# Distributed Data Mining Protocols for Privacy: A Review of Some Recent Results*

Rebecca N. Wright[1] and Zhiqiang Yang[1] and Sheng Zhong[2]**

[1] Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ 07030 USA
[2] Department of Computer Science & Engineering, State University of New York at Buffalo, Buffalo, NY 14260 USA

**Abstract.** With the rapid advance of the Internet, a large amount of sensitive data is collected, stored, and processed by different parties. Data mining is a powerful tool that can extract knowledge from large amounts of data. Generally, data mining requires that data be collected into a central site. However, privacy concerns may prevent different parties from sharing their data with others. Cryptography provides extremely powerful tools which enable data sharing while protecting data privacy.
In this paper, we briefly survey four recently proposed cryptographic techniques for protecting data privacy in distributed settings. First, we describe a privacy-preserving technique for learning Bayesian networks from a dataset vertically partitioned between two parties. Then, we describe three privacy-preserving data mining techniques in a fully distributed setting where each customer holds a single data record of the database.

## 1 Introduction

The advances in networking, data storage, and data processing make it easy to collect data on a large scale. Data, including sensitive data, is generally stored by a number of entities, ranging from individuals and small businesses to national governments. By sensitive data, we mean the data that, if used improperly, can harm data subjects, data owners, data users, or other relevant parties. Data mining provides the power to extract useful knowledge from large amounts of data. However, most data mining techniques need to collect data from different parties; in many situations, privacy concerns may prevent different parties from sharing their data with others. An important technical challenge is how to enable data sharing while protecting data privacy.

Data privacy is an important issue to both individuals and organizations. Loosely speaking, data privacy means the ability to protect selected information against selected parties. More precise definitions of data privacy have been presented in different circumstances. It is still an area of active study to determine

* This work was supported by the National Science Foundation under Grant No. CCR-0331584.

** Work completed while at Stevens Institute of Technology.

To appear in *Proceedings of MADNES'05*, Springer LNCS.

the best definition of data privacy in an environment where many uses are to be enabled (some of which are unknown at the time of initial data processing) and many privacy requirements are to be met (again, some of which are unknown at the time of initial data processing).

Privacy-preserving data mining provides methods that can compute or approximate the output of a data mining algorithm without revealing at least part of the sensitive information about the data. Existing solutions can primarily be categorized into two approaches. One approach adopts cryptographic techniques to provide secure solutions in distributed settings (e.g., [LP02]). Another approach randomizes the original data so that certain underlying patterns, such as the distribution of values, are retained in the randomized data (e.g., [AS00]). Generally, the cryptographic approach can provide solutions with perfect accuracy and perfect privacy. In contrast, the randomization approach is much more efficient than the cryptographic approach, but appears to suffer a tradeoff between privacy and accuracy.

In principle, the elegant and powerful paradigm of secure multiparty computation provides general-purpose cryptographic solutions for any distributed computation [GMW87, Yao86]. However, because the inputs of data mining algorithms are huge, the overheads of the general-purpose solutions are intolerable for most applications. Instead, research in this areas seeks more efficient solutions for specific functions.

Privacy-preserving algorithms have been proposed for different data mining applications, including privacy-preserving collaborative filtering [Can02], decision trees on randomized data [AS00], association rules mining on randomized data [RH02,ESAG02], association rules mining across multiple databases [VC02, KC02], clustering [VC03, JW05, JPW06], and naive Bayes classification [KV03, VC04]. Additionally, several solutions have been proposed for privacy-preserving versions of simple primitives that are very useful for designing privacy-preserving data mining algorithms. These include finding common elements [FNP04,AES03], computing scalar products [CIK$^+$01,AD01,VC02,SWY04,FNP04,GLLM04], and computing correlation matrices [LKR03].

In this paper, we survey four of our recently proposed cryptographic privacy-preserving techniques for data mining in distributed settings [YW06, YZW05b, ZYW05, YZW05a]. Specifically, we consider two different distributed settings. In the first setting, data is distributed between two parties. The challenge is to protect data privacy while enabling the cooperation among those parties. In Section 3, we describe a privacy-preserving solution for the two parties to compute a Bayesian network on their distributed data.

In the second setting, called the *fully distributed setting*, each party holds one record of a virtual database. The fully distributed setting is particularly well suited towards the setting of mobile ad hoc networks because each party retains control of its own information. The parties can decide when they are and are not willing to participate in various data mining tasks. In this setting, we consider the scenario where a data miner wants to carry out data mining applications. The challenge is to enable the miner to learn the results of data mining tasks

while protecting each party's privacy. We describe privacy-preserving solution for three tasks in the fully distributed model in Section 4.

## 2 Privacy Definition in Secure Multiparty Computation

In this work, we define privacy by adapting the general privacy definition in secure multiparty computation [GMW87, Yao86, Gol04]. As usual, we make the distinction between *semi-honest* and *malicious* adversaries in the distributed setting. Semi-honest adversaries only gather information and do not modify the behavior of the parties. Such adversaries often model attacks that take place after the execution of the protocol has completed. Malicious adversaries can cause the corrupted parties to execute some arbitrary, malicious operations. Here, we review the formal privacy definition with respect to semi-honest adversaries [Gol04].

**Definition 1.** *(privacy w.r.t semi-honest behavior) Let* $f : (x_1, \cdots, x_m) \rightarrow (y_1, \cdots, y_m)$ *be an* $m$*-ary function and denote* $(x_1, \cdots, x_m)$ *by* $\overline{x}$*. For* $I = \{i_1, \cdots, i_t\} \subseteq [m] = \{1, \cdots, m\}$*, we let* $f_I(\overline{x})$ *denote* $\overline{y} = \{y_{i_1}, \cdots, y_{i_t}\}$ *and let* $\prod$ *be a* $m$*-party protocol for computing* $f$*. The view of the* $i^{th}$ *party during an execution of* $\prod$ *is denoted by* $\mathsf{view}_i(\overline{x})$ *which includes* $x_i$*, all received messages, and all internal coin flips. For* $I = \{i_1, \cdots, i_t\}$*, we let* $\mathsf{view}_I(\overline{x}) = (\mathsf{view}_{i_1}(\overline{x}), \cdots, \mathsf{view}_{i_t}(\overline{x}))$*. We say that* $\prod$ *privately computes* $F$ *against up to* $t$ *semi-honest adversaries if for all* $I \subseteq \{1, \ldots, m\}$ *(*$|I| = t$*), for all* $\overline{x}$*, there exists a probabilistic polynomial-time algorithm (a simulator), denoted* $S$*, such that*

$$\{S((x_{i_1}, \cdots, x_{i_t}), f(\overline{x}))\} \stackrel{\mathsf{c}}{\equiv} \{(\mathsf{view}_I(\overline{x}), OUTPUT(\overline{x}), \}$$

*where* $OUTPUT(\overline{x})$ *denotes the output of all parties during the execution represented in* $\mathsf{view}_I(\overline{x})$*.*

This definition asserts that the view of the parties in $I$ can be efficiently simulated based solely on their inputs and outputs. In other words, the adversaries cannot learn anything except their inputs and final outputs. The privacy definition related with malicious adversaries can be found in [Gol04]. For two-party computation, privacy can be defined in a way slightly different from the above [Gol04].

## 3 Privacy-Preserving Distributed Data Mining

Cryptographic techniques provide the tools to protect data privacy by exactly allowing the desired information to be shared while concealing everything else about the data. To illustrate how to use cryptographic techniques to design privacy-preserving solutions to enable mining across distributed parties, we describe a privacy-preserving solution for a particular data mining task: learning Bayesian networks on a dataset divided among two parties who want to carry out data mining algorithms on their joint data without sharing their data directly.

### 3.1 Bayesian networks

A Bayesian network (BN) is a graphical model that encodes probabilistic relationships among variables of interest [CH92]. This model can be used for data analysis and is widely used in data mining applications.

Formally, a Bayesian network for a set $V$ of $m$ variables is a pair $(B_{\mathrm{s}}, B_{\mathrm{p}})$. The *network structure* $B_{\mathrm{s}} = (V, E)$ is a directed acyclic graph whose nodes are the set of variables. The *parameters* $B_{\mathrm{p}}$ describe local probability distributions associated with each variable. There are two important issues in using Bayesian networks: (a) Learning Bayesian networks and (b) Bayesian inferences. Learning Bayesian networks includes learning the structure and the corresponding parameters. Bayesian networks can be constructed by expert knowledge, or from a set of data, or by combining those two methods together. Here, we address the problem of privacy-preserving learning of Bayesian networks from a database vertically partitioned between two parties; in vertically partitioned data, one party holds some of the variables and the other party holds the remaining variable.

### 3.2 The BN Learning Protocol

A value $x$ is *secret shared* (or simply *shared*) between two parties if the parties have values (*shares)* such that neither party knows (anything about) $x$, but given both parties' shares of $x$, it is easy to compute $x$. Our protocol for BN learning uses composition of privacy-preserving subprotocols in which all intermediate outputs from one subprotocol that are inputs to the next subprotocol are computed as secret shares. In this way, it can be shown that if each subprotocol is privacy-preserving, then the resulting composition is also privacy-preserving.

Our solution is a modified version of the well known *K2* protocol of Cooper and Herskovitz [CH92]. That protocol uses a score function to determine which edges to add to the network. To modify the protocol to be privacy-preserving, we seek to divide the problem into several smaller subproblems that we know how to solve in a privacy-preserving way. Specifically, noting that only the relative score values are important, we use a new score function $g$ that approximates the relative order of the original score function. This is obtained by taking the logarithm of the original score function and dropping some lower order terms.

As a result, we are able to perform the necessary computations in a privacy-preserving way. We make use of several cryptographic subprotocols, including secure two-party computation (such as the solution of [Yao86], which we apply only on a small number of values, not on something the size of the original database), a privacy-preserving scalar product share protocol (such as the solutions described by [GLLM04]), and a privacy-preserving protocol for computing $x \ln x$ (such as [LP02]). In turn, we show how to use these to compute shares of the parameters $\alpha_{ijk}$ and $\alpha_{ij}$ that are required by the protocol.

Our overall protocol of learning BNs is described as follows. In keeping with cryptographic tradition, we call the two parties engaged in the protocol Alice and Bob.

**Input:** An ordered set of $m$ nodes, an upper bound $u$ on the number of parents for a node, both known to Alice and Bob, and a database $D$ containing $n$ records, vertically partitioned between Alice and Bob.

**Output:** Bayesian network structure $B_{\mathrm{s}}$ (whose nodes are the $m$ input nodes, and whose edges are as defined by the values of $\pi_i$ at the end of the protocol)

As the ordering of variables in $V$, Alice and Bob execute the following steps at each node $v_i$. Initially, each node has no parent. After Alice and Bob run the following steps at each node, each node has $\pi_i$ as its current set of parents.

1. Alice and Bob execute privacy-preserving approximate score protocol to compute the secret shares of $g(i, \pi_i)$ and $g(i, \pi_i \cup \{z\})$ for any possible additional parent $z$ of $v_i$.
2. Alice and Bob execute privacy-preserving score comparison protocol to compute which of those scores in Step 1 is maximum.
3. If $g(i, \pi_i)$ is maximum, Alice and Bob go to the next node $v_{i+1}$ to run from Step 1 until Step 3. If one $z$ generates the maximum score in Step 2, then $z$ is added as the parent of $v_i$ such that $\pi_i = \pi_i \cup \{z\}$ and Alice and Bob go back to Step 1 on the same node $v_i$.
4. Alice and Bob run a secure two-party computation to compute the desired parameter $\alpha_{ijk}/\alpha_{ij}$.

Further details about this protocol can be found in [YW06], where we also show how a privacy-preserving protocol to compute the parameters $B_{\mathrm{p}}$. Experimental results addressing both the efficiency and the accuracy of the structure-learning protocol can be found in [KRWF05].

## 4 Privacy Protection in the Fully Distributed Setting

In this section, we consider the fully distributed setting, in which each party holds its own data record. Together these records make a "virtual database". We assume there is a data miner that wants to learn some information about this virtual database. We call each of the data-holding parties "respondents".

First, let us consider a typical scenario of mining in the fully distributed setting: the miner queries large sets of respondents, and each respondent submits her data to the miner in response. Clearly, this can be an efficient and convenient procedure, assuming the respondents are willing to submit their data. However, the respondents' willingness to submit data is affected by their privacy concerns [Cra99]. Furthermore, once a respondent submits her data to the miner, the privacy of her data is fully dependent on the miner. Because the miner is interested in obtaining a good and accurate response rate, the protection of respondents' privacy is therefore important to both the success of data mining and the respondents. By using cryptographic techniques, we describe three techniques for different mining or data collection tasks in the fully distributed setting.

### 4.1 Privacy-Preserving Learning Classification Model

In this section, we provide a privacy-preserving protocol to enable a data miner to learn certain classification models without collecting respondents' raw data such as to protect respondents' privacy in the fully distributed setting.

To solve this problem, we propose a simple efficient cryptographic approach which provides strong privacy for each respondent and does not give up any accuracy as the cost of privacy. The critical technique is a frequency-learning protocol that allows a data miner to compute frequencies of values or tuples of values in the respondents' data without revealing the privacy-sensitive part of the data. Unlike general-purpose cryptographic protocols, this method requires no interaction between respondents, and each respondent only needs to send a single flow of communication to the data miner. However, we are still able to ensure that nothing about the sensitive data beyond the desired frequencies is revealed to the data miner. We note that this choice of computation can itself be considered a tradeoff between privacy and utility. On one hand, the frequencies have reasonably high utility, as they can be used to enable a number of different data mining computations, but they have less privacy than requiring a different privacy-preserving computation of each kind of data mining computation the miner might later carry out with the frequencies. On the other hand, (except in degenerate cases), the frequencies have less utility than sending the raw data itself, but more privacy.

The protocol design is based on the additively homomorphic property of a variant of ElGamal encryption, which has been used in, e.g., [HS00]. The protocol itself uses the mathematical properties of exponentiation, which allows the miner to combine encrypted results received from the respondents into the desired sums.

Let $G$ be a group where $|G| = q$ and $q$ is a large prime, and let $g$ be a generator of $G$. All computations in this section are carried out in the group $G$. We assume a prior set-up that results in each respondent $U_i$ having two pairs of keys: $(x_i, X_i = g^{x_i}), (y_i, Y_i = g^{y_i})$. Define

$$X = \prod_{i=1}^{n} X_i \tag{1}$$

$$Y = \prod_{i=1}^{n} Y_i \tag{2}$$

The values $x_i$ and $y_i$ are private keys (i.e., each $x_i$ and $y_i$ is known only to respondent $U_i$); $X_i$ and $Y_i$ are public keys (i.e., they can be publicly known). In particular, the protocol requires that all respondents know the values $X$ and $Y$. In addition, each respondent knows the group $G$ and the common generator $g$.

In this protocol, each respondent $U_i$ holds a Boolean value $d_i$, and the miner's goal is to learn $d = \sum_{i=1}^{n} d_i$. The privacy-preserving protocol for the miner to learn the frequency $d$ is shown in Figure 1.

Using the frequency-learning protocol, we can design a privacy-preserving protocol to learn naive Bayes classifiers which are enabled solely by frequency

$U_i \rightarrow$ miner : $m_i = g^{d_i} \cdot X^{y_i}$;
$h_i = Y^{x_i}$.

miner: $r = \prod_{i=1}^{n} \frac{m_i}{h_i}$;
for $d = 1$ to $n$
if $g^d = r$ output $d$.

**Fig. 1.** Privacy-Preserving Protocol for Frequency Mining.

computation. Details about this protocol can be found in [YZW05b]. To test the efficiency of the protocol, we implemented the Bayes classifier learning protocol by using OpenSSL libraries, and we ran a series of experiments in the NetBSD operating system running on an AMD Athlon 2GHz processor with 512M memory, using 512 bit cryptographic keys. Figure 2 studies how the server's (miner's) learning time changes when both the respondent number and the attribute number vary. In this experiment, we fixed the domain size of each non-class attribute to four and the domain size of the class attribute to two.

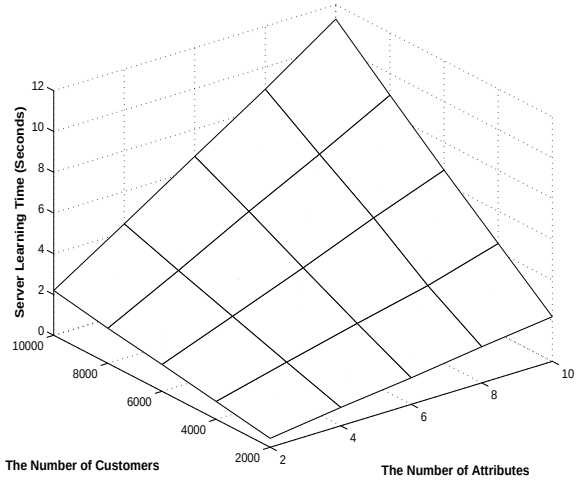


**Fig. 2.** Server's Learning Time for Naive Bayes Classifier vs. Number of respondents and Number of Attributes

### 4.2 Fully Distributed *k*-Anonymization

The frequency-learning protocol can be used to learn only the models which are enabled by frequency computation. However, very often a data miner wants to

collect the respondents' data for the general purpose such that those data can be used for learning any model. An intuitive solution is that each respondent submits their data without any identifiers such that the miner cannot link each respondent with their submitted data and then respondents' privacy can be protected. However, even if the respondents' data do not include explicit identifiable attributes, respondents may often still be identified by using a set of attributes that act as "quasi-identifiers," e.g., {date of birth, zip code}. By using quasi-identifiers, Sweeney [Swe02b] pointed out a privacy attack in which one can find out who has what disease using a public database and voter lists.

*K-anonymization* was first proposed by Samarati and Sweeney [SS98] to address the privacy problem of quasi-identifiers. The basic idea is that a data table is k-anonymized by changing some attributes such that at least $k$ rows have the same quasi-identifier. The existing $k$-anonymization methods work in the centralized setting in which the data table is located in. $K$-anonymization techniques include suppression and generalization methods. Suppression methods substitute the values of some attributes in quasi-identifiers with * but generalization methods substitute the values with more general ones.

$K$-anonymization of data can be viewed as another privacy/utility tradeoff. It publishes data that is not as useful as the original data, but that is intended to be more private. However, existing $k$-anonymization techniques (such as [Swe97, SS98, Sam01, Swe02b, Swe02a, MW04, BA05]) assume that the data is first available in a central location and then modified to produce $k$-anonymous data. In contrast, we add additional privacy protections to the $k$-anonymization process: distributed respondents holding their own data interact with a miner so that the miner learns a $k$-anonymized version of the data but no single participant, including the miner, learns extra information that could be used to link sensitive attributes to corresponding identifiers.

We give two different formulations of this problem:

- In the first formulation, given a table, the protocol needs to extract the *$k$-anonymous part* (i.e., the maximum subset of rows that is already $k$-anonymous) from it. The privacy requirement is that the sensitive attributes outside the $k$-anonymous part should be hidden from any individual respondent including the miner. This formulation is only suitable if the original table is already close to $k$-anonymous, as otherwise the utility of the result will be significantly reduced.
- In the second formulation, given a table, the protocol needs to suppress some entries of the quasi-identifier attributes, so that the entire table is $k$-anonymized. The privacy requirement is that the suppressed entries should be hidden from any individual participant. This formulation is suitable even if the original table is not close to $k$-anonymous.

In [ZYW05], we present efficient solutions to both formulations. Our solutions use cryptography to obtain provable guarantees of their privacy properties, relative to standard cryptographic assumptions. Our solution to the first problem formulation does not reveal any information about the sensitive attributes outside the $k$-anonymous part. Our solution to the second problem formulation

is not fully private, in that it reveals the $k$-anonymous result as well as the distances between each pair of rows in the original table. We prove that it does not reveal any additional information. Our protocols enhance the privacy of $k$-anonymization by maintaining end-to-end privacy from the original data to the final $k$-anonymous results.

### 4.3 Anonymity-Preserving Data Collection

We next consider another task in the fully distributed setting, which can again be considered as different point on the utility/privacy tradeoff. This task is suitable for data collection when the data *is* considered to provide sufficient privacy as long as it can be collected anonymously (i.e., without the data collector learning which data belongs to which respondent). An example of this scenario might be if the miner is a medical researcher who studies the relationship between dining habits and a certain disease. Because a respondent does not want to reveal what food she eats and/or whether she has that disease, she may give false information or decline to provide information. However, even if each respondent's data does not contain any identifiable attribute, the privacy of each respondent cannot be guaranteed because the miner can link the respondent's identity with their submitted data through the communication channel, e.g., by IP address. One possible solution is that the miner collects data *anonymously*. That is, he collects records from the respondents containing each respondent's dining habits and health information related to that disease, but does not know which record came from which respondent. In some settings, this idea that a response is "hidden" among many peers is enough to make participants respond.

We generalize this idea to propose an approach called anonymity-preserving data collection. Specifically, we propose that the miner should collect data in such a way that he is unable to link any piece of data collected to the respondent who provided that piece of data. In this way, respondents do not need to worry about their privacy. Furthermore, the collected data is not modified in any way, and thus the miner will have the freedom to apply any suitable mining algorithms to the data. As discussed above, this is therefore only useful for providing privacy if each respondent's data does not contains identifiable attributes and if the responses themselves do not provide too many clues to the respondent's identity.

We summarize our protocol here. Respondents are divided into many smaller groups of size $N$, in which the respondents' data are denoted by $(d_1, \ldots, d_N)$. A larger $N$ will provide more anonymity but less efficiency, and vice versa. Our goal is that the miner should obtain a random permutation of the respondents' data $(d_1, \ldots, d_N)$, without knowing which piece of data comes from which respondent. To achieve this goal, we use ElGamal encryption together with a *rerandomization* technique and a *joint decryption* technique. In the ElGamal encryption scheme, one cleartext has many possible encryptions, as the random number $r$ can take on many different values. ElGamal supports rerandomization, which means computing a different encryption of $M$ from a given encryption of $M$. A related operation is permutation of the order of items, which means randomly rearranging the order of items. If we rerandomize and permute a sequence of

ciphertexts, then we get another sequence of ciphertexts with the same multiset of cleartexts but in a different order. Looking at these two sequences of ciphertexts, the adversary cannot determine any information about which new ciphertext corresponds to which old ciphertext.

In our solution against semi-honest players including all respondents and the miner, $t$ of the $N$ respondents act as "leaders". Leaders have the special duty of anonymizing the data. At the beginning of the protocol, all respondents encrypt their data using a public key which is the product of all leaders' public keys. Note that the private key corresponding to this public key is the sum of all leaders' private keys; without the help of all leaders, nobody can decrypt any of these encryptions. The leaders then rerandomize these encryptions and permute them. Finally, the leaders jointly help the miner to decrypt the new encryptions, which are in an order independent of the original encryptions. By using digital signature and non-interactive zero-knowledge proofs, we also design the protocols against malicious miner and respondents. Further details can be found in [YZW05a].

To measure the efficiency of our protocols in practice, we implemented them using the OpenSSL libraries and measured the computational overhead. In our experiments, the length of cryptographic keys is 1024 bits. The environment used is the NetBSD operating system running on an AMD Athlon 2GHz processor with 512M memory. In the protocol against semi-honest participants, we measure the computation times of the three types of participants: regular (i.e., non-leader) respondents, leaders, and the miner. A regular respondent's computation time is always about 15ms regardless $N$ and $t$. A leader's computation time is linear in $N$ and does not depend on $t$. For a typical scenario where $N = 20$, the computation time of a leader is about 0.47 seconds. The miner's computation time is linear in both $N$ and $t$. For a typical scenario where $N = 20$ and $t = 3$, the computation time of the miner is about 40ms. In the protocol against the malicious miner, the leader has a 10% increase over the corresponding overhead of the semi-honest protocol. The increased overhead for regular participants and the miner is negligible.

## 5 Discussion

We have described several privacy-preserving protocols. This remains a ripe area for research. We briefly describe some areas worthy of further investigation.

In practice, participants in a privacy-preserving protocol might behave maliciously in order to gain maximum benefits from others. Most existing work on very efficient privacy-preserving data mining, including most of ours, only provides the protocols against semi-honest adversaries. Although in principle those protocols can be modified using a general method to defend against malicious behaviors, the overhead of doing so is intolerable in practice. An important area for future research is the design of efficient mining protocols that remain secure and private even if some of the parties involved behave maliciously.

Because it aims to guarantee strong privacy for all possibilities, the general definition of privacy in secure multi-party computation is very strictly defined. Cryptographic approaches can achieve perfect privacy in principle, but one typically pays a high computational price for such privacy. For specific applications, a relaxed privacy definition might help to design efficient solutions while still be good enough to satisfy practical privacy requirements. Computing approximate mining results rather than the accurate ones might also help get the benefit of efficiency. A particularly interesting question is whether one can identify the quantitative tradeoff among efficiency, privacy, accuracy, and utility, as well as identifying solutions that achieve "good" points in that tradeoff space.

Another interesting question is how to deploy privacy-preserving techniques into practical applications. The techniques of privacy-preserving distributed data mining can be used to learn models across distributed databases. Is it feasible to define a general toolkits which are suitable for all kinds of databases with different data types? Another question is how to implement our methods without introducing covert channels to breach any party's privacy.

Particularly in the fully distributed setting, a question that remains is how to ensure either that participants provide accurate data, or that the miner can produce results in a way that is not heavily dependent on all the data being accurate. Although cryptographic techniques can force each participant to follow the protocol specifications so as to protect data privacy, but they cannot prevent participants from providing faked data to the protocols. Anonymity and privacy remove some disincentive for participants to provide fake data, but it would also be useful to design mechanisms that specifically incent participants to provide their original data.

## References


[AD01] Mikhail Atallah and Wenliang Du. Secure multi-party computational geometry. In *Proc. of the Seventh International Workshop on Algorithms and Data Structures*, pages 165–179. Springer-Verlag, 2001.

[AES03] Rakesh Agrawal, Alexandre Evfimievski, and Ramakrishnan Srikant. Information sharing across private databases. In *Proc. of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 86–97. ACM Press, 2003.

[AS00] Rakesh Agrawal and Ramakrishnan Srikant. Privacy preserving data mining. In *Proc. of the 2000 ACM SIGMOD international conference on Management of data*, pages 439–450. ACM Press, May 2000.

[BA05] Roberto J. Bayardo and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *Proceedings of 21st International Conference on Data Engineering*, 2005.

[Can02] John Canny. Collaborative filtering with privacy. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 45–57, Washington, DC, USA, 2002. IEEE Computer Society.

[CH92] Greg F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Mach. Learn.*, 9(4):309–347, 1992.

[CIK+01] Ran Canetti, Yuval Ishai, Ravi Kumar, Michael K. Reiter, Ronitt Rubinfeld, and Rebecca N. Wright. Selective private function evaluation with applications to private statistics. In *Proc. of the 20th Annual ACM Symposium on Principles of Distributed Computing*, pages 293–304. ACM Press, 2001.

[Cra99] Lorrie F. Cranor. Special issue on internet privacy. *Communications of the ACM*, 42(2), 1999.

[ESAG02] Alexandre Evfimievski, Ramakrishnan Srikant, Rakesh Agrawal, and Johannes Gehrke. Privacy preserving mining of association rules. In *Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 217–228. ACM Press, 2002.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 1–19. Springer-Verlag, 2004.

[GLLM04] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On private scalar product computation for privacy-preserving data mining. In *Proc. of the Seventh Annual International Conference in Information Security and Cryptology*, LNCS. Springer-Verlag, 2004. to appear.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play ANY mental game. In *Proc. of the 19th Annual ACM Conference on Theory of Computing*, pages 218–229. ACM Press, 1987.

[Gol04] Oded Goldreich. *Foundations of Cryptography, Volume II: Basic Applications*. Cambridge University Press, 2004.

[HS00] Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. *Lecture Notes in Computer Science*, 1807:539–556, 2000.

[JPW06] Geetha Jagannathan, Krishnan Pillaipakkamnatt, and Rebecca N. Wright. A new privacy-preserving distributed $k$-clustering algorithm. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, 2006.

[JW05] Geetha Jagannathan and Rebecca N. Wright. Privacy-preserving distributed $k$-means clustering over arbitrarily partitioned data. In *Proc. of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 593–599. ACM Press, 2005.

[KC02] Murat Kantarcioglu and Chris Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *Proc. of the ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'02)*, pages 24–31, June 2002.

[KRWF05] Onur Kardes, Raphael S. Ryger, Rebecca N. Wright, and Joan Feigenbaum. Implementing privacy-preserving Bayesian-net discovery for vertically partitioned data. In *Proceedings of the ICDM Workshop on Privacy and Security Aspects of Data Mining*, Houston, TX, 2005.

[KV03] Murat Kantarcioglu and Jaideep Vaidya. Privacy preserving naive Bayes classifier for horizontally partitioned data. In *IEEE Workshop on Privacy Preserving Data Mining*, 2003.

[LKR03] Kun Liu, Hillol Kargupta, and Jessica Ryan. Multiplicative noise, random projection, and privacy preserving data mining from distributed multi-party data. Technical Report TR-CS-03-24, Computer Science and Electrical Engineering Department, University of Maryland, Baltimore County, 2003.

[LP02] Yehuda Lindell and Benny Pinkas. Privacy preserving data mining. *J. Cryptology*, 15(3):177–206, 2002.

[MW04] Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In *Proc. 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Paris, France, June 2004.

[RH02] Shariq Rizvi and Jayant R. Haritsa. Maintaining data privacy in association rule mining. In *Proc. of the 28th VLDB Conference*, 2002.

[Sam01] Pierangela Samarati. Protecting respondent's privacy in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[SS98] Pierangela Samarati and Latanya Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *Proc. of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, page 188. ACM Press, 1998.

[Swe97] Latanya Sweeney. Guaranteeing anonymity when sharing medical data, the datafly system. *Journal of the American Medical Informatics Association*, 1997.

[Swe02a] Latanya Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness Knowledge-Based Systems*, 10(5):571–588, 2002.

[Swe02b] Latanya Sweeney. k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness Knowledge-Based Systems*, 10(5):557–570, 2002.

[SWY04] Hiranmayee Subramaniam, Rebecca N. Wright, and Zhiqiang Yang. Experimental analysis of privacy-preserving statistics computation. In *Proc. of the VLDB Worshop on Secure Data Management*, pages 55–66, August 2004.

[VC02] Jaideep Vaidya and Chris Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 639–644. ACM Press, 2002.

[VC03] Jaideep Vaidya and Chris Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proc. of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 206–215. ACM Press, 2003.

[VC04] Jaideep Vaidya and Chris Clifton. Privacy preserving naive Bayes classifier on vertically partitioned data. In *2004 SIAM International Conference on Data Mining*, 2004.

[Yao86] Andrew C.-C. Yao. How to generate and exchange secrets. In *Proc. of the 27th IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.

[YW06] Zhiqiang Yang and Rebecca N. Wright. Privacy-preserving Bayesian network computation on vertically partitioned data. In *IEEE Transactions on Knowledge and Data Engineering*, 2006. to appear.

[YZW05a] Zhiqiang Yang, Sheng Zhong, and Rebecca N. Wright. Anonymity-preserving data collection. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.

[YZW05b] Zhiqiang Yang, Sheng Zhong, and Rebecca N. Wright. Privacy-preserving classification of customer data without loss of accuracy. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, 2005.

[ZYW05] Sheng Zhong, Zhiqiang Yang, and Rebecca N. Wright. Privacy-enhancing k-anonymization of customer data. In *Proceedings of the 24th ACM Symposium on Principles of Database Systems*, 2005.