Top$k$ Queries across Multiple Private Databases

Li Xiong, Subramanyam Chitti, Ling Liu
College of Computing
Georgia Institute of Technology
{lxiong,chittis,lingliu}@cc.gatech.edu

Abstract

Advances in distributed service-oriented computing and global communications have formed a strong technology push for large scale data integration among organizations and enterprises. However, concerns about data privacy become increasingly important for large scale mission-critical data integration applications. Ideally, given a database query spanning multiple private databases, we wish to compute the answer to the query without revealing any additional information of each individual database apart from the query result. In practice, we may relax this constraint to allow efficient information integration while minimizing the information disclosure. In this paper, we propose an efficient decentralized peer-to-peer protocol for supporting aggregate queries over multiple private databases while respecting the privacy constraints of participants. The paper has three main contributions. First, it formalizes the notion of loss of privacy in terms of information revealed at individual participating databases. Second, it presents a novel probabilistic decentralized protocol for topk selection across multiple private databases that minimizes the loss of privacy. Third, it experimentally evaluates the protocol in terms of its correctness, efficiency and privacy characteristics.

1 Introduction

Information integration has been an important area of research as there is great benefit for organizations and individuals in sharing their data. Traditionally, information integration research has assumed that information in each database can be freely shared. Recently, it has been recognized that concerns about data privacy increasingly become an important aspect of data integration because organizations or individuals do not want to reveal their private databases for various legal and commercial reasons.

Application Scenarios. The increasing need for privacy preserving data integration is driven by several trends [4]. In the business world, with the push towards end-to-end integration between organizations and their suppliers, service providers, and trade partners, information sharing may occur across multiple autonomous enterprises. Full disclosure of each database is undesirable. It is also becoming common for enterprises to collaborate in certain areas and compete in others. This in turn requires selective information sharing.

Another important application scenario is driven by security. Government agencies realize the importance of sharing information for devising effective security measures. For example, multiple agencies may need to share their criminal record databases to identify suspects in the circumstance of a terrorist attack. However, they cannot indiscriminately open up their databases to all other agencies.

Such concerns about data privacy place limits on information integration. We are faced with the challenge of data integration while respecting privacy constraints. Ideally, given a database query spanning multiple private databases, we wish to compute the answer to the query without revealing any information apart from the query result.

Current Techniques and Research Challenges. There are two main existing techniques that one might use for building privacy preserving data integration applications, and we discuss below why they are inadequate.

One technique is to use a trusted third party and have the participating parties report the data to the trusted third party, which performs the data integration task and reports back the result to each party. However, finding such a trusted third party is not always feasible. The level of trust required for the third party with respect to intent and competence against security breaches is too high. Compromise of the server by hackers could lead to a complete privacy loss for all participating parties should the data be revealed publicly.

The other technique is secure multi-party computation, [9, 8] which has developed theoretical methods for securely computing functions over private information such that parties only know the result of the function and nothing else. However, the methods require substantial computation and communication costs and are impractical for multi-party

large data-base problems.

Agrawal et al [4] recently proposed a new paradigm of information integration with minimal necessary sharing across private databases. As a tradeoff for efficiency and practicability, the constraint of not revealing any additional information apart from the query result can be relaxed sometimes to allow minimal additional information to be revealed. As an example, they developed protocols for computing intersection and equijoin between two parties that is still based on cryptographic primitives but is more efficient than secure multi-party computation and has minimal information disclosure.

Given this paradigm, research opportunities arise for developing efficient specialized protocols for different operations. One important operation is statistics queries over multiple private databases, such as top$k$ data values of a sensitive attribute. In particular, when $k = 1$, it becomes the max(min) query. For example, a group of competing retail companies in the same market sector may wish to find out statistics about their sales, such as the top sales revenue among them, but to keep the sales data private at the same time. The top$k$ function can be also served as a building block for more complex distributed top$k$ queries or data mining algorithms such as kNN classification. The design goal for such protocols is three fold. First, it should guarantee accurate results no matter what techniques are used for preserving data privacy. Second, it should be efficient in terms of both computation and communication costs. In order to minimize the computation cost, expensive cryptographic operations should be limited or avoided. Finally, it should minimize the information disclosure apart from the query results for each participant.

Contributions and Organizations. Bearing these design goals in mind, we propose a protocol for selecting top$k$ data values of a sensitive attribute across multiple ($n \geq 3$) private databases. This paper has three main contributions. First, we formalize the data privacy goal and the notion of loss of privacy in terms of information revealed, by proposing a data privacy metric (Section 2). Second, we propose a novel probabilistic decentralized protocol for privacy preserving top$k$ selection (Section 3). Third, we evaluate the protocol experimentally in terms of its correctness, efficiency and privacy characteristics (Section 4). We provide a brief overview of the related work (Section 5) and conclude the paper with a summary, and a brief discussion of future work (Section 6).

2 Privacy Model

In this section we define the problem of top$k$ queries across private databases. We present the privacy goals that we focus on in the paper, followed by privacy metrics for characterizing and evaluating how the privacy goals are achieved.

Problem Statement. The input to the problem is a set of private databases, $D_1, D_2, \cdots, D_n (n \geq 3)$. A top$k$ query is to find out the top$k$ values of a common attribute of all the individual databases. We assume all data values of the attribute belong to a publicly known data domain. Now the problem is to select the top$k$ values with minimal disclosure of the data values that each database has besides the final result.

Adversary Model. We adopt the *semi-honest* model of adversaries [8] that is commonly used in multi-party 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. Such kind of behavior is referred to as honest-but-curious behavior [8] and also referred to as passive logging [18] in research on anonymous communication protocols. The *semi-honest* model is realistic for our context where each participating party will want to follow the agreed protocol to get the correct result for their mutual benefit and at the same time reduce the probability and amount of information disclosure about their private data during the protocol execution.

Privacy Goal. We focus on achieving data privacy for top$k$ queries in this paper. Ideally, besides the query result that is revealed to all the databases, nodes should not gain any more information about each others' data. In other words, our goal is to minimize data exposure among the multiple parties apart from the result of the top$k$ query.

We describe the different types of data exposure we consider and formulate our privacy goals 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 ($a \leq v_i \leq b$) but not prove its exact value; and (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. Intuitively, data value exposure is the most detrimental privacy breach. Due to the space restrictions, we will focus our discussions to data value exposures in the rest of this paper.

Our privacy goal is to minimize the "degree of data value exposures" (defined in the next section) for each individual node. We treat all the nodes in the system equally and no extra consideration will be given to nodes who contribute to the final top$k$ values. In addition to protecting the data exposure of each node, a related goal is protecting the anonymity of the nodes who contribute to the final results. However, it is not the focus of this paper.

Privacy Metrics. Given the data privacy goal, we need to

characterize the degree with which privacy is attained. The key question is how to measure the amount of information disclosed during the computation. 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 [14, 2]. For our purpose, we adopted the probabilistic privacy spectrum [14] and propose a more general and improved metric.

A node's privacy is compromised when an adversary is able to make a claim C about the data at other nodes (e.g., node i has a value $v = 10$) with a high degree of certainty. Besides the results of the query (denoted by R) that are made public to all the nodes, an adversary will also have access to intermediate results (denoted by IR) during the execution of the protocol. Thus the loss of privacy due to the execution of the protocol is the added information an adversary can infer due to his knowledge of the intermediate results, over and above the information he may derive by his knowledge of the final results.

We propose a general metric - Loss of Privacy - to characterize how severe a data exposure is by measuring the additional certainty an adversary gains in making a claim. Let $P(C|R, IR)$ denote the probability of predicate C being true given the final result and the intermediate results, and let $P(C|R)$ be the probability given only the final result. We define Loss of Privacy (LoP) in Equation 1:

$$LoP = |P(C|R, IR) - P(C|R)| \quad (1)$$

Intuitively, Loss of Privacy quantifies the amount of extra information an adversary obtains from his knowledge of the intermediate results. The metric gives a value within the range $[0, 1)$. We use the phrase "complete loss of privacy" to refer to the case where the Loss of Privacy is close to 1.

Concretely, consider the case where C is in the form of $v_i = a$ (i.e., a data value predicate) and R is the final top$k$ value set denoted as $TopK$. If $a \in TopK$, every node has the same probability to hold a so we have $P(v_i = a|TopK) = 1/n$ (for n participants). Otherwise, when ($a \notin TopK$), it is close to impossible for an adversary to guess the exact value of a node given only the final result. This is especially true when the data domain is large enough, because a node can take any of the values in the data domain. So we approximate $P(v_i = a|TopK)$ with 0. Thus, Loss of Privacy is equal to either $P(v_i = a|IR, TopK) - 1/n$ or $P(v_i = a|IR, TopK)$, depending on whether $a \in TopK$.

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 the node while participating in the protocol. Intuitively, when nodes participate in the protocol with their local top$k$ values, the more values that get disclosed, the larger the LoP for the node. We measure the privacy characteristics for the protocol using the average LoP of all the nodes.

3 The Decentralized Protocol

In this section we describe a decentralized protocol PrivateTop$k$ - for multiple organizations to execute top$k$ queries across n private databases (nodes), such that each node suffers minimum information disclosure.

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 be sufficiently random so as to prevent an adversary from being able to determine that node's data value or data range with certainty. Second, the protocol should be able to produce the correct final output of a top$k$ query (correctness) 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$k$ queries across n private databases ($n \geq 3$). To facilitate the discussion of our protocol, we first present a naive protocol to execute top$k$ queries, and then present our distributed probabilistic protocol.

3.1 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 this value to its successor. This continues until the first node receives a value from its predecessor. At this point, the starting node will know the global max value, and can broadcast this information.

Clearly, the scheme does not provide good data privacy. The starting node has complete loss of privacy to its successor regarding its value. 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 would protect the starting node and avoid the worst case, but it would not help with the average data value disclosure of all the nodes on the ring. Furthermore, every node i ($1 \leq i \leq n$) suffers a complete loss of privacy regarding its data range, i.e. the successor knows for sure that node i has a value smaller than the value it passes on. This leads us to consider alternative protocols for better privacy preservation.

3.2 Protocol Structure

To address the shortcomings of the naive protocol, we devised a probabilistic protocol. First, we give a brief overview of the key components of the protocol and then use max (min) queries (top$k$ queries with $k = 1$) to illustrate how the two design principles are achieved by the computation performed at each node to achieve minimum disclosure of information. The protocol is designed to run over a decentralized network with a ring topology, and consists of the node communication scheme, the local computation module and initialization module at each node.

Nodes are mapped onto a ring randomly; thus each node has a predecessor and successor. The random mapping reduces the ability of two colluding adversaries to be the predecessor and successor of an innocent node. The ring setting is commonly used by distributed consensus protocols such as leader election algorithm [13]. We also plan to explore other topologies such as hierarchy for designing potentially more efficient protocols. We assume secure communication channels between a node and its successor. The local algorithm is a standalone component that each node executes. 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.

In this paper 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 [7] for some approaches to the problem of schema heterogeneity.

3.3 PrivateMax Protocol

We first present the protocol for max(min) queries (the special case of top$k$ with $k = 1$) over n private databases, referred to as PrivateMax protocol, to help readers understand the key ideas and techniques used in our protocol design. Later, we generalize to top$k$ queries in next subsection.

The intuitive idea of using a probabilistic protocol is to inject the right amount of randomization into the local computation at each node, such that the probability of data value disclosure at each node is minimized while ensuring that the eventual result of the protocol is guaranteed to be correct. Concretely, the protocol consists of multiple rounds in which a global value is passed from node to node along the ring. During each round, nodes inject some randomization in their local computation with a certain randomization probability that depends on the round. The randomization probability monotonically approaches 0 as the number of rounds increases, ensuring that the protocol eventually produces 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. The randomization probability for round r denoted as $P_r(r)$ is defined as follows:

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

PrivateMax Local 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 and how to inject 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 output should not generate potential errors that lead to incorrect final output of the protocol.

Algorithm 1 PrivateMax Local Algorithm (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 $P_r(r)$: $g_i(r) \leftarrow$ a random value between $[g_{i-1}(r), v_i)$
  with probability $1 - P_r(r)$: $g_i(r) \leftarrow v_i$
end if

A sketch of the PrivateMax local algorithm is 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 way. 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 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 ensure that the node will not reveal its private value v_i.

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. This is 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. 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 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 randomization will not generate an erroneous value which is greater than the global maximum, because the global value passed around the ring is always smaller than or equal to 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.

Protocol Details. At initialization, each node in the network chooses its local max value to participate in the global max selection. The protocol randomly chooses a node from the n participating nodes, say node i, with $i = 1$. Node 1 initializes the global value $g_0(1)$ to the lowest possible value in the data domain; the randomization probability to p_0, the dampening factor d (recall equation 2), and the round counter r.

Upon the completion of the initiation process, the local computation module is invoked at node 1. Each node j, 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. Note that if we set the initial randomization probability to be 0 ($p_0 = 0$), the protocol is reduced to the naive deterministic protocol.

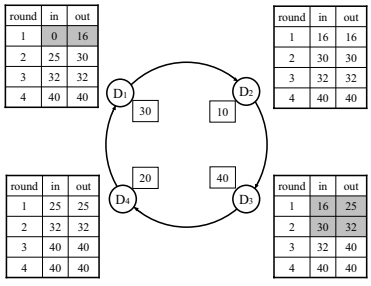


Figure 1. Illustration for PrivateMax Protocol

Figure 1 shows an example walk-through of the PrivateMax 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 $g_0(1) = 0$. In the first round ($r = 1$), the randomization probability $P_r(1)$ is equal to 1, so if a node receives a value smaller than its own value, it will always return a random value between the received value and its own value. As a result, 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. Node 3 returns a random value between [16,40), say 25; and node 4 passes value 25 to the first node because it is greater than its own value 20. The protocol proceeds this way until the final round. After the final round all nodes simply pass on the final result.

This example illustrates how the decentralized probabilistic protocol works and why it ensures that each node retains good privacy about the exact value and the range of their data.

3.4 PrivateTop$k$ Protocol

Now we describe the general PrivateTop$k$ protocol for top$k$ selection. It works similarly as PrivateMax protocol ($k = 1$) in terms of the probabilistic scheme. The complexity of extending the protocol from max to general top$k$ lies in the design of the randomized algorithm.

As the initial step, each node chooses its local 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. As in PrivateMax protocol, the initialization module randomly picks a node from the n participating nodes as the starting node, initializes the global top$k$ vector to the lowest possible values in the corresponding data domain, sets the round counter r, and initializes the randomization probability p_0 and the dampening factor d.

The protocol performs multiple rounds in which a current global top$k$ vector is passed from node to node along the ring network. 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. The starting node terminates the protocol after a sufficient number of rounds.

PrivateTop$k$ Local Algorithm. The randomized algorithm is the key component of the probabilistic PrivateTop$k$ protocol. We want the algorithm to have the same properties as those of PrivateMax algorithm (Algorithm 1), namely, guaranteeing correctness while minimizing data value disclosures. To accomplish this, we can use the same idea of generating random values and injecting them into the output of the global top$k$ vector at node i ($1 \leq i \leq n$) in order to hide the nodes own values. However, with k values in the local top$k$ vector, we need to make sure that the ran-

domly generated values will eventually be shifted out from the final global top$k$ vector.

Algorithm 2 PrivateTop$k$ Local Algorithm (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 PrivateTop$k$ protocol with respect to node i executing at round r. The inputs to the algorithm are (1) the global vector node i receives 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 correct top$k$ vector, denoted as $G'_i(r)$, over the union of the set of values in $G_{i-1}(r)$ and V_i. It then computes a sub-vector 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, i.e. 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 need not expose its own values.

Case 2: Node i contributes $m (0 < m \leq k)$ values in the current top$k$. Figure 2 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, i.e. 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 the first $k-m$ values from $G_{i-1}(r)$ and generate 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

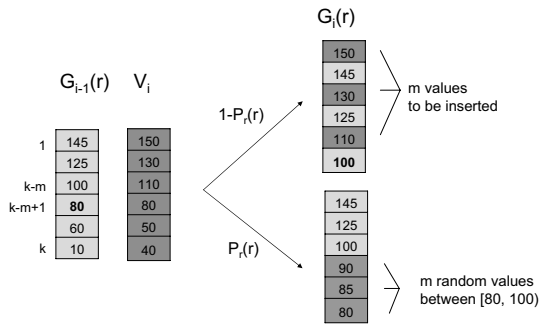


Figure 2. Illustration for PrivateTop$k$ Local Algorithm

$G'_i(r)$, $G_{i-1}(r)[k-m+1]$ 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 is 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 PrivateTop$k$ local algorithm becomes the same as the PrivateMax local algorithm. We report our experimental evaluation on the correctness and privacy characteristics of the general protocol in Section 4.

4 Experimental Evaluations

In this section, we present an initial set of experiments evaluating the protocol in terms of its correctness, efficiency, and privacy characteristics. We also conducted formal analysis on the protocol. The analytical results are omitted in this paper due to space restrictions. Please refer to our technical report [20] for detailed initial analytical results.

4.1 Experiment Setup

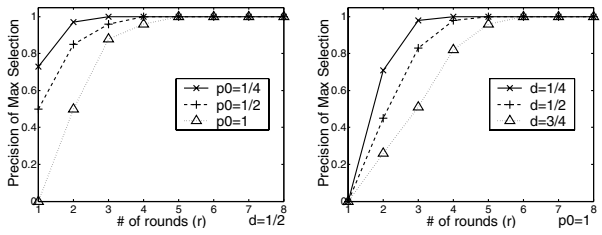
The system consists of n nodes. The attribute values at each node are randomly generated over the integer domain

Param.	Description
n	# of nodes in the system
k	parameter in top$k$
p_0	initial randomization prob.
d	dampening factor for randomization prob.

Table 1. Experiment Parameters

$[1, 10000]$. 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. The experiment proceeds by having the nodes compute top$k$ values using the probabilistic protocol. We evaluate the accuracy and privacy properties. Each plot is averaged over 100 experiments. Table 1 lists the main parameters for the experiments.

4.2 Precision and Efficiency of Max Selection



(a) (b)

Figure 3. Precision of Max Selection with Increasing Number of Rounds

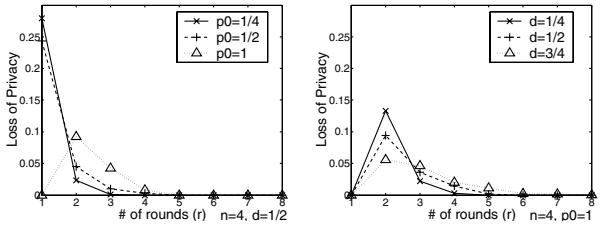
We first verify the correctness of the PrivateMax protocol ($k = 1$). Figure 3(a) and (b) show the precision with increasing number of rounds (r) for different initial randomization probability (p_0 and dampening factor (d) respectively. We observe that the precision increases to 100% as the number of rounds increases. A smaller p_0 results in a higher precision in the first round and makes the precision converge to 100% faster. A smaller d makes the precision reach 100% much faster. In other words, a smaller p_0 and d provides better efficiency in terms of the number of rounds required given a precision guarantee.

Note that the communication cost for a single round is proportional to the number of nodes on the ring. One possible way to improve the efficiency for a system with a larger number of nodes is to break the nodes into a number of small groups and have each group compute their group maximum value in parallel and then compute the global maximum value at designated nodes, which could be randomly selected from each small group.

4.3 Privacy Characteristics of Max Selection

We evaluate the privacy characteristics of the protocol in terms of their data value loss of privacy. In particular, we want to answer a number of questions. What is the loss of privacy during the execution of the algorithm? How does the number of nodes affect the privacy characteristics? How do the randomization parameters affect the privacy characteristics and how to select them? How does the protocol compare to the naive protocol?

Loss of Privacy in Different Rounds. We first study the loss of privacy of the protocol in each round during the execution with different randomization parameters.



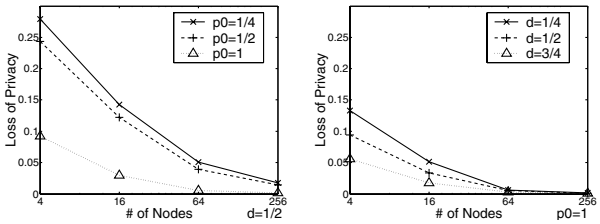
(a) (b)

Figure 4. Loss of Privacy for Max Selection in Different Rounds

Figure 4(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. With a smaller p_0, the loss of privacy reaches highest in the first round and gradually decreases. With a larger p_0 (e.g., $p_0 = 1$), the loss of privacy is zero in the first round and reaches the peak in the second round and then gradually decreases. If we look at the peak loss of privacy, a larger p_0 provides a better privacy. In Figure 4(b), the trend is the same as the one of $p_0 = 1$. However, a larger d results in a lower peak loss of privacy. Intuitively, a larger p_0 and d add more randomization into the protocol and thus provide better privacy.

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.

Effect of Number of Nodes. We now report the experiments showing how the number of nodes affects the loss of privacy of the protocol.



(a) (b)

Figure 5. Loss of Privacy for Max Selection with Varying Number of Nodes

Figure 5(a) and (b) show the average data value loss of privacy for all nodes with varying initial randomization probability (p_0) and dampening factor (d) respectively. We can see that the loss of privacy decreases with increasing number of nodes. This is very intuitive because the larger the number of nodes, the faster the global value increases and thus the less probability the nodes have to disclose their own values. Again, the result shows that a smaller p_0 and d provide a better privacy.

Selection of Randomization Parameters. This set of experiments is dedicated to study the effect of randomization parameters on both privacy characteristics and efficiency of the protocol. Recall the experiments described so far, a larger p_0 and d provide better privacy but requires more cost in terms of number of rounds required for a given precision guarantee. Our design goal is to increase the efficiency while minimizing the loss of privacy. Put differently, we want to see what parameters give good tradeoff between privacy and efficiency.

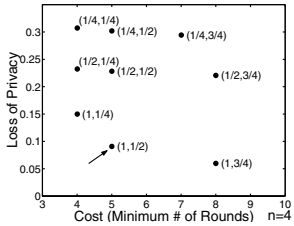
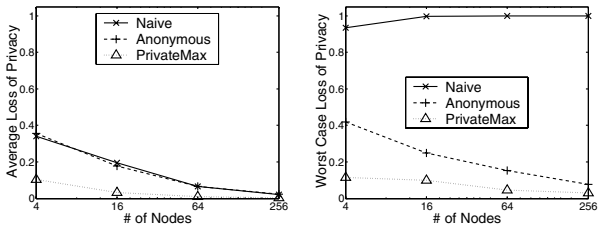


Figure 6. Tradeoff between Privacy and Efficiency with Randomization Parameters

Figure 6 shows the loss of privacy on X axis and the cost in terms of number of rounds for a given precision guarantee ($\epsilon = 0.001$) on Y axis for varying randomization parameter pairs (p_0, d). We can see that p_0 has a dominating effect on the loss of privacy while d has a dominating effect on efficiency. An optimal set of parameters may be chosen according to the requirements or preferences on privacy and efficiency of different applications. In general, the data points in the lower left corner of the graph provide a good tradeoff between efficiency and privacy. In the rest of our experiments, we use $p_0 = 1$ and $d = 1/2$ as default parameters.

Comparison of Different Protocols. We have discussed the naive protocol with fixed starting node, and our probabilistic protocol. The experiments reported below compare the probabilistic protocol with the naive protocol. For comparison purposes, we also include the anonymous naive protocol which uses a randomized starting scheme, instead of 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 effectives of our probabilistic protocol over the naive one and the benefit of randomly selecting the starting node.



(a) (b)

Figure 7. Comparison of Loss of Privacy with Varying Number of Nodes

Figure 7(a) and (b) show the average and worst case data value loss of privacy for all nodes with different number of nodes (n) respectively. 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 7(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 7(a). The probabilistic PrivateMax protocol achieves significantly better privacy than the naive protocols. It is close to 0 in most cases. All 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 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 occurs in the data integrations scenarios we envision.

4.4 Precision and Efficiency of Top$k$ Selection

We have presented results for PrivateMax protocol so far. Now we evaluate the general PrivateTop$k$ protocol in terms of its correctness, efficiency, and privacy characteristics. In addition to running the same set of experiments for PrivateMax 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 in these two subsections the results for PrivateTop$k$ protocol with varying k to show the effect of k.

We first verify the correctness of PrivateTop$k$ 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 $|R \cap TopK|/k$.

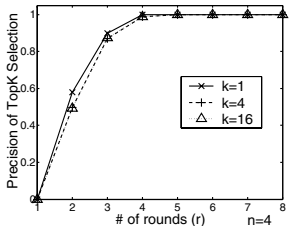


Figure 8. Precision of Top$k$ Selection with Increasing Number of Rounds

Figure 8 shows the precision of PrivateTop$k$ protocol with increasing number of rounds (r) for varying k. The precision reaches to 100% in all lines after a fairly small 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.

4.5 Privacy Characteristics of Top-k Selection

Now we report the loss of privacy for PrivateTop$k$ protocol with varying k and its comparison to the naive protocol. Figure 9(a) and (b) show the average and worst case data value loss of privacy for all nodes of different protocols with varying k. We can make a few interesting observations. We see that the probabilistic PrivateTop$k$ protocol achieves significantly better privacy than the naive protocols. Interestingly, PrivateTop$k$ 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.

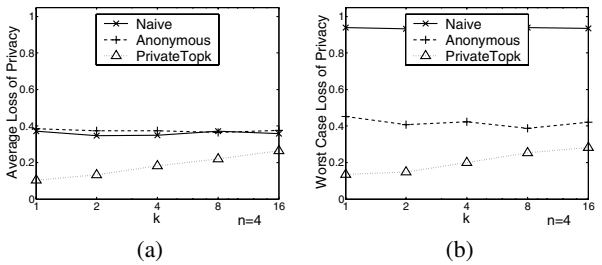


Figure 9. Comparison of Loss of Privacy with Varying k

5 Related Work

Privacy related problems in databases have been an active and important research area. Research in secure databases, Hippocratic databases and privacy policy driven systems [12, 5, 3] has been focused on enabling access of sensitive information through centralized role-based access control. More recently research has been done in areas such as privacy preserving data mining, privacy preserving query processing on outsourced databases, and privacy preserving information integration.

In privacy preserving data mining [17], the main approach is to use data perturbation techniques to hide precise information in individual data records, as the primary task in data mining is the development of models and patterns about aggregated data. In database outsourcing scenarios, the main technique is to use data partitioning to evaluate obfuscated range queries with minimal information leakage [10, 11]. However, these techniques may not apply to information integration tasks where precise results are desired.

In the information integration domain, Agrawal et al. [4] introduced the paradigm of minimal information sharing for privacy preserving information integration. A few specialized protocols have been proposed under this paradigm, typically in a two party setting, e.g., for finding intersections [4], and kth ranked element [1]. Though still based on cryptographic primitives, they achieve better efficiency than traditional multi-party secure computation methods by allowing minimal information disclosure. In contrast, our protocol does not require any cryptographic operations. It leverages the multi-party network and utilizes a probabilistic scheme to achieve minimal information disclosure and minimal overhead.

As a recent effort, there is also research on privacy preserving top$k$ queries across vertically partitioned data using k-anonymity privacy model [16]. Our protocol computes top$k$ selection across horizontally partitioned data and uses a different data privacy model. The top$k$ selection alone can be also served as a primitive function for more complex

aggregate queries or data integration tasks.

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 [15]), anonymous web transactions (Crowds [14]), anonymous indexing (Privacy Preserving Indexes [6]) and anonymous peer-to-peer systems (Mutual anonymity protocol [19]). 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 [13] provide system models for designing distributed algorithms. However they are not concerned about data privacy constraints of individual nodes.

6 Conclusion

We have presented PrivateTop$k$ protocol for top-k selections across multiple private databases. We formalized the notion of loss of privacy in terms of information revealed and developed an efficient decentralized probabilistic protocol, which aims at selecting top$k$ data items across multiple private databases with minimal information disclosure. We evaluated the correctness and privacy characteristics of the proposed protocol through experimental evaluations.

Our work on privacy preserving data integration continues along several directions. First, we are continuing our formal analysis of the protocol including privacy analysis with potential collusions among the nodes. Also, given the probabilistic scheme, it is possible to design other forms of randomization probability and randomized algorithms. We are interested in formally discovering the optimal randomized algorithm. Second, we are exploring optimization techniques for networks with a large number of participating databases. Third, we plan to relax the semi-honest model assumption and address the situations where adversaries may not follow the protocol correctly. Finally, we are developing a privacy preserving kNN classifier on top of the top$k$ protocol. We are also interested in developing efficient protocols for more complex aggregate or top$k$ queries.

Acknowledgement

The authors would like to thank the anonymous reviewers for their valuable comments. This research is partially supported by NSF CNS, NSF CCR, NSF ITR, DoE SciDAC, DARPA, CERCS Research Grant, IBM Faculty Award, IBM SUR grant, HP Equipment Grant, and LLNL LDRD.

References

[1] G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the kth ranked element. In *IACR Conference on Eurocrypt*, 2004.

[2] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *ACM PODS*, 2001.

[3] R. Agrawal, P. Bird, T. Grandison, J. Kieman, S. Logan, and W. Rjaibi. Extending relational database systems to automatically enforce privacy policies. In *21st ICDE*, 2005.

[4] R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *SIGMOD*, 2003.

[5] R. Agrawal, J. Kieman, R. Srikant, and Y. Xu. Hippocratic databases. In *VLDB*, 2002.

[6] M. Bawa, R. J. Bayardo, and R. Agrawal. Privacy-preserving indexing of documents on the network. In *29th VLDB*, 2003.

[7] A. Elmagarmid, M. Rusinkiewicz, and A. Sheth, editors. *Management of Heterogeneous and autonomous database systems*. Morgan Kaufmann; 1st edition, 1999.

[8] O. Goldreich. Secure multi-party computation, 2001. Working Draft, Version 1.3.

[9] S. Goldwasser. Multi-party computations: past and present. In *ACM Symposium on Principles of Distributed Computing*, 1997.

[10] H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the database service provider model. In *SIGMOD*, 2002.

[11] B. Hore, S. Mehrotra, and G. Tsudik. A privacy-preserving index for range queries. In *30th VLDB*, 2004.

[12] S. Jajodia and R. Sandhu. Toward a multilevel secure relational data model. In *ACM SIGMOD*, 1991.

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

[14] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. *ACM Transactions on Information and System Security (TISSEC)*, 1(1), 1998.

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

[16] J. Vaidya and C. Clifton. Privacy-preserving top-k queries. In *ICDE*, 2005.

[17] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *ACM SIGMOD Record*, 33(1), 2004.

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

[19] L. Xiao, Z. Xu, and X. Zhang. Mutual anonymity protocols for hybrid peer-to-peer systems. In *ICDCS*, 2003.

[20] L. Xiong, S. Chitti, and L. Liu. Topk queries across multiple private databases. Technical report, Georgia Institute of Technology, College of Computing, 2004.