Privacy Preserving Association Rule Mining in Vertically Partitioned Data

Jaideep Vaidya
Department of Computer Sciences
Purdue University
West Lafayette, Indiana
jsvaidya@cs.purdue.edu

Chris Clifton
Department of Computer Sciences
Purdue University
West Lafayette, Indiana
clifton@cs.purdue.edu

ABSTRACT

Privacy considerations often constrain data mining projects. This paper addresses the problem of association rule mining where transactions are distributed across sources. Each site holds some attributes of each transaction, and the sites wish to collaborate to identify globally valid association rules. However, the sites must not reveal individual transaction data. We present a two-party algorithm for efficiently discovering frequent itemsets with minimum support levels, without either site revealing individual transaction values.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.2.4 [**Database Management**]: Systems—*Distributed databases*; H.2.7 [**Database Management**]: Database Administration—*Security, integrity, and protection*

1. INTRODUCTION

Data mining technology has emerged as a means for identifying patterns and trends from large quantities of data. Mining encompasses various algorithms such as clustering, classification, association rule mining and sequence detection. Traditionally, all these algorithms have been developed within a centralized model, with all data being gathered into a central site, and algorithms being run against that data.

Privacy concerns can prevent this approach – there may not *be* a central site with authority to see all the data. We present a privacy preserving algorithm to mine association rules from vertically partitioned data. By *vertically partitioned*, we mean that each site contains some elements of a transaction. Using the traditional "market basket" example, one site may contain grocery purchases, while another has clothing purchases. Using a key such as credit card number and date, we can join these to identify relationships between purchases of clothing and groceries. However, this discloses the individual purchases at each site, possibly violating consumer privacy agreements.

There are more realistic examples. In the sub-assembly manufacturing process, different manufacturers provide components of the finished product. Cars incorporate several subcomponents; tires, electrical equipment, etc.; made by independent producers. Again, we have proprietary data collected by several parties, with a single key joining all the data sets, where mining would help detect/predict malfunctions. The recent trouble between Ford Motor and Firestone Tire provide a real-life example. Ford Explorers with Firestone tires from a *specific* factory had tread separation problems in certain situations, resulting in 800 injuries. Since the tires did not have problems on other vehicles, and other tires on Ford Explorers did not pose a problem, neither side felt responsible. The delay in identifying the real problem led to a public relations nightmare and the eventual replacement of 14.1 million tires[16]. Many of these were probably fine – Ford Explorers accounted for only 6.5 million of the replaced tires [11]. Both manufacturers had their own data – early generation of association rules based on *all* of the data may have enabled Ford and Firestone to resolve the safety problem before it became a public relations nightmare.

Informally, the problem is to mine association rules across two databases, where the columns in the table are at different sites, splitting each row. One database is designated the *primary*, and is the initiator of the protocol. The other database is the *responder*. There is a *join key* present in both databases. The remaining attributes are present in one database or the other, but not both. The goal is to find association rules involving attributes other than the join key.

We must also lay out the privacy constraints. Ideally we would achieve complete zero knowledge, but for a practical solution controlled information disclosure may be acceptable. Finally, we need to quantify the accuracy and the efficiency of the algorithm, in view of the security restrictions.

We first outline related work and the problem background. In Section 3, we formally define the problem and give the overall solution. Section 4 presents the component scalar product protocol, the key *privacy-preserving* part of the solution. We discuss what the method discloses in Section 5, and present extensions that further limit disclosure. Section 6 analyzes the security provided and communication costs of the algorithm. We conclude by summarizing the contributions of this paper and giving directions for future work.

SIGKDD '02 Edmonton, Alberta, Canada

2. BACKGROUND AND RELATED WORK

The centralized data mining model assumes that all the data required by any data mining algorithm is either available at or can be sent to a central site. A simple approach to data mining over multiple sources that will not share data is to run existing data mining tools at each site independently and combine the results[5, 6, 17]. However, this will often fail to give globally valid results. Issues that cause a disparity between local and global results include:

- Values for a single entity may be split across sources. Data mining at individual sites will be unable to detect cross-site correlations.
- The same item may be duplicated at different sites, and will be over-weighted in the results.
- Data at a single site is likely to be from a homogeneous population, hiding geographic or demographic distinctions between that population and others.

Algorithms have been proposed for distributed data mining. Cheung et al. proposed a method for horizontally partitioned data[8], and more recent work has addressed privacy in this model[14]. Distributed classification has also been addressed. A meta-learning approach has been developed that uses classifiers trained at different sites to develop a global classifier [6, 17]. This *could* protect the individual entities, but it remains to be shown that the individual classifiers do not disclose private information. Recent work has addressed classification using Bayesian Networks in vertically partitioned data [7], and situations where the distribution is itself interesting with respect to what is learned [19]. However, none of this work addresses *privacy* concerns.

There has been research considering how much information can be inferred, calculated or revealed from the data made available through data mining algorithms, and how to minimize the leakage of information [15, 4]. However, this has been restricted to classification, and the problem has been treated with an "all or nothing" approach. We desire quantification of the security of the process. Corporations may not require absolute zero knowledge protocols (that leak no information at all) as long as they can keep the information shared within bounds.

In [4], data perturbation techniques are used to protect individual privacy for classification, by adding random values from a normal/Gaussian distribution of mean 0 to the actual data values). One problem with this approach is a tradeoff between privacy and the accuracy of the results[1]. More recently, data perturbation has been applied to boolean association rules [18]. One interesting feature of this work is a flexible definition of privacy; e.g., the ability to correctly guess a value of '1' from the perturbed data can be considered a greater threat to privacy than correctly learning a '0'. [15] use cryptographic protocols to achieve complete zero knowledge leakage for the ID3 classification algorithm for two parties with horizontally partitioned data.

There has been work in cooperative computation between entities that mutually distrust one another. Secure two party computation was first investigated by Yao [20], and later generalized to multiparty computation. The seminal paper by Goldreich proves existence of a secure solution for *any* functionality[12]. The idea is as follows: the function F to be computed is first represented as a combinatorial circuit, and then the parties run a short protocol to securely compute every gate in the circuit. Every participant gets corresponding shares of the input wires and the output wires for every gate. This approach, though appealing in its generality and simplicity, means that the size of the protocol depends on the size of the circuit, which depends on the size of the input. This is inefficient for large inputs, as in data mining. In [9], relationships have been drawn between several problems in Data Mining and Secure Multiparty Computation. Although this shows that secure solutions exist, achieving *efficient* secure solutions for privacy preserving distributed data mining is still open.

3. PROBLEM DEFINITION

We consider the heterogeneous database scenario considered in [7], a vertical partitioning of the database between two parties A and B. The association rule mining problem can be formally stated as follows[2]: Let $\mathcal{I} = \{i_1, i_2, \cdots, i_m\}$ be a set of literals, called items. Let $\mathcal{D}$ be a set of transactions, where each transaction T is a set of items such that $T \subseteq \mathcal{I}$. Associated with each transaction is a unique identifier, called its *TID*. We say that a transaction T *contains* X, a set of some items in $\mathcal{I}$, if $X \subseteq T$. An *association rule* is an implication of the form, $X \Rightarrow Y$, where $X \subset \mathcal{I}$, $Y \subset \mathcal{I}$, and $X \cap Y = \phi$. The rule $X \Rightarrow Y$ holds in the transaction set $\mathcal{D}$ with *confidence* c if c% of transactions in $\mathcal{D}$ that contain X also contain Y. The rule $X \Rightarrow Y$ has *support* s in the transaction set $\mathcal{D}$ if s% of transactions in $\mathcal{D}$ contain $X \cup Y$.

Within this framework, we consider mining boolean association rules. The absence or presence of an attribute is represented as a 0 or 1. Transactions are strings of 0 and 1.

To find out if a particular itemset is frequent, we count the number of records where the values for all the attributes in the itemset are 1. This translates into a simple mathematical problem, given the following definitions:

Let the total number of attributes be $l + m$, where A has l attributes A_1 through A_l, and B has the remaining m attributes B_1 through B_m. Transactions/records are a sequence of $l + m$ 1s or 0s. Let k be the support threshold required, and n be the total number of transaction/records.

Let $\vec{X}$ and $\vec{Y}$ represent columns in the database, i.e., $x_i =$ 1 iff row i has value 1 for attribute X. The scalar (or dot) product of two cardinality n vectors $\vec{X}$ and $\vec{Y}$ is defined as:

$$\vec{X} \cdot \vec{Y} = \sum_{i=1}^{n} x_i * y_i$$

Determining if the two-itemset $\langle XY \rangle$ is frequent thus reduces to testing if $\vec{X} \cdot \vec{Y} \geq k$.

In Section 4 we present an efficient way to compute scalar product $\vec{X} \cdot \vec{Y}$ without either side disclosing its vector. First we will show how to generalize the above protocol from two-itemsets to general association rules without sharing information other than through scalar product computation.

Generalizing this protocol to a w-itemset is straightforward. Assume A has p attributes $a_1 \ldots a_p$ and B has q attributes $b_1 \ldots b_q$, and we want to compute the frequency of the $w = p + q$-itemset $\langle a_1, \ldots, a_p, b_1, \ldots, b_q \rangle$. Each item in $\vec{X}$ ($\vec{Y}$) is composed of the product of the corresponding individual elements, i.e., $x_i = \prod_{j=1}^{p} a_j$ and $y_i = \prod_{j=1}^{q} b_j$. This computes $\vec{X}$ and $\vec{Y}$ without sharing information between A and B. The scalar product protocol then securely computes the frequency of the entire w-itemset.

For example, suppose we want to compute if a particular 5-itemset is frequent, with A having 2 of the attributes, and B having the remaining 3 attributes. I.e., A and B want to know if the itemset $l = \langle A_a, A_b, B_a, B_b, B_c \rangle$ is frequent. A creates a new vector $\vec{X}$ of cardinality n where $\vec{X} = \vec{A_a} * \vec{A_b}$ (component multiplication) and B creates a new vector $\vec{Y}$ of cardinality n where $\vec{Y} = \vec{B_a} * \vec{B_b} * \vec{B_c}$. Now the scalar product of $\vec{X}$ and $\vec{Y}$ provides the (in)frequency of the itemset.

The complete algorithm to find frequent itemsets is:

1. L_1 = {large 1-itemsets}
2. for (k=2; $L_{k-1} \neq \phi$; k++) do begin
3. C_k = apriori-gen(L_{k-1});
4. for all candidates $c \in C_k$ do begin
5. if all the attributes in c are entirely at A or B
6. that party independently calculates *c.count*
7. else
8. let A have l of the attributes and B have the remaining m attributes
9. construct $\vec{X}$ on A's side and $\vec{Y}$ on B's side where $\vec{X} = \prod_{i=1}^{l} \vec{A_i}$ and $\vec{Y} = \prod_{i=1}^{m} \vec{B_i}$
10. compute $c.count = \vec{X} \cdot \vec{Y} = \sum_{i=1}^{n} x_i * y_i$
11. endif
12. $L_k = L_k \cup c | c.count \geq minsup$
13. end
14. end
15. Answer = $\cup_k L_k$

In step 3, the function apriori-gen takes the set of large itemsets L_{k-1} found in the $(k-1)th$ pass as an argument and generates the set of candidate itemsets C_k. This is done by generating a superset of possible candidate itemsets and pruning this set. [3] discusses the function in detail.

Given the counts and frequent itemsets, we can compute all association rules with $support \geq minsup$.

Only the steps 1, 3, 10 and 12 require sharing information. Since the final result $\cup_k L_k$ is known to both parties, steps 1, 3 and 12 reveal no extra information to either party. We now show how to compute step 10 without revealing information.

4. THE COMPONENT ALGORITHM

Secure computation of scalar product is the key to our protocol. Scalar product protocols have been proposed in the Secure Multiparty Computation literature[10], however these cryptographic solutions do not scale well to this data mining problem. We give an algebraic solution that hides true values by placing them in equations masked with random values. The knowledge disclosed by these equations only allows computation of private values if one side learns a substantial number of the private values from an outside source. (A different algebraic technique has recently been proposed [13], however it requires at least twice the bitwise communication cost of the method presented here.)

We assume without loss of generality that n is even.

Step 1: A generates randoms $R_1 \ldots R_n$. From these, $\vec{X}$, and a matrix C forming coefficients for a set of linear independent equations, A sends the following vector $\vec{X'}$ to B:

$$\langle x_1 + c_{1,1} * R_1 + c_{1,2} * R_2 + \cdots + c_{1,n} * R_n \rangle$$
$$\langle x_2 + c_{2,1} * R_1 + c_{2,2} * R_2 + \cdots + c_{2,n} * R_n \rangle$$
$$\vdots$$
$$\langle x_n + c_{n,1} * R_1 + c_{n,2} * R_2 + \cdots + c_{n,n} * R_n \rangle$$

In step 2, B computes $S = \vec{X'} \cdot \vec{Y}$. B also calculates the following n values:

$$\langle c_{1,1} * y_1 + c_{2,1} * y_2 + \cdots + c_{n,1} * y_n \rangle$$
$$\langle c_{1,2} * y_1 + c_{2,2} * y_2 + \cdots + c_{n,2} * y_n \rangle$$
$$\vdots$$
$$\langle c_{1,n} * y_1 + c_{2,n} * y_2 + \cdots + c_{n,n} * y_n \rangle$$

But B can't send these values, since A would then have n independent equations in n unknowns $(y_1 \ldots y_n)$, revealing the y values. Instead, B generates r random values, $R'_1 \ldots R'_r$. The number of values A would need to know to obtain full disclosure of B's values is governed by r.

B partitions the n values created earlier into r sets, and the R' values are used to hide the equations as follows:

$$\langle c_{1,1} * y_1 + c_{2,1} * y_2 + \cdots + c_{n,1} * y_n + R'_1 \rangle$$
$$\vdots$$
$$\langle c_{1,n/r} * y_1 + c_{2,n/r} * y_2 + \cdots + c_{n,n/r} * y_n + R'_1 \rangle$$
$$\langle c_{1,(n/r+1)} * y_1 + c_{2,(n/r+1)} * y_2 + \cdots + c_{n,(n/r+1)} * y_n + R'_2 \rangle$$
$$\vdots$$
$$\langle c_{1,2n/r} * y_1 + c_{2,2n/r} * y_2 + \cdots + c_{n,2n/r} * y_n + R'_2 \rangle$$
$$\vdots$$
$$\langle c_{1,((r-1)n/r+1)} * y_1 + c_{2,((r-1)n/r+1)} * y_2 + \cdots + c_{n,((r-1)n/r+1)} * y_n + R'_r \rangle$$
$$\vdots$$
$$\langle c_{1,n} * y_1 + c_{2,n} * y_2 + \cdots + c_{n,n} * y_n + R'_r \rangle$$

Then B sends S and the n above values to A, who writes:

$$\begin{aligned} S = {} & (x_1 + c_{1,1} * R_1 + c_{1,2} * R_2 + \cdots + c_{1,n} * R_n) * y_1 \\ & + (x_2 + c_{2,1} * R_1 + c_{2,2} * R_2 + \cdots + c_{2,n} * R_n) * y_2 \\ & \vdots \\ & + (x_n + c_{n,1} * R_1 + c_{n,2} * R_2 + \cdots + c_{n,n} * R_n) * y_n \end{aligned}$$

Simplifying further and grouping the $x_i * y_i$ terms gives:

$$\begin{aligned} S = {} & (x_1 * y_1 + x_2 * y_2 + \cdots + x_n * y_n) \\ & + (y_1 * c_{1,1} * R_1 + y_1 * c_{1,2} * R_2 + \cdots + y_1 * c_{1,n} * R_n) \\ & + (y_2 * c_{2,1} * R_1 + y_2 * c_{2,2} * R_2 + \cdots + y_2 * c_{2,n} * R_n) \\ & \vdots \\ & + (y_n * c_{n,1} * R_1 + y_n * c_{n,2} * R_2 + \cdots + y_n * c_{n,n} * R_n) \end{aligned}$$

The first line of the R.H.S. can be succinctly written as $\sum_{i=1}^{n} x_i * y_i$, the desired final result. In the remaining portion, we group all multiplicative components vertically, and rearrange the equation to factor out all the R_i values, giving:

$$\begin{aligned} S = {} & \sum_{i=1}^{n} x_i * y_i \\ & + R_1 * (c_{1,1} * y_1 + c_{2,1} * y_2 + \cdots + c_{n,1} * y_n) \\ & + R_2 * (c_{1,2} * y_1 + c_{2,2} * y_2 + \cdots + c_{n,2} * y_n) \\ & \vdots \\ & + R_n * (c_{1,n} * y_1 + c_{2,n} * y_2 + \cdots + c_{n,n} * y_n) \end{aligned}$$

Adding and subtracting the same quantity from one side of the equation does not change the equation in any way.

Hence, the above equation can be rewritten as follows:

$$
\begin{aligned}
S = & \sum_{i=1}^{n} x_i * y_i \\
& + \{R_1 * (c_{1,1} * y_1 + c_{2,1} * y_2 + \cdots + c_{n,1} * y_n) \\
& \quad + R_1 * R'_1 - R_1 * R'_1\} \\
& \vdots \\
& + \{R_{n/r} * (c_{1,n/r} * y_{n/r} + c_{2,n/r} * y_2 + \cdots + c_{n,n/r} * y_n) \\
& \quad + R_{n/r} * R'_1 - R_{n/r} * R'_1\} \\
& + \{R_{n/r+1} * (c_{1,n/r+1} * y_{n/r+1} + c_{2,n/r+1} * y_2 + \\
& \qquad \cdots + c_{n,n/r+1} * y_n) \\
& \quad + R_{n/r+1} * R'_2 - R_{n/r+1} * R'_2\} \\
& \vdots \\
& + \{R_{2n/r} * (c_{1,2n/r} * y_{2n/r} + c_{2,2n/r} * y_2 + \\
& \qquad \cdots + c_{n,2n/r} * y_n) \\
& \quad + R_{2n/r} * R'_2 - R_{2n/r} * R'_2\} \\
& \vdots \\
& \vdots \\
& + \{R_{(r-1)n/r+1} * (c_{1,(r-1)n/r+1} * y_{(r-1)n/r+1} + \\
& \qquad c_{2,(r-1)n/r+1} * y_2 + \cdots + c_{n,(r-1)n/r+1} * y_n) \\
& \quad + R_{(r-1)n/r+1} * R'_r - R_{(r-1)n/r+1} * R'_r\} \\
& \vdots \\
& + \{R_n * (c_{1,n} * y_1 + c_{2,n} * y_2 + \cdots + c_{n,n} * y_n) \\
& \quad + R_n * R'_r - R_n * R'_r\}
\end{aligned}
$$

Now A factors out the R_i from the first two components and groups the rest vertically, giving:

$$
\begin{aligned}
S = & \sum_{i=1}^{n} x_i * y_i \\
& + R_1 * (c_{1,1} * y_1 + c_{2,1} * y_2 + \cdots + c_{n,1} * y_n + R'_1) \\
& \vdots \\
& + R_{n/r} * (c_{1,n/r} * y_{n/r} + c_{2,n/r} * y_2 + \\
& \qquad \cdots + c_{n,n/r} * y_n + R'_1) \\
& + R_{n/r+1} * (c_{1,n/r+1} * y_{n/r+1} + c_{2,n/r+1} * y_2 + \\
& \qquad \cdots + c_{n,n/r+1} * y_n + R'_2) \\
& \vdots \\
& + R_{2n/r} * (c_{1,2n/r} * y_{2n/r} + c_{2,2n/r} * y_2 + \\
& \qquad \cdots + c_{n,2n/r} * y_n + R'_2) \\
& \vdots \\
& + R_{(r-1)n/r+1} * (c_{1,(r-1)n/r+1} * y_{(r-1)n/r+1} + \\
& \qquad c_{2,(r-1)n/r+1} * y_2 + \cdots + c_{n,(r-1)n/r+1} * y_n + R'_r) \\
& \vdots \\
& + R_n * (c_{1,n} * y_1 + c_{2,n} * y_2 + \cdots + c_{n,n} * y_n + R'_r) \\
& - R_1 * R'_1 - \cdots - R_{n/r} * R'_1 \\
& - R_{n/r+1} * R'_2 - \cdots - R_{2n/r} * R'_2 \\
& \vdots \\
& - R_{(r-1)n/r+1} * R'_r - \cdots - R_n * R'_r
\end{aligned}
$$

A already knows the n R_i values. B also sent n other values, these are the coefficients of the n R_i values above.

A multiplies the n values received from B with the corresponding R_i and subtracts the sum from S to get:

$$
\begin{aligned}
Temp = & \sum_{i=1}^{n} x_i * y_i \\
& - R_1 * R'_1 - \cdots - R_{n/r} * R'_1 \\
& - R_{n/r+1} * R'_2 - \cdots - R_{2n/r} * R'_2 \\
& \vdots \\
& - R_{(r-1)n/r+1} * R'_r - \cdots - R_n * R'_r
\end{aligned}
$$

Factoring out the R'_i gives:

$$
\begin{aligned}
Temp = & \\
& \sum_{i=1}^{n} x_i * y_i \\
& - (R_1 + R_2 + \cdots + R_{n/r}) * R'_1 \\
& - (R_{n/r+1} + R_{n/r+2} + \cdots + R_{2n/r}) * R'_2 \\
& \vdots \\
& - (R_{((r-1)n/r)+1} + R_{((r-1)n/r)+2} + \cdots + R_n) * R'_r
\end{aligned}
$$

To get the desired final result (viz. $\sum_{i=1}^{n} x_i * y_i$), A needs to add the sum of the r multiplicative terms to Temp.

In step 3, A sends the r values to B, and B (knowing R') computes the final result. Finally B replies with the result.

4.1 Selection of $c_{i,j}$

The above protocol requires a matrix C of values that form coefficients of linear independent equations. The necessity of this is obvious from the fact that the equations are used to hide the data values. If any equation can be eliminated using less than half of the other equations, a linkage between less than $n/2$ of the unknowns is created.

With high probability, a coefficient matrix generated by a pseudo-random function will form linearly independent equations. This enables construction of the $c_{i,j}$ matrix by sharing only a seed and a generating function.

5. WHAT THIS METHOD DISCLOSES

The goal of this work is to create a practical, efficient method to compute association rules without disclosing entity values. This does *not* require a complete zero-knowledge solution. We will now discuss what is disclosed, and problems posed by boolean attributes. Other issues, such as preventing disclosure of what items exist at each site, or

preventing a "cheater" from probing to find a specific value (e.g., by setting all values but one to 0 - a problem for true zero-knowledge scalar product protocols, but detectable and thus preventable with our method) are omitted due to space constraints.

5.1 What Must be Disclosed

The nature of the problem – each party knows its own data, and learns the resulting global association rules – results in some disclosure. For example, if we have a support threshold of 5%, a rule $A_1 \rightarrow B_1$ holds, and exactly 5% of the items on A have item A_1, A knows that at least those same items on B have property B_1. It is acceptable for the protocol to disclose knowledge that could be obtained from the global rules and one's own database.

This method discloses more than just the presence or absence of a rule with support above a threshold. Side A learns the exact support for an itemset. This increases the probability that A will learn that a set of items on B have a given property. This occurs when the global support equals A's support, not just when A's support is at the threshold. In any case, A learns the probability that an item in the set supported by A has a property in B, computed as the ratio of the actual support to A's support.

It is unlikely that specific individual data values will be disclosed with certainty by this method. This possibility can be reduced further by allowing approximate answers.

5.2 The Trouble with $\{0, 1\}$

When we mine boolean association rules, the input values (x_i and y_i values) are restricted to 0 or 1. This creates a disclosure risk, both with our protocol and with other scalar product protocols [10, 13]. Recall that A provides $n + r$ equations in $2n$ unknowns. From these B can get r equations in only in the x_i values. If the equations have a unique solution, B could try all possible values of 0 and 1 for all the y_i to obtain the correct solution. This 2^n brute force approach enables B to *know* the correct values for the y_i. An analogous situation exists for B since A can get $n-r$ equations in only the y_i values.

One way to eliminate this problem is to ensure there are multiple solutions to the equations, by cleverly selecting the $c_{i,j}$ values such that it is not evident exactly which of the x_i/y_i are 1's. One way to accomplish that is as follows. Consider the form of any equation sent by B: $c_{1,1} * y_1 + c_{2,1} * y_2 + \cdots + c_{n,1} * y_n + R'_1$. B can group the y_i into pairs of 0 and 1 and selectively set the coefficient of some pairs to be the same. Thus, even if A finds a solution to the equation, it is not unique and does not disclose *which* of the y_i is 0 and which is 1. The duplicated $c_{i,j}$ values can also be used to force multiple solutions to the equations sent by A. Thus B is unable to verify that a given set of x_i by checking if they allow a unique solution to the R_i values.

This requires that B communicate which $c_{i,j}$ should be the same. Alternatively, B can specify a periodic generator and communicate the order of the x/y values so pairs match the period. This increases communication cost by $O(n)$. Also, this creates a problem in security for A. As a result of duplicating half of the $c_{i,j}$, the security of A's x values decrease; if B knows an x value, B can determine the value for the corresponding x of the pair.

Table 1: Security Analysis of Protocol

	Protected values	*Number of randoms generated*	*Total number of unknowns*	*Number of equations revealed*
A	$x_1 \cdots x_n$	n	$2n$	$n + r$
B	$y_1 \cdots y_n$	r	$n + r$	n

Table 2: Communication Cost

Rounds	*Bitwise cost*	
4	$2 * n * MaxValSz$	$O(n)$

*MaxValSz = Maximum bits to represent any input value

6. SECURITY / COMMUNICATION ANALYSIS

6.1 Security Analysis

The security of the scalar product protocol is based on the inability of either side to solve k equations in more than k unknowns. Some of the unknowns are randomly chosen, and can safely be assumed as private. However, if enough data values are known to the other party, the equations can be solved to reveal all values. Therefore, the disclosure risk in this method is based on the number of data values that the other party might know from some external source. Table 1 presents the number of unknowns and equations generated. This shows the number of data values the other party must have knowledge of to obtain full disclosure.

The scalar product protocol is used once for every candidate item set. This could introduce extra equations. When the candidate itemset contains multiple attributes from each side, there is no question of linear equations so it does not perceptibly weaken the privacy of the data.

It is also possible to have multiple w-itemsets in the candidate set split as 1, $w-1$ on each side. Consider two possible candidate sets A_1, B_1, B_2, B_5 and A_1, B_2, B_3. If A uses new/different equations for each candidate set, it imperils the security of A_1. However, B can reuse the values sent the first time. The equations sent by B can be reused for the same combinations of B_i, only a new sum must be sent. This reveals an additional equation, limiting the number of times B can run the protocol. This limit is adjusted by r, and is unlikely to be a reached if the number of entities is high relative to the number of attributes.

6.2 Communication Analysis

The total communication cost depends on the number of candidate itemsets, and can best be expressed as a (constant) multiple of the i/o cost of the apriori algorithm. Computing support for each candidate itemset requires one run of the component scalar product protocol. The cost of each run (based on the number of items n is as follows: A sends one message with n values. B replies with a message consisting of $n + 1$ values. A then sends a message consisting of r values. Finally B sends the result, for a total of four communication rounds. The bitwise communication cost is O(n) with constant approximately 2 (assuming r is constant). This is summarized in Table 2.

There is also the quadratic cost of communicating the $c_{i,j}$ values. However, this cost can made constant by agreeing

on a function and a seed value to generate the values.

7. CONCLUSION AND FUTURE WORK

The major contributions of this paper are a privacy preserving association rule mining algorithm given a privacy preserving scalar product protocol, and an efficient protocol for computing scalar product while preserving privacy of the individual values. We show that it is possible to achieve good individual security with communication cost comparable to that required to build a centralized data warehouse.

There are several directions for future research. Handling multiple parties is a non-trivial extension, especially if we consider collusion between parties as well. This work is limited to boolean association rule mining. Non-categorical attributes and quantitative association rule mining are significantly more complex problems.

The same privacy issues face other types of data mining, such as Clustering, Classification, and Sequence Detection. Our grand goal is to develop methods enabling *any* data mining that can be done at a single site to be done across various sources, while respecting their privacy policies.

8. ACKNOWLEDGMENTS

The authors thank Murat Kantarcioglu for suggestions on addressing boolean attributes. Portions of this work were supported by Grant EIA-9903545 from the National Science Foundation, and by sponsors of the Center for Education and Research in Information Assurance and Security.

9. REFERENCES

[1] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Santa Barbara, California, USA, May 21-23 2001. ACM.

[2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., May 26–28 1993.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases*, Santiago, Chile, Sept. 12-15 1994. VLDB.

[4] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD Conference on Management of Data*, Dallas, TX, May 14-19 2000. ACM.

[5] P. Chan. *An Extensible Meta-Learning Approach for Scalable and Accurate Inductive Learning*. PhD thesis, Department of Computer Science, Columbia University, New York, NY, 1996. (Technical Report CUCS-044-96).

[6] P. Chan. On the accuracy of meta-learning for scalable data mining. *Journal of Intelligent Information Systems*, 8:5–28, 1997.

[7] R. Chen, K. Sivakumar, and H. Kargupta. Distributed web mining using bayesian networks from multiple data streams. In *The 2001 IEEE International Conference on Data Mining*. IEEE, Nov. 29 - Dec. 2 2001.

[8] D. W.-L. Cheung, V. Ng, A. W.-C. Fu, and Y. Fu. Efficient mining of association rules in distributed databases. *Transactions on Knowledge and Data Engineering*, 8(6):911–922, Dec. 1996.

[9] W. Du and M. J. Atallah. Secure multi-party computation problems and their applications: A review and open problems. In *Proceedings of the 2001 New Security Paradigms Workshop*, Cloudcroft, New Mexico, Sept. 11-13 2001.

[10] W. Du and M. J. Atallah. Secure multi-party computational geometry. In *Proceedings of the Seventh International Workshop on Algorithms and Data Structures*, Providence, Rhode Island, Aug. 8-10 2001.

[11] Ford Motor Corporation. Corporate citizenship report. http://www.ford.com/en/ourCompany/communityAndCulture/buildingRelationships/strategicIssues/firestoneTireRecall.htm, May 2001.

[12] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game - a completeness theorem for protocols with honest majority. In *19th ACM Symposium on the Theory of Computing*, pages 218–229, 1987.

[13] I. Ioannidis, A. Grama, and M. Atallah. A secure protocol for computing dot products in clustered and distributed environments. In *The International Conference on Parallel Processing*, Vancouver, Canada, Aug. 18-21 2002.

[14] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *The ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'02)*, June 2 2002.

[15] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology – CRYPTO 2000*, pages 36–54. Springer-Verlag, Aug. 20-24 2000.

[16] National Highway Traffic Safety Administration. Firestone tire recall. http://www.nhtsa.dot.gov/hot/Firestone/Index.html, May 2001.

[17] A. Prodromidis, P. Chan, and S. Stolfo. *Meta-learning in distributed data mining systems: Issues and approaches*, chapter 3. AAAI/MIT Press, 2000.

[18] S. J. Rizvi and J. R. Haritsa. Privacy-preserving association rule mining. In *Proceedings of 28th International Conference on Very Large Data Bases*. VLDB, Aug. 20-23 2002.

[19] R. Wirth, M. Borth, and J. Hipp. When distribution is part of the semantics: A new problem class for distributed knowledge discovery. In *Ubiquitous Data Mining for Mobile and Distributed Environments workshop associated with the Joint 12th European Conference on Machine Learning (ECML'01) and 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'01)*, Freiburg, Germany, Sept.3-7 2001.

[20] A. C. Yao. How to generate and exchange secrets. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*, pages 162–167. IEEE, 1986.