004) "Privacy-preserving distributed mining of association rules on horizontally partition data," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 9, pp.1026-1037.

[16] Kantardzic, M. (2002) "Data mining: concepts, models, methods, and algorithms," *John Wiley & Sons, Inc.*, New York.

[17] Li, Y.-C. and Chang, C.-C. (2004) "A new FP-tree algorithm for mining frequent itemsets," in Chi, C.-H. and Lam, K.-Y. (Eds.): *Lecture Notes in Computer Science*, Vol. 3309, Springer-Verlag, pp.266-277.

[18] Lindell, Y. and Pinkas, B. (2002) "Privacy preserving data mining," *Journal of Cryptology*, Vol. 15, No. 3, pp.177-206.

[19] Oliveira, S.R.M. and Zaïane, O.R. (2002) "Privacy preserving frequent itemset mining," *Proceedings of the*

2002 IEEE ICDM Workshop on Privacy, Security and Data Mining, Maebashi City, Japan, December, pp.43-54.
[20] Oliveira, S.R.M. and Zaïane, O.R. (2003) "Algorithms for balancing privacy and knowledge discovery in association rule mining," *Proceedings of 7th International Database Engineering and Applications Symposium*, Hong Kong, China, July, pp.54-63.
[21] Pohlig, S.C. and Hellman, M.E. (1978) "An improved algorithm for computing logarithms over GF(P) and its cryptographic significance," *IEEE Transaction on Information Theory*, Vol. 24, No. 1, pp.106-110.
[22] Park, J.S., Chen, M.-S. and Yu, P.S. (1995) "An effective hash-based algorithm for mining association rules," *Proceedings of the 1995 ACM-SIGMOD International Conference on Management of Data*, San Jose, CA, May, pp.175-186.
[23] Savasere, A., Omiecinski, E. and Navathe, S. (1995) "An efficient algorithm for mining association rules in large databases," *Proceedings of the 21th International Conference on Very Large Data Bases*, Zurich, Switzerland, September, pp.432-444.
[24] Saygin, Y., Verykios, V.S. and Clifton, C. (2001) "Using unknowns to prevent discovery of association rules," *ACM SIGMOD Record*, Vol. 30, No. 4, pp.45-54.
[25] Veloso, A.A., Meira Jr., W., Parthasarathy, S. and de Carvalho, M.B. (2003) "Efficient, accurate and privacy-preserving data mining for frequent itemsets in distributed databases," *Proceedings of the Brazilian Symposium on Databases*, Manaus, Amazonas, Brazil, October, pp.281-292.
[26] Verykios, V.S., Elmagarmid, A.K., Bertino, E., Saygin, Y. and Dasseni, E. (2004) "Association rule hiding," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 4, pp.434-447.
[27] Xiao, M.J., Huang, L.S., Shen, H. and Luo, Y.L. (2005) "Privacy preserving ID3 algorithm over horizontally partitioned data," *Proceedings of the 6th International Conference on Parallel and Distributed Computing, Applications and Technologies*, Dalian, China, December, pp.239-243.

Chin-Chen Chang received his BS degree in applied mathematics in 1977 and the MS degree in computer and decision sciences in 1979, both from the National Tsing Hua University, Hsinchu, Taiwan. He received his Ph.D. in computer engineering in 1982 from the National Chiao Tung University, Hsinchu, Taiwan. During the academic years of 1980-1983, he was on the faculty of the Department of Computer Engineering at the National Chiao Tung University. From 1983-1989, he was on the faculty of the Institute of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan. From August 1989 to July 1992, he was the head of, and a professor in, the Institute of Computer Science and Information Engineering at the National Chung Cheng University, Chiayi, Taiwan. From August 1992 to July 1995, he was the dean of the college of Engineering at the same university. From August 1995 to October 1997, he was the provost at the National Chung Cheng University. From September 1996 to October 1997, Dr. Chang was the Acting President at the National Chung Cheng University. From July 1998 to June 2000, he was the director of Advisory Office of the Ministry of Education of the R.O.C. From 2002 to 2005, he was a Chair Professor of National Chung Cheng University. Since February 2005, he has been a Chair Professor of Feng Chia University. In addition, he has served as a consultant to several research institutes and government departments. His current research interests include database design, computer cryptography, image compression and data structures.

Jieh-Shan Yeh received his bachelor's and mater's degrees in Mathematics from National Taiwan University, Taiwan. He obtained his Ph.D. degree in Mathematics from the Ohio State University, USA. Dr. Yeh is an assistant professor in the Department of Computer Science and Information Management, Providence University, Taiwan, since Sep. 2003. Prior to joining Providence University, Dr. Yeh held positions at Federated Department Stores and the Ohio State University. His research interests include data mining, cryptography, XML, and database systems. He is a member of the IEEE and the ACM.

Yu-Chiang Li received his B.Ed. degree in Mathematics and Science Education from National Pingtung Teachers College, Taiwan, in 1996, and received his M.Ed. degree in Computer Science and Information Education from National Tainan Teachers College, Taiwan, in 2001. He is currently pursuing the Ph.D. degree in Computer Science and Information Engineering at National Chung Cheng University in Taiwan. His research interests include data mining, image processing, OLAP, and computational biology.