Preserving Data Privacy in Outsourcing Data Aggregation Services

LI XIONG
Emory University
and
SUBRAMANYAM CHITTI and LING LIU
Georgia Institute of Technology

Advances in distributed service-oriented computing and Internet technology have formed a strong technology push for outsourcing and information sharing. There is an increasing need for organizations to share their data across organization boundaries both within the country and with countries that may have lesser privacy and security standards. Ideally, we wish to share certain statistical data and extract the knowledge from the private databases without revealing any additional information of each individual database apart from the aggregate result that is permitted. In this article, we describe two scenarios for outsourcing data aggregation services and present a set of decentralized peer-to-peer protocols for supporting data sharing across multiple private databases while minimizing the data disclosure among individual parties. Our basic protocols include a set of novel probabilistic computation mechanisms for important primitive data aggregation operations across multiple private databases such as max, min, and top k selection. We provide an analytical study of our basic protocols in terms of precision, efficiency, and privacy characteristics. Our advanced protocols implement an efficient algorithm for performing kNN classification across multiple private databases. We provide a set of experiments to evaluate the proposed protocols in terms of their correctness, efficiency, and privacy characteristics.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications

General Terms: Algorithms, Design, Experimentation

Additional Key Words and Phrases: Privacy, confidentiality, classification, outsourcing

ACM Reference Format:

Xiong, L., Chitti, S., and Liu, L. 2007. Preserving data privacy in outsourcing data aggregation services. ACM Trans. Intern. Tech. 7, 3, Article 17 (August 2007), 28 pages. DOI = 10.1145/1275505.1275510 http://doi.acm.org/10.1145/1275505.1275510

This research was partially supported by an Emory UC Grant for L. Xiong and grants from NSF CSR, NSF CyberTrust, NSF ITR, AFOSR, an IBM Faculty Award and an IBM SUR grant for L. Liu.

Authors' addresses: L. Xiong, Emory University, Atlanta, GA; email: lxiong@mathcs.emory.edu; S. Chitti and L. Liu, Georgia Institute of Technology.

1. INTRODUCTION

Modern Internet technology has collapsed geographical boundaries for global information sharing. Outsourcing has been an important and increasing driving force for global information sharing. It involves transferring or sharing management control of a business function to an outside supplier and involves information exchange among the service providers and outsourcing clients. One of the most notable outsourcing services is database outsourcing where organizations outsource the data storage and management to third-party service providers. Traditionally, information integration in both industry and research has assumed that information in each database can be freely shared. It is widely recognized today that data confidentiality and privacy are increasingly becoming an important aspect of data sharing and integration because organizations or individuals do not want to reveal their private databases for various legal and commercial reasons.

The exposure a company risks by not taking additional steps in outsourcing its data processing to third-party vendors can be illustrated by numerous privacy breach incidents both domestically and offshore [Markey 2005]. In February and March 2005, some major data aggregators/resellers, including ChoicePoint Inc. and LexisNexis, allegedly suffered massive fraudulent intrusions into databases that contained personally identifiable information including the social security numbers of thousands of people[1]. Also, in the largest security breach of its kind, 40 million credit card numbers were reported to be at risk of being used for fraud because of a data breach at a processing center in Tucson operated by one of the payment transfer companies[2]. Another incident involved a California hospital that outsourced some data processing work to an international provider and a transcriber in Pakistan threatened to post the hospital's patient information on the Internet if she did not receive a pay raise[3].

The database outsourcing trend along with concerns and laws governing data confidentiality and privacy have led to great interest in enabling secure database services. Previous approaches to enabling such a service have been focused on the data exchange between a single outsourcer and a client and are typically based on data encryption to protect the data privacy for the client. However, Internet technology has enabled data sharing on a global scale that is often among multiple autonomous enterprises across multiple countries. We consider a number of important multiparty distributed outsourcing scenarios in this article and motivate the need for distributed data sharing across multiple private databases.

Multiclient Data Aggregation. Enterprises and organizations have great interest in sharing their data and extracting interesting knowledge. They may outsource the data aggregation service to a third-party data service provider, but, at the same time, they do not want to reveal their private databases for various legal and commercial reasons. Figure 1(a) shows a sketch of the outsourcing scenario. Multiple clients outsource their data aggregation service to

[1] http://blog.washingtonpost.com/securityfix/2006/01/choicepoint _to_pay_15m_for_pri.html.
[2] http://www.washingtonpost.com/wp-dyn/content/article/ 2005/06/17/AR2005061701031.html.
[3] http://www.patientprivacyrights.org/site/PageServer? pagename=True_Stories.

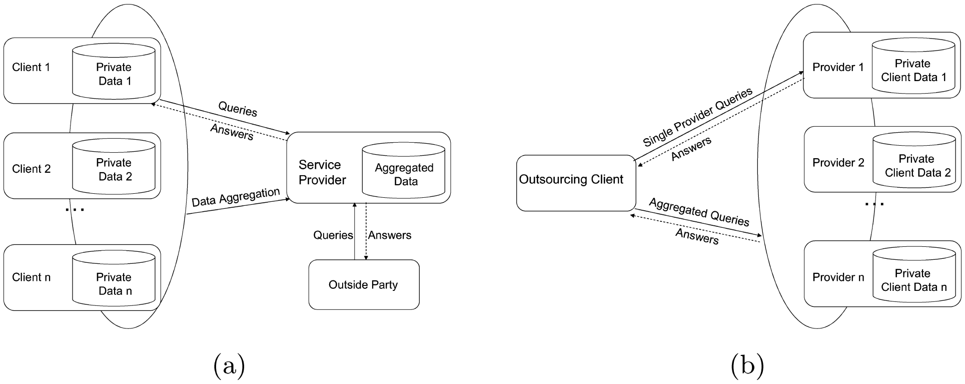


Fig. 1. (a) Multiclient data aggregation outsourcing. (b) Multiprovider outsourcing.

a service provider which will collect and aggregate data from multiple organizations and answer aggregate (statistical) queries from the clients as well as from outside parties. Internet technology plays a key role for enabling this type of outsourcing application where the clients can reside in different geographic locations and data can be accessed through the wide network.

This data aggregation outsourcing scenario is driven by several trends [Agrawal et al. 2003]. In the business world, for example, multiple retailers may wish to share their data through a data aggregation service to study interesting sales patterns and trends of both their own private databases and their competitors' information bases. However, the information flow for such sharing is restricted in two ways. First, the retailers can be autonomous enterprises across multiple countries and they cannot indiscriminately open up their databases to others, especially their competitors. Second, the data aggregation service provider can not be fully trusted. Compromise of the server by hackers could lead to a complete privacy loss for all clients should the data be revealed publicly. So the clients cannot completely disclose their data to the service provider or to other clients. In such a situation, it will be important and beneficial to have a distributed data sharing mechanism that is capable of aggregating the data across clients while keeping the data of its participants private.

Data sharing in the health sector is also becoming increasingly important. For instance, many insurance companies collect data on disease incidents, seriousness of the disease, and patient background [Clifton 2002]. There is great interest in sharing their data through a data aggregation service. In addition, the aggregation service would also allow organizations such as the Center for Disease Control to look at the data held by various insurance companies for patterns that are indicative of disease outbreaks. Government agencies also realize the importance of sharing information across the world for devising effective security measures. For example, multiple agencies may need to share their criminal record databases in identifying certain suspects in the case of a terrorist attack. However, they cannot indiscriminately open up their databases to all other agencies. A distributed data sharing mechanism across private databases would greatly facilitate such data aggregation outsourcing services.

Multiprovider Outsourcing. Secure Database Service is a DBMS that provides reliable storage and efficient query execution while not knowing the contents of the database [Hacigumus et al. 2002]. Existing proposals for secure database services have typically been founded on encryption [Hacigumus et al. 2002; Agrawal et al. 2004; Kantarcioglu and Clifton 2004b] where data is encrypted on the (trusted) client side before being stored in the (untrusted) external database and query processing typically involves large overhead.

A new, distributed architecture has been recently proposed for enabling secure data services by allowing an organization to outsource (partition) its data management to two (or any number of) untrusted servers in order to preserve data privacy [Aggarwal et al. 2005]. Figure 1(b) shows a sketch of the architecture. Partitioning of data is performed in a fashion to ensure that the exposure of the contents of any one service provider does not result in a violation of privacy. The client executes queries by transmitting appropriate subqueries to each service provider, and then piecing together the results at the client side. However, we argue that the prequery processing as well as postquery processing incurs an undesirable overhead for the client. A better alternative is to deploy a distributed multiparty data sharing scheme among the service providers for them to share data while maintaining the data privacy at each site and answer the client's queries directly.

Contributions and Organizations. Ideally, given a data sharing task spanning multiple private databases (typically defined by the final knowledge extracted from the databases) in an outsourcing scenario, we wish to achieve the task without revealing any additional information of each individual database apart from the final result. This article proposes a set of decentralized peer-to-peer protocols for supporting data sharing across multiple private databases while minimizing the data disclosure of each individual database.

The article has a number of unique contributions. First, we formalize the design goals and the notion of data confidentiality in terms of information revealed and propose a data privacy metric (Section 3). Second, we describe a set of novel probabilistic computation protocols for important primitive operations such as max, min, and top$k$ that select top$k$ data values of a sensitive attribute across multiple ($n > 2$) private databases (Section 4). Third, we show how complex data sharing tasks can be divided into individual steps that utilize the primitive protocols. In particular, we propose a model for performing a kNN classification across multiple private databases (Section 5). Finally, we perform a formal analysis of the primitive protocol and an experimental evaluation of each of the protocols in terms of their correctness, efficiency, and privacy characteristics (Section 7).

Compared to our preliminary work reported in Xiong et al. [2005] in which only the basic protocol is outlined, this article focuses on the data privacy issues in distributed outsourcing applications. We formalize the design goals and present a detailed analytical study of our basic protocol in terms of precision, efficiency, and privacy characteristics. We show that our experimental results match well with the formal analytical results. More importantly, we develop a new aggregate protocol for kNN classification across multiple private databases by utilizing the basic top$k$ protocol as a building block and provide experimental evaluation on the effectiveness of this new protocol with respect to performance and privacy protection. This aggregation protocol design also demonstrates that by using our basic privacy-preserving data sharing protocol as a building block, we can develop complex protocols to address various data sharing needs in different outsourcing applications that demand data privacy protection for data owners. We emphasize and discuss the impact of outsourcing and Internet technology on the protocols.

2. RELATED WORK

We first discuss the areas of work relevant to our problem and discuss potential techniques and their limitations in addressing the problem.

Data Integration. Data integration has been an important problem that emerges in a variety of situations both commercial (when companies need to integrate their databases) and scientific (combining research results from different bioinformatics repositories). Traditionally, data integration has assumed that the data can be shared freely and that there is little concern with the data confidentiality and privacy constraints. Academic research in data integration has been focused on the scalability and correctness [Garcia-Molina et al. 2001] and recently the semantic integration problem of solving semantic conflicts between heterogeneous data sources [Doan and Halevy 2005]. In industry, several Enterprise Information Integration (EII) products have appeared in the marketplace (such as Actuate Enterprise Information Integration[4] and IBM Information Integration[5]). A collection of articles [Halevy et al. 2005] provides a good discussion on the current state-of-art of the EII industry, the challenges that lie ahead of it, and the controversies surrounding it.

Secure Databases. Research in secure databases, Hippocratic databases, and privacy policy-driven systems [Jajodia and Sandhu 1991; Agrawal et al. 2002, 2005] has been focused on enabling access of sensitive information through centralized role-based access control. Access control does not solve the problem of sharing data among potentially untrusted parties.

Secure Database Outsourcing. The outsourcing of data management has motivated the model where a DBMS provides reliable storage and efficient query execution while not knowing the contents of the database [Hacigumus et al. 2002]. Under this model, one main approach is to encrypt data on the client side and then store the encrypted database on the server side [Hacigumus et al. 2002; Agrawal et al. 2004; Kantarcioglu and Clifton 2004b]. The main limitation of these works is the privacy-efficiency trade-off. Weak encryption functions that allow efficient queries leak far too much information and thus do not preserve data privacy. On the other hand, stronger encryption functions often necessitate impractically expensive query cost. Moreover, encryption and decryption cost are expensive despite the increasing processor speed. Another type of technique includes data partitioning and binning based on a generalization principle to minimize precise information leakage [Hore et al. 1997; Bertino et al. 2005].

[4] http://www.actuate.com/products/enterprisereporting/dataintegration.asp.

[5] http://www-306.ibm.com/software/data/integration/.

Distributed Privacy Preserving Data Integration and Data Mining. The approach of protecting privacy of distributed sources was first addressed by the construction of decision trees [Lindell and Pinkas 2002]. This work closely followed the traditional secure multiparty computation approach and achieved perfect privacy. There has since been work to address association rules [Vaidya and Clifton 2002; Kantarcioglu and Clifton 2004a], naive Bayes classification [Kantarcoglu and Vaidya 2003; Vaidya and Clifton 2003b; Yang et al. 2005], and k-means clustering [Vaidya and Clifton 2003a]. As a recent effort, there is also research on privacy-preserving top$k$ queries [Vaidya and Clifton 2005] and privacy preserving-distributed k-NN classifier [Kantarcioglu and Clifton 2005], both across vertically partitioned data using a k-anonymity privacy model. Agrawal et al. [2003] also introduced the paradigm of minimal information sharing in the information integration domain and proposed a privacy-preserving join protocol between two parties. A few specialized protocols have been proposed, typically in a two-party setting, for example, for finding intersections [Agrawal et al. 2003], and the kth-ranked element [Aggarwal et al. 2004]. Wang et al. [2005] studied the problem of integrating private data sources with vertically partitioned data while satisfying k-anonymity of the data. In contrast, we will show that our protocol does not require any cryptographic operations. It leverages the multiparty network and utilizes a probabilistic scheme to achieve minimal information disclosure and minimal overhead.

Anonymous Network Communication. Another related area is the anonymous network where the requirement is that the identity of a user be masked from an adversary. There have been a number of application-specific protocols proposed for anonymous communication, including anonymous messaging (onion routing [Syverson et al. 1997]), anonymous Web transactions (crowds [Reiter and Rubin 1998]), anonymous indexing (privacy-preserving indexes [Bawa et al. 2003]) and anonymous peer-to-peer systems (mutual anonymity protocol [Xiao et al. 2003]). Some of these techniques may be applicable for data integration tasks where parties opt to share their information anonymously. However, anonymity is a less strong requirement than data privacy. Finally, distributed consensus protocols such as leader election algorithms [Lynch 1996] provide system models for designing distributed algorithms. However, they are not concerned about data privacy constraints of individual nodes.

3. DESIGN GOALS AND THE PRIVACY MODEL

The goal of a data sharing task in our multiclient data aggregation outsourcing scenario is to extract certain knowledge R across a set of private databases, $D_1, D_2, \ldots, D_n (n > 2)$. For example, multiple retailers in the same market sectors may outsource their data aggregation service to a third-party service provider and try to find out the aggregation statistics of their sales such as the total sales or the top$k$ sales among them for a given product line or time period, while keeping their own sales data private. Abstractly, in the top$k$ selection case, the goal is to select top$k$ values of a common attribute across the set of private databases and, at the same time, minimize the data disclosed apart from the final top$k$ values. We will use this example as a running example while discussing vthe problem and presenting our protocol that minimizes the data disclosure at each individual database apart from the final result R. In this section, we first present our design goals, discuss the adversary model, followed by the privacy goal and privacy metrics we use for characterizing and evaluating how the privacy goal is achieved.

3.1 Design Goals

One of the most important requirements of designing such protocols is to guarantee the correctness (deterministic) or accuracy of the results with certain bounds (probabilistic). For example, when designing a classification algorithm across multiple private databases, we would like to have the accuracy as high as that of its counterpart when we assume the data can be freely shared. By utilizing randomization techniques, the accuracy may be traded off for the data confidentiality. The design challenge is to determine the right amount of randomization that the algorithm should insert into its computation such that both the accuracy and privacy requirements can be met.

Another important consideration in designing the protocols is their efficiency. Data sharing tasks often operate on databases containing very large amounts of data; thus secure multiparty computation protocols are too inefficient to be used due to their substantial computation and communication costs.

Finally, the protocols should minimize the information disclosure and maximize the data privacy and confidentiality. Ideally, no information should be disclosed apart from the final result. However, achieving this objective completely might make a protocol very inefficient as shown in multiparty secure computation. In practice, we would like to design highly efficient protocols while minimizing the information disclosure. We will formally define the privacy model and metrics in Section 3.

Thinking of the design space in terms of these three dimensions presents many advantages. At one end of the spectrum, we have the secure multiparty computation protocols that are provably secure in the sense that they reveal the least amount of information and have the highest accuracy; however these protocols are very inefficient. At the other end of the spectrum, we have the counterpart of the protocols that assume the databases are completely open to each other and are highly efficient but not secure. Thus, the challenge is to design efficient protocols while sacrificing as little as possible on accuracy and privacy.

3.2 Privacy Goal

Considering the privacy objective in our design goals, ideally, nodes should not gain any more information about each other's data besides the final results that are public to all the databases. In order to model the individual behaviors of the participating parties, we adopt the *semi-honest* model [Goldreich 2001] that is commonly used in multiparty secure computation research. A semi-honest party follows the rules of the protocol, but it can later use what it sees during execution of the protocol to compromise other parties' data privacy. This kind of behavior is referred to as *honest-but-curious* behavior [Goldreich 2001] or *passive logging* [Wright et al. 2003] in research on anonymous communication protocols. The semi-honest model is realistic for our context as each participating party will want to follow the agreed protocol to get the correct result for their mutual benefit and, at the same time, to reduce the probability and the amount of information leaked (disclosure) about their private data during the protocol execution due to competition or other purposes.

We describe the different types of data exposure we consider and discuss our privacy goal in terms of such exposures. Given a node i and a data value v_i it holds, we identify the following data exposures in terms of the level of knowledge an adversary can deduce about v_i.

(1) *Data value exposure*. An adversary can prove the exact value of v_i ($v_i = a$).

(2) *Data range exposure*. An adversary can prove the range of v_i ($a \leq v_i \leq b$) even though it may not prove its exact value.

(3) *Data probability distribution exposure*. An adversary can prove the probability distribution of v_i ($pdf(v_i) = f$) even though it may prove neither its range nor exact value.

Both data value and data range exposures can be expressed by data probability distribution exposure, in other words, they are special cases of probability distribution exposure. Data value exposure is again a special case of data range exposure. Intuitively, data value exposure is the most detrimental privacy breach. We will focus our privacy analysis on the data value exposures in the rest of this article.

The privacy goal we aim at achieving is to minimize the degree of data value exposures for each individual node. This includes the principle that we are treating all the nodes in the system equally, and no extra considerations will be given to the nodes who contribute to the final results (e.g., the node who owns the global maximum value in top*k* selection). In addition to protecting the data exposure of each node, a related goal could be protecting the anonymity of the nodes who contribute to the final results, though it is not the focus of this article.

3.3 Privacy Metrics

Given the data privacy goal, we need to characterize the degree with which the privacy is attained. The key question is how to measure the amount of disclosure during the computation and what privacy metrics are effective for such measurement. Concretely, we need to quantify the degree of data exposure for a single data item v_i that node i holds. There are a few privacy metrics that are being proposed and used in the existing literature [Reiter and Rubin 1998; Agrawal and Aggarwal 2001]. For our purpose of data value exposure, we extend the probabilistic privacy spectrum [Reiter and Rubin 1998] proposed and adopted for Web transaction anonymity and document ownership privacy [Bawa et al. 2003] to a more general and improved metric. As part of the future work, we are also considering adopting information entropy-based metrics for studying different types of disclosures besides data value exposure.

Our general metric, loss of privacy, characterizes how severe a data exposure is by measuring the relative loss in the degree of exposure. Let R denote the final result set after the execution and IR denote the intermediate result set during the execution. Let $P(C|IR, R)$ denote the probability of C being true given the intermediate results and the final result, and similarly, $P(C|R)$ the probability given only the final result. We define loss of privacy (LoP) in Equation (1). Intuitively, this gives us a measure of the additional information an adversary may obtain given the knowledge of the intermediate result besides the final query result.

$$LoP = P(C|IR, R) - P(C|R). \tag{1}$$

Given the definition of LoP for a single data item at a single node, we define LoP for a node as the average LoP for all the data items used by a node in participating the protocol. We measure the privacy characteristics for the system using the average LoP of all the nodes.

4. PRIVATETOPK PROTOCOL

In this section, we describe a decentralized computation protocol for multiple organizations to perform a top$k$ selection over n private databases (nodes) with minimum information disclosure from each organization. Bearing the privacy goal in mind, we identify two important principles for our protocol design. First, the output of the computation at each node should prevent an adversary from being able to determine the node's data value or data range with any certainty. Second, the protocol should be able to produce the correct final output of a top$k$ query (effectiveness) in a small and bounded number of rounds of communication among the n nodes (efficiency). Using these principles as the design guidelines, we propose a probabilistic protocol with a randomized local algorithm for top$k$ selection across n private databases ($n \geq 3$). To facilitate the discussion of our protocol, we first present a naive protocol as the intuitive motivation and then describe the rational and the algorithmic details of our decentralized probabilistic protocol.

4.1 A Naive Protocol

Consider a group of n databases who wish to select the max value ($k = 1$) of a common attribute. A straightforward way to compute the result without a central server is to have the nodes arranged in a ring in which a global value is passed from node-to-node along the ring. The first node sends its value to its successor. The next node computes the current max value between the value it gets from its predecessor and its own value, and then passes the current max value to its successor. At the end of the round, the output will be the global max value.

Clearly, the scheme does not provide good data privacy. First, the starting node has provable exposure to its successor regarding its value. Second, the nodes that are close to the starting node in the ring have a fairly high probability of disclosing their values. A randomized starting scheme can be used to protect the starting node and avoid the worst case but would not help with the average data value disclosure of all the nodes on the ring. In addition, every node i ($1 \leq i \leq n$) suffers provable exposure to its successor regarding its data range,

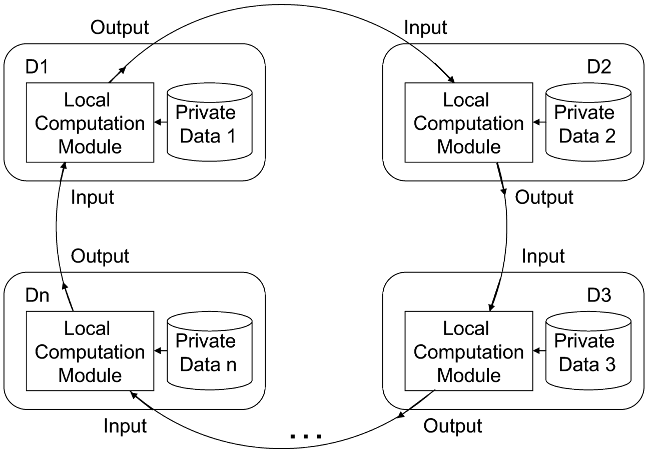


Fig. 2. Distributed data sharing protocol overview.

that is, the successor knows for sure that node i has a value smaller than the value it passes on.

In the rest of this section, we present our probabilistic protocol. We first give a brief overview of the key components of the protocol, and then use the max (min) selection (the top$k$ selection with $k = 1$) to illustrate how the two design principles are implemented in the computation logic used at each node (private database) to achieve the necessary minimum disclosure of private information (our privacy goal).

4.2 Protocol Structure

Figure 2 presents a system overview. Nodes are mapped into a *ring topology* randomly. Each node has a predecessor and successor. It is important to have the random mapping to reduce the cases where two colluding adversaries are the predecessor and successor of an innocent node. We will discuss this more in Section 6. The ring setting is commonly used by distributed consensus protocols such as the leader election algorithm [Lynch 1996]. We also plan to explore other topologies such as hierarchy for designing potentially more efficient protocols. Encryption techniques can be used so that data are protected on the communication channel (from each node to its successor). In case there is a node failure on the ring, the ring can be reconstructed from scratch or by connecting the predecessor and successor of the failed node using ring self-stabilization techniques [Dijkstra 1974]. The *local computation module* is a standalone component that each node executes independently. Nodes follow the semi-honest model and execute the algorithm correctly. The *initialization module* is designed to select the starting node among the n participating nodes and then initialize a set of parameters used in the local computation algorithms.

We assume data are horizontally partitioned across the private databases. In this article, we do not handle the data schema heterogeneity issues. We assume that the database schemas and attribute names are known and are well matched across n nodes. Readers who are interested in this issue may refer to

Elmagarmid et al. [1999] and Doan and Halevy [2005] for some approaches to the problem of schema heterogeneity.

Impact of the Internet. It is important to note that the protocols we will discuss can be applied to a single database, clustered database or distributed database, regardless of the use of the Internet. However, we have to consider a number of specific factors when dealing with data transferred or shared over the Internet. First, it may be advantageous to utilize the underlying network topology instead of a ring topology for optimizing the performance. This is an interesting direction that we would like to explore for future work. Second, when data are shared across many databases in different geographic locations across countries, one way to improve the efficiency is to break the set of nodes into a number of small groups based on geographic proximity and have each group engage in a distributed protocol to compute aggregated data within their group in parallel. Then designated nodes from each group can be selected randomly to represent the data in their group and engage in another global protocol to compute the global aggregated data. Finally, if authentication is done remotely and data is transferred across the Internet, data caching schemes can be deployed to minimize the data transfer cost.

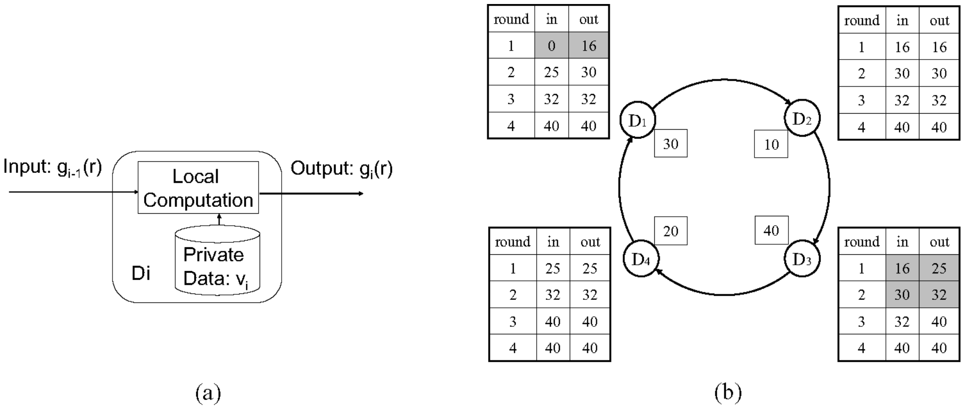
4.3 PrivateMax Protocol

We first present the PrivateMax protocol for max(min) selection (the special case of top$k$ with $k = 1$) over n private databases. The intuitive idea of using a probabilistic protocol is to inject some randomization into the local computation at each node such that the chance of data value disclosure at each node is minimized, and, at the same time, the eventual result of the protocol is guaranteed to be correct. Concretely, the protocol performs multiple rounds in which a global value is passed from node-to-node along the ring. A randomization probability is associated with each round and decreased in the next round to ensure that the final result will be produced in a bounded number of rounds. During each round, nodes inject certain randomization in their local computation with the given probability. The randomization probability is eventually decreased to 0 so that the protocol outputs the correct result.

Randomization Probability. We first define the randomization probability. It starts with an initial probability denoted as p_0 in the first round and decreases exponentially with a dampening factor denoted as d so that it tends to 0 with sufficient number of rounds. Formally, the randomization probability for round r denoted as $P_r(r)$ is defined as follows:

$$P_r(r) = p_0 * d^{r-1}. \tag{2}$$

Randomized Algorithm. Each node, upon receiving the global value from its predecessor, performs the local randomized algorithm and passes the output to its successor. The core idea of this algorithm is to determine when (the right time) to inject randomization and how much (the right amount of randomization) in order to implement the two design principles of the protocol, namely, the output of the algorithm should prevent an adversary from inferring the value or range of the data that the node holds with any certainty, and the randomized



round	in	out
1	0	16
2	25	30
3	32	32
4	40	40

round	in	out
1	16	16
2	30	30
3	32	32
4	40	40

round	in	out
1	25	25
2	32	32
3	40	40
4	40	40

round	in	out
1	16	25
2	30	32
3	32	40
4	40	40

Fig. 3. PrivateMax protocol: (a) local algorithm and (b) illustration.

output should not generate potential errors that lead to incorrect final output of the protocol.

Algorithm 1 Local Algorithm for PrivateMax Protocol (executed by node i at round r)

```
INPUT: $g_{i-1}(r)$, $v_i$, OUTPUT: $g_i(r)$
$P_r(r) \leftarrow p_0 * d^{r-1}$
**if** $g_{i-1}(r) \geq v_i$ **then**
    $g_i(r) \leftarrow g_{i-1}(r)$
**else**
    with probability $P_r$: $g_i(r) \leftarrow$ a random value between $[g_{i-1}(r), v_i)$
    with probability $1 - P_r$: $g_i(r) \leftarrow v_i$
**end if**
```

The randomized algorithm is illustrated in Figure 3(a) and given in Algorithm 1 for node i at round r. The algorithm takes two inputs: (1) the global value node i receives from its predecessor $i-1$ in round r, denoted as $g_{i-1}(r)$, and (2) its own value, denoted as v_i. The algorithm compares these two input values and determines the output value, denoted as $g_i(r)$, in the following two cases. First, if the global value $g_{i-1}(r)$ is greater than or equal to its own value v_i, node i simply returns the current local maximum value ($g_{i-1}(r)$ in this case). There is no need to inject any randomization because the node does not expose its own value in this case. Second, if $g_{i-1}(r)$ is smaller than v_i, instead of always returning the current local maximum value (v_i in this case), node i returns a random value with probability $P_r(r)$ and only returns v_i with probability $1 - P_r(r)$. The random value is generated uniformly from the range $[g_{i-1}(r), v_i)$. Note that the range is open-ended at v_i to warrant that the node will not return the actual v_i but rather a constrained random value in the specified range.

Such randomization has a number of important properties. First, it successfully prevents an adversary from deducing the value or range of v_i with any certainty because the output of node i can be either a random value, or the global value passed by the predecessor of node i, or its own value v_i. Second, the global value monotonically increases as it is passed along the ring, even in the randomization case. Recall the case when randomization is injected, the random value output $g_i(r)$ can be smaller than v_i but has to be greater than or equal to $g_{i-1}(r)$, which ensures that the global value keeps increasing. This monotonic increasing property further minimizes the need for other nodes after node i to have to disclose their own values because they can simply pass on the global value if it is greater than their own values. Finally, the randomized value will not generate any potential errors for the protocol because it is always smaller than v_i and thus smaller than the global maximum value. It will be replaced by the value that is held either by the node i itself or any other node that holds a greater value in a later round as the randomization probability decreases. We will analyze the correctness and data value privacy of the protocol formally in Section 6.

Protocol Details. At the initiation state, every node in the network sorts their values and takes the local max value to participate in the global max selection. The protocol randomly chooses a node from the n participating nodes, say indexed by i with $i = 1$. In addition, the initialization module will set the default global value $g_0(1)$ to the lowest possible value in the corresponding data domain and initialize the randomization probability p_0, the dampening factor d, and the round counter r. Upon completion of the initiation process, node i, upon receiving the global value $g_{i-1}(r)$ from its predecessor at round r, executes the local computation algorithm and passes the output $g_i(r)$ to its successor. The protocol terminates at the starting node after a sufficient number of rounds. We will discuss how to determine the number of rounds needed and what we mean by sufficient in Section 6. It is interesting to note that if we set the initial randomization probability to be 0 ($p_0 = 0$), the protocol is reduced to the naive deterministic protocol.

Figure 3(b) shows an example walk-through of the protocol over a network of 4 nodes, initialized with $p_0 = 1$ and $d = 1/2$. Assume the protocol starts from node 1 with the initial global value $g_0(1) = 0$. Node 1 returns a random value between [0,30), say 16. Node 2 passes 16 to node 3 because it is greater than its own value 10. The protocol continues till the termination round when all nodes pass on the final results.

4.4 PrivateTop*k* Protocol

Now we describe the general protocol PrivateTop*k* for top*k* selection. It works in a similar manner as PrivateMax ($k = 1$) in the probabilistic scheme. At the initialization step, each node sorts its values and takes the local set of top*k* values as its local top*k* vector to participate in the protocol since it will have at most k values that contribute to the final top*k* result. The protocol performs multiple rounds in which a current global top*k* vector is passed from node-to-node along the ring. Each node i, upon receiving the global vector from its predecessor at round r, performs a randomized algorithm and passes its output to its successor node.

Randomized Algorithm. The complexity of extending the protocol from max to general top*k* lies in the design of the randomized algorithm. We want it to have the same properties as the PrivateMax local algorithm (Algorithm 1), namely, to guarantee the correctness on one hand and minimize the data value disclosure on the other hand. We might consider using the same idea of generating random values and inject them into the output of the global top$k$ vector at node i ($1 \leq i \leq n$) in order to hide the node's own values. However, with k values in the local top$k$ vector, we need to make sure that the randomly generated values will eventually be shifted out from the final global top$k$ vector. In other words, it is not as straightforward as in the PrivateMax algorithm where a random value less than a node's value will be replaced eventually.

Algorithm 2 Local Algorithm for PrivateTop$k$ Protocol (executed by node i at round r)

```
INPUT: $G_{i-1}(r)$, $V_i$, OUTPUT: $G_i(r)$
$P_r(r) \leftarrow p_0 * d^{r-1}$
$G'_i(r) = \text{topK}(G_{i-1}(r) \cup V_i)$
$V'_i \leftarrow G'_i(r) - G_{i-1}(r)$
$m \leftarrow |V'_i|$
**if** $m = 0$ **then**
   $G_i(r) \leftarrow G_{i-1}(r)$
**else**
   with probability $1 - P_r(r)$: $G_i(r) \leftarrow G'_i(r)$
   with probability $P_r(r)$:
   $G_i(r)[1 : k - m] \leftarrow G_{i-1}(r)[1 : k - m]$
   $G_i(r)[k - m + 1 : k] \leftarrow$ sorted list of $m$ random values from $[min(G'_i(r)[k] - \delta, G_{i-1}(r)[k - m + 1]), G'_i(r)[k])$
**end if**
```

Algorithm 2 gives a sketch of a randomized algorithm for general top$k$ selection executed by node i at round r. The input of the algorithm is (1) the global vector node i received from its predecessor $i - 1$ in round r, denoted as $G_{i-1}(r)$, and (2) its local top$k$ vector, denoted as V_i. The output of the algorithm is the global vector denoted as $G_i(r)$. Note that the global vector is an ordered multiset that may include duplicate values. The algorithm first computes the real current top$k$ vector, denoted as $G'_i(r)$, over the union of the set of values in $G_{i-1}(r)$ and V_i, using a merge-sort algorithm. It then computes a subvector of V_i, denoted as V'_i, which contains only the values of V_i that contribute to the current top$k$ vector $G'_i(r)$ by taking a set difference of the set of values in $G'_i(r)$ and $G_{i-1}(r)$. Note that the union and set difference here are all multiset operations. The algorithm then works under two cases.

Case 1. The number of elements in V'_i, m, is 0, that is, node i does not have any values to contribute to the current top$k$. In this case, node i simply passes on the global top$k$ vector $G_{i-1}(r)$ as its output. There is no randomization needed because the node does not expose its own values.

Case 2. Node i contributes $m (0 < m \leq k)$ values in the current top$k$. Figure 4 gives an illustrative example where $m = 3$ and $k = 6$. In this case, node i only returns the real current top$k$, $(G'_i(r))$ with probability $1 - P_r(r)$. Note that a node only does this once, that is, if it inserts its values in a certain round, it will simply pass on the global vector in the rest of the rounds. With probability $P_r(r)$, it copies $G_{i-1}(r)[1 : k - m]$, the first $k - m$ values from $G_{i-1}(r)$, and generates the last m values randomly and independently from $[min(G'_i(r)[k] - \delta, G_{i-1}(r)[k - m + 1]), G'_i(r)[k])$, where $G'_i(r)[k]$ denotes the kth (last) item in

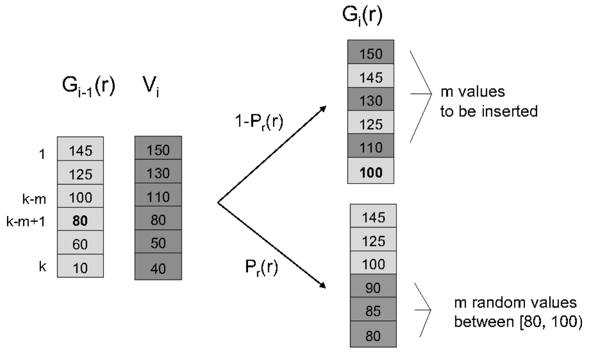


Fig. 4. Illustration of PrivateTop$k$ local algorithm.

$G_i'(r)$, $G_{i-1}(r)[k-m+1]$ denotes the $k-m+1$th item in $G_{i-1}(r)$, and δ denotes a minimum range for generating the random values. The reason for generating m random values is because only the last m values in the output are guaranteed to be shifted out in a later round when the node inserts its real values if the global vector has not been changed by other nodes. The range is designed in such a way that it increases the values in the global vector as much as possible while guaranteeing the random values do not exceed the smallest value in the current top$k$ so they will be eventually replaced. In an extreme case when $m = k$ (the current top$k$ vector is equal to V_i), it will replace all k values in the global vector with k random values each randomly picked from the range between the first item of $G_{i-1}(r)$ and the kth (last) item of V_i.

It is worth noting that when $k = 1$, the local PrivateTop$k$ algorithm becomes the same as the local PrivateMax algorithm. We report our experimental evaluation on the correctness and privacy characteristics of the general protocol in Section 7.

5. BUILDING AGGREGATE PROTOCOLS

We have presented a set of protocols for primitive operations such as max, min, and top$k$ selection. In addition to being used to select the statistical information across multiple private databases, they can also be used as a building block for more complex data-sharing tasks. We show in this section how we build a kNN classifier across multiple private databases based on the previous protocols and other existing primitive protocols.

5.1 Classification Problem and the *k*NN Classifier

The motivating applications we discussed in Section 1 and many other data-sharing applications can be modeled as data classification problems and solved using a classification algorithm such as k-nearest neighbor classifier. In its training phase, a k-nearest neighbor classifier simply stores all the training examples. Every time a new query instance is encountered, its relationship to the previously stored instances is examined according to certain distance or similarity measure, and the k-nearest neighbors that are most similar to the new instance are used to assign a classification to the new instance. For instance, to determine whether a pattern of disease is indicative of a outbreak, we can train a kNN classifier to recognize disease outbreak patterns and use it to classify a query pattern as an outbreak or not, based on the distance (similarity) between the new query pattern and existing patterns.

A kNN classifier has many advantages over other machine learning classifiers. For example, in a distributed setting, the overhead of training a classifier is avoided until a query instance must be classified. In addition, instead of estimating the classification function once for the entire instance space, a kNN classifier can estimate it locally and differently for each new instance to be classified. This is extremely useful in situations in which multiple parties are sharing dynamic databases with complex data. In the rest of the section, we present a model for the basic distance-weighted kNN algorithm across private databases based on the primitive protocols previously presented.

5.2 Private$k$NN Protocol

Given the set of n private databases D_1 , D_2 , . . . , D_n distributed at n organizations, we assume that data is horizontally partitioned. A kNN classifier views instances as points in a $|A|$-dimensional space, where $|A|$ is the number of attributes of each instance.

Algorithm 3 kNN Classification Protocol

```
INPUT: $x$, OUTPUT: $classification(x)$

(1) Each node computes the distance between $x$ and each point $y$ in its database, $d(x, y)$, selects $k$ smallest distances (locally), and stores them in a local distance vector $ldv$.
(2) Using $ldv$ as input, the nodes use the PrivateTop$k$ protocol to select $k$ nearest distances (globally), and stores them in $gdv$.
(3) Each node selects the $k$th nearest distance $\Delta$: $\Delta = gdv(k)$.
(4) Assuming there are $v$ classes, each node calculates a local classification vector $lcv$ for all points $y$ in its database: $\forall 1 \leq i \leq v$, $lcv(i) = \sum_y w(d(x, y)) * [f(y) == i] * [d(x, y) \leq \Delta]$, where $d(x, y)$ is the distance between $x$ and $y$, $f(y)$ is the classification of point $y$, and $[p]$ is a function that evaluates to 1 if the predicate $p$ is true, and 0 otherwise.
(5) Using $lcv$ as input, the nodes use a privacy preserving addition protocol [Clifton et al. 2003] to calculate the global classification vector $gcv$.
(6) Each node classifies $x$ as $classification(x) \leftarrow \max_{i \in V} gcv(i)$.
```

Given a query instance (point), a kNN classifier uses the k-nearest neighbors of the query to classify it. We perform the kNN classification across private databases in two steps. The first step is to find the k-nearest neighbors that can be distributed among n nodes. Each node calculates the distance of every point in its database from x and selects the local k smallest distances. In order to minimize the disclosure of the distance information, we can transfer the problem of finding k-nearest neighbors into the top$k$ (bottom$k$) selection and utilize the PrivateTop k protocol to find the bottom$k$ distances. Note that the PrivateTop$k$ protocol can be easily modified to perform bottom$k$ selection if we define the linear order among the values by a $<$ relation.

Once a node determines which points in its private database are among the k-nearest neighbors of the query instance based on the k-nearest distances, the second step is to determine the global classification based on the local classifications. Each of these neighbors is supposed to vote in favor of its own class, with the strength of the vote determined by its distance to x. All the votes are then added and the class with the highest votes is selected as the classification of x. In order to minimize the disclosure of individual classification information, we can represent the local classification of x as a vector of v votes, assuming the number of classes is v. The ith element of this vector is the amount of votes the ith class received from the points in this node's database which are among the k-nearest neighbors of x. Then we can add these local classification vectors by a simple privacy-preserving addition protocol [Clifton et al. 2003] to determine the global classification vector. Once each node knows the global classification vector, it can determine the classification of x without disclosing its local classifications. Algorithm 3 shows a sketch of the complete protocol.

6. ANALYSIS

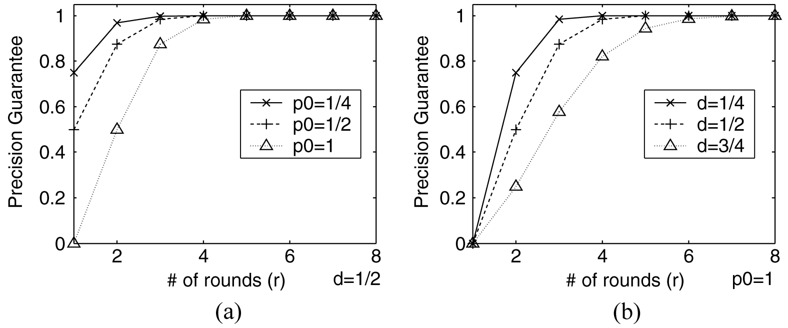
We conducted a formal analysis on the PrivateMax protocol and experimental evaluations on each of the protocols in terms of their correctness, efficiency, and privacy characteristics. In this section, we present the analytical results we obtained.

6.1 Correctness

Let $g(r)$ denote the global value at the end of round r and $P(g(r) = v_{max})$ denote the probability that $g(r)$ is equal to the global max value v_{max}. At round $j(1 \leq j \leq r)$, if the global value has not reached v_{max}, the nodes who own v_{max} have a probability $1-P_r(j)$ to replace the global value with v_{max}. Once the global value reaches v_{max}, all nodes simply pass it on. So after round r, the global value will be equal to v_{max} as long as one of the nodes that owns v_{max} has replaced the global value with v_{max} in any of the previous rounds. Thus the probability of the protocol returning the global maximum value after round r can be computed as $P(g(r) = v_{max}) \geq 1 - \prod_{j=1}^{r} P_r(j)$ and we can derive Equation (3). It shows that, for any $0 < p_0 \leq 1$ and $0 < d < 1$, the precision bound increases monotonically with increasing r. For any given number of nodes, we can make the computed global value equal to the global max value with a probability very close to 1 by increasing the number of rounds.

$$P(g(r) = v_{max}) \geq 1 - p_0^r * d^{\frac{r(r-1)}{2}}. \tag{3}$$

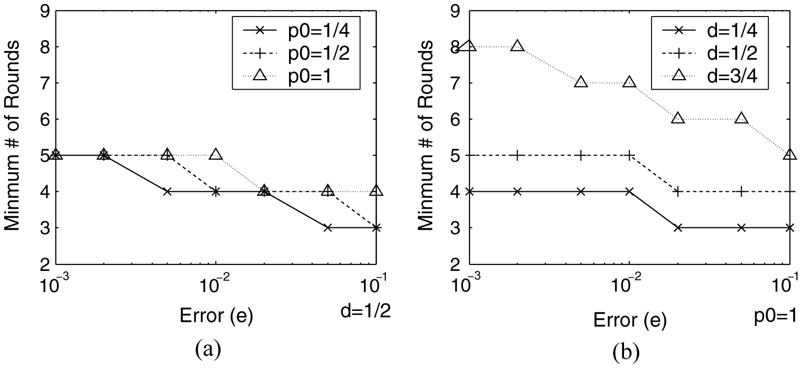
Figure 5(a) and (b) plot the precision bound in Equation (3) with an increasing number of rounds (r) for varying initial randomization probability (p_0) and dampening factor (d), respectively. We can see that the precision increases with the number of rounds. A smaller p_0 with a fixed d results in a higher precision in the earlier rounds and reaches the near-perfect precision of 100% faster. A smaller d with a fixed p_0 makes the protocol reach the near-perfect precision of 100% even faster.



(a)

(b)

Fig. 5. Precision guarantee with number of rounds.



(a)

(b)

Fig. 6. Required number of rounds with precision guarantee.

6.2 Efficiency

Now we analyze the efficiency of the protocol in terms of the computation and communication cost. The computation at each node in the protocol does not involve any cryptographic operations and should be negligible compared to the communication cost over a network of n nodes. The communication cost is determined by two factors. The first is the cost for a single round which is proportional to the number of nodes on the ring. The second is the number of rounds that is required for a desired precision. For any precision bound $\epsilon (0 < \epsilon < 1)$, we can determine a minimum number of rounds, denoted by r_{min}, such that the result is equal to the global max value with the probability greater than or equal to $1-\epsilon$. Since we have $P(g(r) = v_{max}) \geq 1 - p_0^r * d^{\frac{r(r-1)}{2}} \geq 1 - p_0 * d^{\frac{r(r-1)}{2}}$ from Equation (3) we can ensure $P(g(r) = v_{max}) \geq 1 - \epsilon$ by requiring $1 - p_0 * d^{\frac{r(r-1)}{2}} \geq 1 - \epsilon$. We solve this equation and derive a minimum number of rounds for the desired precision $(1-\epsilon)$ in Equation (4). We can see that it scales well with the desired precision $(1-\epsilon)$ in the order of $O(\sqrt{log\epsilon})$.

$$r_{min} = \left\lceil \frac{1}{2} * \left(1 + \sqrt{8 * \frac{\log(\epsilon/p_0)}{\log d} - 1} \right) \right\rceil . \tag{4}$$

Figure 6(a) and (b) plot the minimum number of rounds in Equation (4) for varying error bound (ϵ) with varying initial randomization probability (p_0)

and dampening factor (d), respectively. Note that the X axis is of logarithmic scale. We can see that the protocol scales well with increasing desired precision (decreasing ϵ). In addition, a smaller p_0 and a smaller d are desired for better efficiency, with d having a larger effect on the reduction of the required number of rounds.

6.3 Data Value Privacy

In addition to ensuring correct output and increasing efficiency of the protocol, another important goal of the protocol is preserving the data privacy of individual participating nodes in the network. The communication between nodes consists of sending the current global value from one node to its successor. An adversary may utilize the value it receives from its predecessor to try to gain information about the data its predecessor and other nodes hold. We dedicate this section to analyzing the loss of data value privacy in the protocol using the metric we proposed in Section 3.

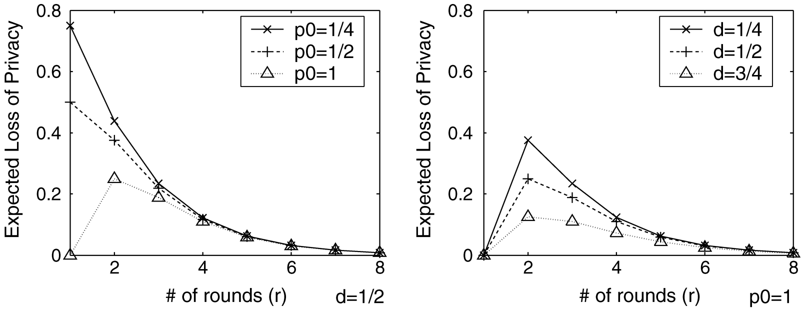
Without loss of generality, we assume the protocol starts from node 1. We now analyze the loss of privacy for node i with value v_i. Since node i passes its current global value $g_i(r)$ to its successor $i+1$ in round r, the best the successor can do with respect to the exact data value that node i holds is to guess that $v_i = g_i(r)$. Recall Equation (1) in Section 3 where the loss of privacy is defined as the relative probability of a node holding a particular value with and without the intermediate result. Let $P(v_i = g_i(r)|g_i(r), v_{max})$ denote the probability that node i holds the value $g_i(r)$ with the knowledge of the intermediate result $g_i(r)$ and $P(v_i = g_i(r)|v_{max})$ denote the probability without it. If $g_i(r) = v_{max}$, we have $P(v_i = g_i(r)|v_{max}) = 1/n$ as all nodes have the same probability of holding the global maximum value. Otherwise, we approximately have $P(v_i = g_i(r)|v_{max}) = 0$, assuming a large data value domain. Now let us look at $P(v_i = g_i(r)|g_i(r), v_{max})$ for both naive protocol and probabilistic protocol and derive the loss of privacy (LoP).

Naive Protocol. In the naive protocol where only one round is needed, the global value g_i that node i passes on is the current maximum value of all its previous nodes and itself. Since all of them have the same probability to hold the current maximum value so we have $P(v_i = g_i(r)|g_i(r), v_{max}) = 1/i$. Thus the data value loss of privacy for node i in naive protocol is $1/i - 1/n$ if $g_i = v_{max}$ and $1/i$ otherwise.

It is clear that the loss of privacy for node i in the naive protocol depends on its position. Nodes closer to the starting node suffer a larger loss of privacy. In the worst case, the starting node ($i = 1$) has provable exposure of its value to its successor. On average, the loss of privacy is greater than $\sum_{i=1}^{n}(1/i - 1/n)/n$. Using the inequality bound for $\sum_{i=1}^{n} 1/i$ (the nth Harmonic number), we can derive the average LoP bound for the naive protocol in Equation (5). We can see that it is fairly high especially when n is small.

$$LoP_{Naive} > \frac{\ln n}{n}. \tag{5}$$

Probabilistic Protocol. The probabilistic max selection protocol requires multiple rounds. Since aggregating the values a node passes on in different rounds



(a)

(b)

Fig. 7. Expected loss of privacy in different rounds.

does not help with determining its exact data value, though it may help with determining the probability distribution of the value, we first compute the loss of privacy for node i at each round r, and then take the highest result in all the rounds as the final loss of privacy for node i.

Recall that in the probabilistic computation in the PrivateMax Algorithm (shown in Algorithm 1), a node only replaces the global value with its own value with probability $1 - P_r(r)$ when the value it receives is smaller than its own value. Thus we have $P(v_i = g_i(r)|g_i(r), v_{max}) = P(v_i > g_{i-1}(r)) * (1 - P_r(r)) + P(v_i = g_{i-1}(r))$. We performed an analysis on the expected value for $g_i(r)$ and derived an approximate bound of expected *LoP* for node i in round r. By taking the highest (maximum) *LoP* of all rounds for each individual node, we derive an upper bound on the average expected *LoP* for all the nodes in Equation (6). Note here that the term $-P(vi = gi(r)|vmax)$ is intentionally left out to derive an n-independent upper bound for the loss of privacy. As a result, the derived analytical bound is conservative, and our experiments show the actual loss of privacy is much lower (see Section 7).

$$E(LoP_{probabilistic}) \leq max_r \left(\frac{1}{2^{r-1}} * (1 - p_0 * d^{r-1}) \right) \tag{6}$$

Figure 7 plots the bound for each individual round (the term inside the *max* function of Equation (7)) with varying randomization parameters. Figure 7(a) shows the effect of p_0 with d set to 1/2. It is interesting to see that p_0 plays an important role in the loss of privacy. A larger p_0 results in a lower loss of privacy in the first round. The reason is quite intuitive since a larger p_0 implies that more nodes have injected randomized values instead of returning the real current max value in the computation. With a smaller p_0, the loss of privacy gradually decreases from the peak of loss as the protocol converges. With a larger p_0, such as $p_0 = 1$, the loss of privacy starts with 0 in the first round and increases in the second round to the peak of loss, and then gradually decreases. If we compare the peak loss of privacy in different rounds, we conclude that a larger p_0 provides better privacy. Figure 7(b) shows the effect of d with p_0 set to 1. We see that a larger d corresponds to a lower loss of privacy, starting from the second round, though with a small margin. Overall, by tuning the parameters p_0 and d, we can keep the loss of privacy very low. Our experimental evaluation in Section 7 confirms with our analytical results regarding the loss of privacy.

Now we briefly discuss the loss of privacy under the scenario where the predecessor and the successor of node i happen to collude with each other. Assuming that an adversary has intermediate results of both $g_{i-1}(r)$ and $g_i(r)$, we have $P(v_i = g_i(r)|g_{i-1}(r), g_i(r), v_{max}) = 1 - P_r(r)$ when $g_{i-1}(r) < g_i(r)$. Knowing this still does not give the adversary any certainty in determining the data value of node i, especially in the beginning rounds when $P_r(r)$ is large. Intuitively, when $P_r(r)$ gets close to 0, $g_{i-1}(r)$ should already be getting close to v_{max} so there is a very small chance for the data at node i to be disclosed. It is interesting to note though, if node i happens to hold v_{max}, then it will be susceptible to provable exposure if it has two colluding neighbors. One technique to minimize the effect of collusion is for a node to ensure that at least one of its neighbors is trustworthy. This can be achieved in practice by having nodes arrange themselves along the network ring(s) according to certain trust relationships such as digital certificate-based [Blaze et al. 1996] combined with reputation-based [Xiong and Liu 2004]. Further, we can extend the probabilistic protocol by performing the random ring mapping at each round so that each node will have different neighbors at each round.

7. EXPERIMENTAL EVALUATIONS

This section presents a set of initial results from our experimental evaluation of the protocols in terms of correctness and privacy characteristics.

7.1 Experiment Setup

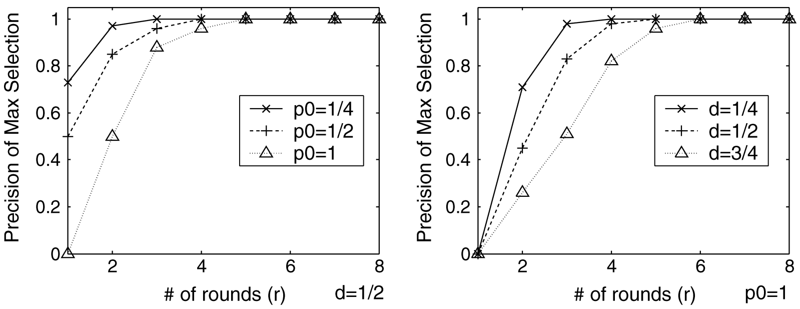
We implemented the privateTop$k$ protocol in MatLab and the Private$k$NN protocol in C++. For PrivateMax and PrivateTop k protocols, we used a synthetic dataset. The attribute values at each node are randomly generated over the integer domain [1, 10000]. We experimented with various distributions of data, such as uniform distribution, normal distribution, and zipf distribution. The results are similar so we only report the results for the uniform distribution. For Private$k$NN protocol, we used a publicly available medical dataset PIMA[6] used for diagnostic purposes. It predicts whether a patient shows signs of diabetes given data like the 2-hour serum insulin concentration and body mass index. This dataset contains 768 instances belonging to 8 different classes, with each instance having 8 different attributes. In each run, we randomly partitioned the data into a training set containing $\frac{3}{4}$ of the data and a test set containing $\frac{1}{4}$ of the data.

The experiment consists of n nodes. In each experiment, we performed 100 separate runs on each different dataset. We evaluate the average accuracy and loss of privacy using LoP. The LoP is computed using the estimated probability of whether the guess of a data value is true at each round. Table I lists the main parameters for the experiments.

[6] http://www.ics.uci.edu/mlearn/databases/pima-indians-diabetes.

Table I. Experiment Parameters

Parameter	Description
n	# of nodes in the system
k	parameter in top$k$
p_0	initial randomization probability
d	dampening factor for randomization probability



(a) (b)

Fig. 8. Precision of PrivateMax with number of rounds.

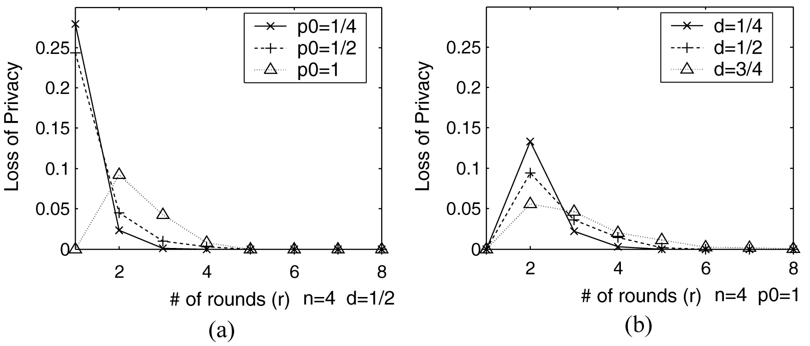
7.2 PrivateMax Protocol

Precision. We first verify the correctness of PrivateMax protocol ($k = 1$). Figure 8(a) and (b) show the precision with increasing number of rounds (r) for different initial randomization probability (p_0) and dampening factor (d), respectively. We see that the experimental results match the analytical bounds in Figure 5. The precision reaches to 100% as the number of rounds increases. A smaller p_0 results in a higher precision in the first round and makes the precision go up to 100% faster as the number of rounds increases, though with a small margin. A smaller d reaches the perfect precision of 100% much faster.

Privacy Characteristics. We evaluate the privacy characteristics of PrivateMax protocol in terms of their data value loss of privacy. We first study the loss of privacy of the protocol in each round during the execution with different randomization parameters. We experimented with different numbers of nodes and the trends in different rounds are similar but most pronounced with a small number of nodes. So we only report the results for $n = 4$ to show the different loss of privacy in different rounds and will present another set of experiments later to show the effect of a varying number of nodes.

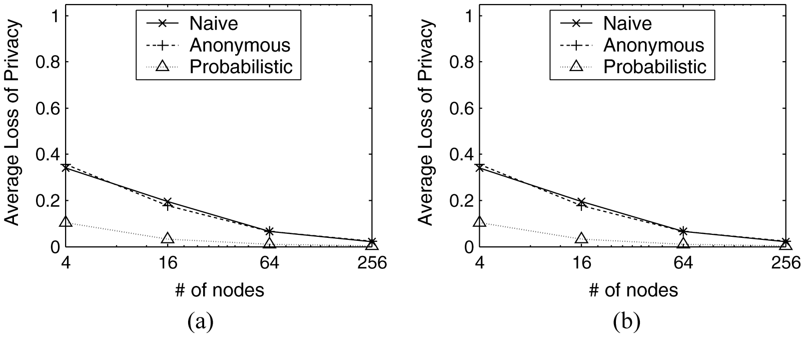
Figure 9(a) and (b) show the average data value loss of privacy for all nodes in different rounds with varying initial randomization probability (p_0) and dampening factor (d), respectively. Note that these results match the analytical results in the parameter effects but much lower than the analytical upper bound (recall Section 6.3).

In the previous experiments, we have shown the loss of privacy in different rounds during the execution. For the rest of the experiments, we will take the highest (peak) loss of privacy among all the rounds for a given node to measure its overall loss of privacy because that gives us a measure of the highest level of knowledge an adversary can obtain regarding the node's data value.



(a) (b)

Fig. 9. Loss of privacy for PrivateMax in different rounds.

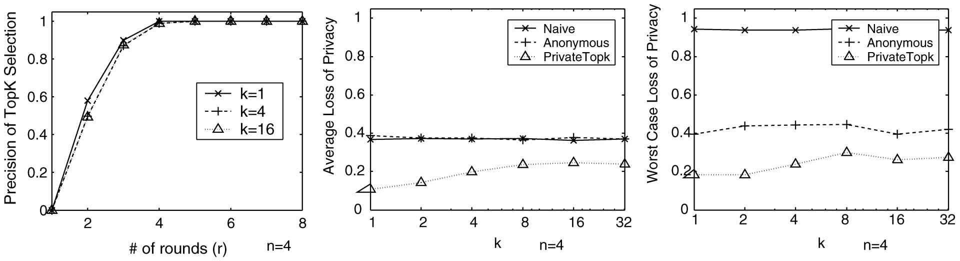


(a) (b)

Fig. 10. Comparison of loss of privacy with number of nodes.

The experiments reported in the following compare the probabilistic protocol with the naive protocol. For comparison purposes, we include the anonymous naive protocol which uses a randomized starting scheme instead of a fixed starting node to provide the anonymity of the starting node. We show both average and worst-case loss of privacy for different nodes in the system. By worst case, we mean highest loss of privacy among all the nodes, and it typically happens at the starting node in the fixed starting scheme. Our goal is to show the effectiveness of our probabilistic protocol over the naive ones and the benefit of randomly selecting the starting node.

Figure 10(a) and (b) show the average and worst-case data value loss of privacy for all nodes with different number of nodes (n). We make a few interesting observations. First, the anonymous starting scheme has the same average LoP as the naive protocol but avoids the worst-case scenario. This can be seen in Figure 10(b) where the naive protocol suffers a loss of privacy close to 100% (at the starting node), while the anonymous protocol does not change significantly from the average case in Figure 10(a). Second, the probabilistic protocol achieves significantly better privacy than the naive protocols. The loss of privacy is close to 0 in most cases. Finally, all of the protocols have a decreasing loss of privacy as the number of nodes increases. Interestingly, when the number of nodes is sufficiently large, the anonymous naive protocol performs reasonably well compared to the probabilistic protocol. However, most of the



(a) (b) (c)

Fig. 11. (a) Precision and (b-c) loss of privacy of the PrivateTop*k* protocol.

privacy-preserving data integration will be among tens or hundreds of nodes with membership controls. A network of size on the order of thousands seldom happens.

7.3 PrivateTop*k* Protocol

We have presented results for the PrivateMax protocol so far. Now we evaluate the general PrivateTop*k* protocol in terms of its correctness and privacy characteristics. In addition to running the same set of experiments for max protocol, we also run a set of experiments with varying k. Since most of the results we obtained are similar to those for PrivateMax protocol, we only report the results with varying k for PrivateTop*k* protocol in the following.

Precision. We first verify the correctness of the protocol. In order to evaluate the precision of top-k selection, we first define the precision metric we use. Assume *TopK* is the real set of top-k values and R is the set of top-k values returned. We define the precision as $|R \cap TopK|/K$. Figure 11(a) shows the precision of the top$k$ protocol with an increasing number of rounds (r) for varying k. The precision reaches to 100% in all lines after a sufficient number of rounds. The effect of k on the convergence is not significant. We also ran experiments for varying n with $k > 1$, and the result did not show any significant effect.

Privacy Characteristics. Now we report the loss of privacy for the general top$k$ protocol with varying k and its comparison to the naive protocol. Figure 11(b) and (c) show the average and worst-case data value loss of privacy for all nodes with varying k. We see that the probabilistic protocol achieves significantly better privacy than the naive protocols. Interestingly, the probabilistic protocol has an increasing loss of privacy as k increases. An intuitive explanation is that the larger the k, the more information a node exposes to its successor and hence the larger the loss of privacy.

7.4 Private*k*NN Protocol

Accuracy. In this experiment, we compare the accuracy of Private$k$NN against its counterpart nonprivate distributed kNN classifier, which assumes data can be freely shared, and study the effect of the number of rounds in the nearest-neighbor selection protocol on the classification accuracy. To do this, we measure the absolute value of the difference between the accuracies of the two classifiers

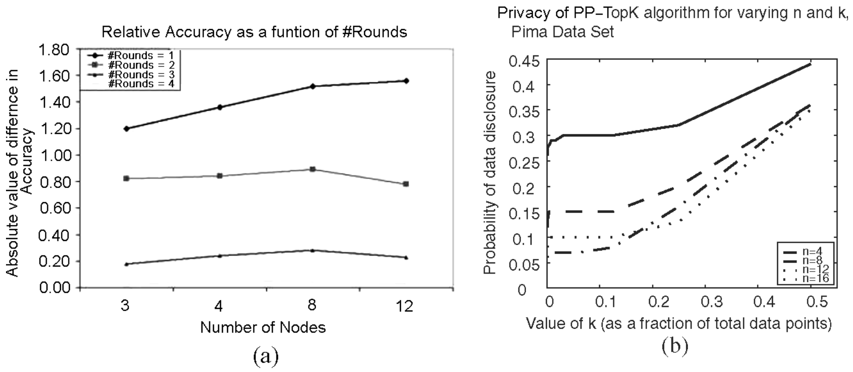


Fig. 12. (a) Relative accuracy and (b) loss of privacy of Private*k*NN protocol.

when trained and tested on the same data with a varying number of rounds in Private*k*NN protocol.

The result is presented in Figure 12(a). It confirmed that we are able to make the classifier as accurate as an ordinary classifier by running nearest-distance selection for a sufficient number of rounds (4 in this case which is still efficient). Note that even if we choose very conservative settings for the Private*k*NN classifier by running a small number of rounds for efficiency, the accuracy is still comparable to an ordinary classifier. This is not a contradiction because an incorrect result of the k-nearest distance selection step (recall the probabilistic output of the PrivateTop*k* protocol) could actually produce a list of approximate k-nearest neighbors for which the *k*NN algorithm performs reasonably well or even better due to the characteristics of the dataset.

Privacy Characteristics. Information disclosure for the Private*k*NN protocol is mainly on the distance information during the k-nearest neighbor selection step using the PrivateTop*k* protocol. It has been proven [Clifton et al. 2003] that the addition protocol does not disclose any information. Thus, in this section, we measure the amount of distance information revealed by a node to its successor during the execution of the PrivateTop*k* algorithm. In this experiment, we run the protocol for $r = 2$ rounds with $p_0 = 1$ and $d = 1/2$. We measure the probability that a node is able to correctly identify a value in the global vector it received from its predecessor as belonging to its predecessor.

Figure 12(b) shows the loss of privacy of distance information for the Private*k*NN protocol with a varying number of nodes and varying k. We note that for all values of n, there is a large range of k such that the probability of a node revealing information to its successor is less than half. With very large values of k, however, a node has a higher probability of inserting its values in the global vector and this increases its chances of revealing its values to its successor. Our experiments indicate that even if we run the algorithm for a larger number of rounds and use a range of values for the randomization factor d, the probability that a node reveals its values to its successor is still very low.

It is important to note that in the Private*k*NN protocol, we utilize only the distances of points from the query instance and not the actual points in a node's database. Thus, whatever information is leaked is only about the distance of a point from the query instance and never the actual coordinates of an instance itself. This is an inherent strength of our model and guarantees that the actual coordinates of a point in a node's database is never revealed to other nodes compared to the ordinary distributed classifier where the exact positions of data points are communicated.

8. CONCLUSION

We described two scenarios for outsourcing data aggregation services and presented a set of decentralized peer-to-peer protocols for supporting data sharing across multiple private databases while minimizing the data disclosure among individual parties. Our work continues in several directions. First, we are exploring alternative privacy metrics such as information entropy-based measures [Agrawal and Aggarwal 2001] for a more extensive evaluation and analysis of the privacy characteristics of the protocols. Second, we are exploring different topologies and other performance optimization techniques for outsourcing applications over the Internet. Finally, we are also interested in investigating the possibility of building adaptive protocols for outsourcing services across organizations and countries with different privacy requirements.

ACKNOWLEDGMENTS

Our special thanks are due to the TOIT special issue editors and reviewers for their valuable comments and suggestions that helped to improve the article.

REFERENCES

Aggarwal, G., Bawa, M., Ganesan, P., Garcia-Molina, H., Kenthapadi, K., Motwani, R., Srivastava, U., Thomas, D., and Xu, Y. 2005. Two can keep a secret: A distributed architecture for secure database services. *Conference on Innovative Data Systems Research (CIDR)*.

Aggarwal, G., Mishra, N., and Pinkas, B. 2004. Secure computation of the kth ranked element. *IACR Conference on Eurocryption*.

Agrawal, D. and Aggarwal, C. C. 2001. On the design and quantification of privacy preserving data mining algorithms. *Symposium on Principles of Database Systems*.

Agrawal, R., Bird, P., Grandison, T., Kieman, J., Logan, S., and Rjaibi, W. 2005. Extending relational database systems to automatically enforce privacy policies. *21st International Conference on Data Engineering (ICDE)*.

Agrawal, R., Evfimievski, A., and Srikant, R. 2003. Information sharing across private databases. *ACM SIGMOD International Conference on Management of Data*.

Agrawal, R., Kieman, J., Srikant, R., and Xu, Y. 2002. Hippocratic databases. *International Conference on Very Large Databases (VLDB)*.

Agrawal, R., Kiernan, J., Srikant, R., and Xu, Y. 2004. Order-preserving encryption for numeric data. *ACM SIGMOD International Conference on Management of Data*.

Bawa, M., Bayardo, R. J., and Agrawal, R. 2003. Privacy-preserving indexing of documents on the network. *29th International Conference on Very Large Databases (VLDB)*.

Bertino, E., Ooi, B., Yang, Y., and Deng, R. H. 2005. Privacy and ownership preserving of outsourced medical data. *International Conference on Data Engineering (ICDE)*.

Blaze, M., Feigenbaum, J., and Lacy, J. 1996. Decentralized trust management. *IEEE Conference on Privacy and Security*.

Clifton, C. 2002. Tutorial on privacy, security, and data mining. *13th European Conference on Machine Learning and 6th European Conference on Principles and Practice of Knowledge Discovery in Databases*.

Clifton, C., Kantarcioglu, M., Lin, X., Vaidya, J., and Zhu, M. 2003. Tools for privacy preserving distributed data mining. *SIGKDD Explorations*.

Dijkstra, E. W. 1974. Self-stabilizing systems in spite of distributed control. *Commun. ACM 17*, 11.

Doan, A. and Halevy, A. 2005. Semantic integration research in the database community: A brief survey. *AI Magazine* (Special Issue on Semantic Integration).

Elmagarmid, A., Rusinkiewicz, M., and Sheth, A., Eds. 1999. *Management of Heterogeneous and Autonomous Database Systems* 1st Ed. Morgan Kaufmann.

Garcia-Molina, H., Ullman, J. D., and Widom, J. D. 2001. *Information Integration*, Chapter 20. Prentice Hall.

Goldreich, O. 2001. Secure multi-party computation. Working Draft, version 1.3.

Hacigumus, H., Iyer, B., Li, C., and Mehrotra, S. 2002. Executing SQL over encrypted data in the database service provider model. *ACM SIGMOD Conference on Management of Data*.

Hacigumus, H., Iyer, B., and Mehrotra, S. 2002. Providing database as a service. *International Conference on Data Engineering (ICDE)*.

Halevy, A. Y., Ashish, N., Bitton, D., Carey, M. J., Draper, D., Pollock, J., Rosenthal, A., and Sikka, V. 2005. Enterprise information integration: successes, challenges and controversies. *ACM SIGMOD International Conference on Management of Data*.

Hore, B., Mehrotra, S., and Tsudik, G. 1997. A privacy-preserving index for range queries. *ACM Symposium on Principles of Distributed Computing*.

Jajodia, S. and Sandhu, R. 1991. Toward a multilevel secure relational data model. *ACM SIGMOD International Conference on Management of Data*.

Kantarcioglu, M. and Clifton, C. 2004a. Privacy preserving data mining of association rules on horizontally partitioned data. *IEEE Trans. Knowl. Data Engin. 16*, 9.

Kantarcioglu, M. and Clifton, C. 2004b. Security issues in querying encrypted data. Tech. rep. TR-04-013, Purdue University.

Kantarcioglu, M. and Clifton, C. 2005. Privacy preserving k-nn classifier. *International Conference on Data Engineering (ICDE)*.

Kantarcoglu, M. and Vaidya, J. 2003. Privacy preserving naive Bayes classifier for horizontally partitioned data. *IEEE ICDM Workshop on Privacy Preserving Data Mining*.

Lindell, Y. and Pinkas, B. 2002. Privacy preserving data mining. *J. Crypto. 15*, 3.

Lynch, N. A. 1996. *Distributed Algorithms*. Morgan Kaufmann Publishers.

Markey, E. J. 2005. Outsourcing privacy: Countries processing U.S. social security numbers, health information, tax records lack fundamental privacy safeguards. A staff report prepared at the request of Edward J. Markey, U.S. House of Representatives.

Reiter, M. K. and Rubin, A. D. 1998. Crowds: Anonymity for Web transactions. *ACM Trans. Inform. Syst. Secur. (TISSEC) 1*, 1.

Syverson, S., Coldsehlag, D. M., and Reed, M. C. 1997. Anonymous connections and onion routing. *IEEE Symposium on Security and Privacy*.

Vaidya, J. and Clifton, C. 2002. Privacy preserving association rule mining in vertically partitioned data. *The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Vaidya, J. and Clifton, C. 2003a. Privacy-preserving k-means clustering over vertically partitioned data. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Vaidya, J. and Clifton, C. 2003b. Privacy preserving naive Bayes classifier for vertically partitioned data. *The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Vaidya, J. and Clifton, C. 2005. Privacy-preserving top-k queries. *International Conference on Data Engineering (ICDE)*.

Wang, K., Fung, B. C. M., and Dong, G. 2005. Integrating private databases for data analysis. *IEEE Intelligence and Security Informatics Conference (ISI)*.

Wright, M., Adler, M., Levine, B. N., and Shields, C. 2003. Defending anonymous communications against passive logging attacks. *IEEE Symposium on Security and Privacy*.

Xiao, L., Xu, Z., and Zhang, X. 2003. Mutual anonymity protocols for hybrid peer-to-peer systems. *International Conference on Distributed Computing Systems (ICDCS)*.

Xiong, L., Chitti, S., and Liu, L. 2005. Topk queries across multiple private databases. *25th International Conference on Distributed Computing Systems (ICDCS)*.

Xiong, L. and Liu, L. 2004. PeerTrust: supporting reputation-based trust in peer-to-peer communities. *IEEE Trans. Knowl. Data Engin. 16*, 7.

Yang, Z., Zhong, S., and Wright, R. N. 2005. Privacy-preserving classification of customer data without loss of accuracy. *SIAM Conference on Data Mining (SDM)*.

Received December 2005; revised March 2007; accepted April 2007