# Distributed Anonymization: Achieving Privacy for Both Data Subjects and Data Providers

Pawel Jurczyk and Li Xiong

Emory University, Atlanta GA 30322, USA

**Abstract.** There is an increasing need for sharing data repositories containing personal information across multiple distributed and private databases. However, such data sharing is subject to constraints imposed by privacy of individuals or data subjects as well as data confidentiality of institutions or data providers. Concretely, given a query spanning multiple databases, query results should not contain individually identifiable information. In addition, institutions should not reveal their databases to each other apart from the query results. In this paper, we develop a set of decentralized protocols that enable data sharing for horizontally partitioned databases given these constraints. Our approach includes a new notion, *l-site-diversity*, for data anonymization to ensure anonymity of data providers in addition to that of data subjects and a distributed anonymization protocol that allows independent data providers to build a virtual anonymized database while maintaining privacy for both data subjects and data providers.

## 1 Introduction

Current information technology enables many organizations to collect, store, and use various types of information about individuals in large repositories. Government and organizations increasingly recognize the critical value and opportunities in sharing such a wealth of information across multiple distributed databases.

**Problem scenario**. An example scenario is the Shared Pathology Informatics Network (SPIN)[1] initiative by the National Cancer Institute. The objective is to establish an Internet-based *virtual* database that will allow investigators access to data that describe archived tissue specimens across multiple institutions while still allowing those institutions to maintain local control of the data. There are some important privacy considerations in such a scenario. First, personal health information is protected under the Health Insurance Portability and Accountability Act (HIPAA)[2][3] and cannot be revealed without de-identification

[1] Shared Pathology Informatics Network. http://www.cancerdiagnosis.nci.nih.gov/spin/

[2] Health Insurance Portability and Accountability Act (HIPAA). http://www.hhs.gov/ocr/hipaa/.

[3] State law or institutional policy may differ from the HIPAA standard and should be considered as well.

or anonymization. In addition, institutions can not reveal their private databases to each other due to confidentiality of the data. In addition, they may not want to reveal the ownership of their records even if the records are anonymized.

These scenarios can be generalized into the problem of privacy-preserving data publishing for multiple distributed databases where multiple data custodians or providers wish to publish an integrated view of the data for querying purposes while preserving privacy for both *data subjects* and *data providers*. We consider two privacy constraints in the problem. The first is the privacy of individuals or *data subjects* (such as the patients) which requires that the published view of the data should not contain individually identifiable information. The second is the privacy of *data providers* (such as the institutions) which requires that data providers should not reveal their private data or the ownership of the data to each other besides the published view.

**Existing and potential solutions**. Privacy preserving data publishing or data anonymization for a single database has been extensively studied in recent years. A large body of work contributes to algorithms that transform a dataset to meet a privacy principle such as $k$-anonymity using techniques such as generalization, suppression (removal), permutation and swapping of certain data values so that it does not contain individually identifiable information [6].

There are a number of potential approaches one may apply to enable data anonymization for distributed databases. A naive approach is for each data provider to perform data anonymization independently as shown in Fig. 1a. Data recipients or clients can then query the individual anonymized databases or an integrated view of them. One main drawback of this approach is that data is anonymized before the integration and hence will cause the data utility to suffer. In addition, individual databases reveal their ownership of the anonymized data.

An alternative approach assumes an existence of third party that can be trusted by each of the data owners as shown in Fig. 1b. In this scenario, data owners send their data to the trusted third party where data integration and anonymization are performed. Then, clients can query the centralized database. However, finding such a trusted third party is not always feasible. Compromise

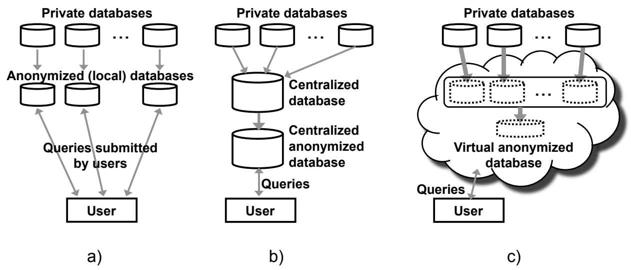
Private databases
Anonymized (local) databases
Queries submitted by users
User
a)
Private databases
Centralized database
Centralized anonymized database
Queries
User
b)
Private databases
Virtual anonymized database
Queries
User
c)

**Fig. 1.** Architectures for privacy preserving data publishing

of the server by hackers could lead to a complete privacy loss for all participating parties and data subjects.

In this paper, we propose a distributed data anonymization approach as illustrated in Fig. 1c. In this approach, data providers participate in distributed protocols to produce a *virtual* integrated and anonymized database. Important to note is that the anonymized data still resides at individual databases and the integration and anonymization of the data is performed through the secure distributed protocols. The local anonymized datasets can simply be unioned using secure union algorithms [11, 2] and then published or can alternatively serve as a virtual database that can be queried. In that case, when users query the virtual database, each individual database executes the query on its local anonymized dataset, and then engage in the distributed secure union protocol to assemble the results that are guaranteed to be anonymous.

**Contributions.** We study the problem of data anonymization for horizontally partitioned databases in this paper and present the distributed anonymization approach for the problem. Our approach consists of two main contributions.

First, we propose a *distributed anonymization protocol* that allows multiple data providers with horizontally partitioned databases to build a virtual anonymized database based on the integration (or union) of the data, and a *distributed querying protocol* that allows clients to query the virtual database. As the output of the distributed anonymization protocol, each database produces a local anonymized dataset and their union forms a virtual database that is guaranteed to be anonymous based on an anonymization principle. The protocol utilizes *secure multi-party computation* protocols for sub-operations such that information disclosure between individual databases is minimal during the virtual database construction and query answering.

Second, we propose a new notion, *l-site-diversity*, to ensure anonymity of data providers in addition to that of data subjects for anonymized data. We present heuristics and adapt existing anonymization algorithms for $l-site-diversity$ so that anonymized data achieve better utility.

**Organization.** The remainder of this paper is organized as follows. Section 2 briefly reviews work related to our research. Section 3 discusses the privacy model we are using and presents our new notion on $l-site-diversity$. Section 4 presents our distributed anonymization approach including the distributed anonymization protocol and distributed querying protocol. Section 5 presents a set of experimental evaluations and Section 6 concludes the paper.

## 2 Related work

Our work is inspired and informed by a number of areas. We briefly review the closely related areas below and discuss how our work leverages and advances the current state-of-the-art.

**Privacy preserving data publishing.** Privacy preserving data publishing for centralized databases has been studied extensively [6]. One thread of work aims at devising privacy principles, such as $k$-anonymity, $l$-diversity, $t$-closeness, and $m$-invariance, that serve as criteria for judging whether a published dataset provides sufficient privacy protection. Another large body of work contributes to algorithms that transforms a dataset to meet one of the above privacy principles (dominantly $k$-anonymity). In this study, our distributed anonymization protocol is built on top of the $k$-anonymity and $l$-diversity principle and the greedy top-down Mondrian multidimensional $k$-anonymization algorithm [12].

There are some works focused on data anonymization of distributed databases. [9] presented a two-party framework along with an application that generates $k$-anonymous data from two vertically partitioned sources without disclosing data from one site to the other. [20] proposed provably private solutions for $k$-anonymization in the distributed scenario by maintaining end-to-end privacy from the original customer data to the final $k$-anonymous results.

In contrast to the above work, our work is aimed at horizontal data distribution and arbitrary number of sites. More importantly, our anonymization protocol aims to achieve anonymity for both data subjects and data providers.

**Secure multi-party computation.** Our approach also has its roots in the secure multi-party computation (SMC) problem [7, 4, 13, 18, 5]. In SMC, a given number of participants, each having a private data, wants to compute the value of a public function. An SMC protocol is *secure* if no participant can learn more from the description of the public function and the result of function.

Our problem can be viewed as designing secure SMC protocols for anonymization (building virtual anonymized database) and query processing (assembling query results). Our distributed anonymization protocol utilizes existing secure SMC protocols for subroutines such as computing sum [16], $k$th element [1] and set union [11, 2]. Our protocol is carefully designed so that intermediate information disclosure can be minimized.

## 3 Privacy Model

In this section we present the privacy goals that we focus on in this paper, followed by models and metrics for characterizing how these goals are achieved, and propose a new notion for protecting anonymity for data providers. As we identified in Section 1, we have two privacy goals. First, the privacy of individuals or data subjects needs be protected, i.e. the published virtual database and query results should not contain individually identifiable information. Second, the privacy of data providers needs to be protected, i.e. individual databases should not reveal their data or their ownership of the data apart from the query results of the published virtual database.

**Privacy for data subjects based on anonymity**. Among the many privacy principles that protect against individual identifiability, the seminal work on $k$-anonymity [15, 17] requires that a set of $k$ records (entities) to be indistinguishable from each other based on a quasi-identifier set. Given a relational table $T$, attributes are characterized into: *unique identifiers* which identify individuals; *quasi-identifier (QID)* which is a minimal set of attributes ($X_1$, ..., $X_d$) that can be joined with external information to re-identify individual records; and *sensitive attributes* that should be protected. The set of all tuples containing identical values for the QID set is referred to as an *equivalence class*. An improved principle, $l$-diversity [14], demands every group to contain at least $l$ well-represented sensitive values.

Given our research goals of extending the anonymization techniques and integrating them with secure computation techniques to preserve privacy for both data subjects and data providers, we based our work on $k$-anonymity and $l$-diversity to achieve anonymity for data subjects. While we realize they are relatively weak compared to principles such as differential privacy, the reason we chose them for this paper is that they are intuitive and have been justified to be useful in many practical applications such as privacy-preserving location services, therefore, techniques enforcing them in the distributed environment will still be practically important. In addition, their fundamental concepts serve as basis for many other principles and there is a rich set of algorithms for achieving $k$-anonymity and $l$-diversity. We can study the subtle differences and effects of different algorithms and their interactions with secure multi-party computation protocols. Finally, our protocol structure, and the underlying concepts will be orthogonal to these privacy principles, and our framework will be extensible so as to easily incorporate more advanced privacy principles.

**Privacy for data providers based on secure multi-party computation.** Our second privacy goal is to protect privacy for data providers. It resembles the goal of secure multi-party computation (SMC). In secure SMC, a protocol is *secure* if no participant can learn more from the description of the public function and the result of function. It is important to note that for practical purposes, we may relax the security goal for a tradeoff for efficiency. Instead of attempting to guarantee absolute security in which individual databases reveal nothing about their data apart from the query results of the anonymized virtual database, we wish to minimize data exposure and achieve a sufficient level of security.

We also adopt the *semi-honest* adversary model commonly used in secure SMC problems. A semi-honest party follows the rules of the protocol, but it can attempt to learn additional information about other nodes by analyzing the data received during the execution of the protocol. The *semi-honest* model is realistic for our problem scenario where multiple organizations are collaborating with each other to share data and will follow the agreed protocol to get the correct result for their mutual benefit.

**Privacy for data providers based on anonymity: a new notion.** Now we show that by simply coupling the above anonymization principles and the multi-party secure computation principles are insufficient in our scenario. While the secure multi-party computation can be used for the anonymization to preserve privacy for data providers during anonymization, the anonymized data itself (considered as results of the secure computation) may compromise the privacy of data providers. The data partitioning at distributed data sources and certain background knowledge can introduce possible attacks that may reveal the ownership of certain data by certain data providers. We illustrate such an attack, a homogeneity attack, through a simple example.

Table 1 shows anonymized data that satisfies 2-anonymity and 2-diversity at two distributed data providers (QID: City, Age; sensitive attribute: Disease).

Even if secure SMC protocols are used to answer queries, given some background knowledge on data partitioning, the ownership of records may be revealed. For instance, if it is known that records from *New York* are provided only by *node 0*, then records with ID 1 and 2 can be linked to that node directly. In consequence, privacy of data providers is compromised. Essentially, the compromise is due to the anonymized data and cannot be solved by multi-party secure computation. One way to fix the problem is to generalize the location for record 1 and 2 so that they cannot be directly linked to a particular data provider.

**Table 1.** Illustration of Homogeneity Attack for Data Providers

| **ID** | **City** | **Age** | **Disease** |
|---|---|---|---|
| 1 | New York | 30-40 | Heart attack |
| 2 | New York | 30-40 | AIDS |

Node 0

| **ID** | **City** | **Age** | **Disease** |
|---|---|---|---|
| 3 | Northeast | 40-43 | AIDS |
| 4 | Northeast | 40-43 | Flu |

Node 1

To address such a problem, we propose a new notion, *l-site-diversity*, to enhance privacy protection for data providers. We define a *quasi-identifier set* with respect to data providers as a minimal set of attributes that can be used with external information to identify the ownership of certain records. For example, the location is a QID in the above scenario as it can be used to identify the ownership of the records based on the knowledge that certain providers are responsible for patients from certain locations. The parameter, $l$, specifies minimal number of distinct sites that records in each equivalence class belong to. This notion protects the anonymity of data providers in that each record can be linked to at least $l$ providers. Formally, the table $T^*$ satisfies *l-site-diversity* if for every equivalence class $g$ in $T^*$ the following condition holds:

$$count(distinct\ nodes(g)) \geq l \tag{1}$$

where *nodes(g)* returns node IDs for every record in group $g$.

It can be noted that our definition of *l-site-diversity* is closely related to *l-diversity*. The two notions, however, have some subtle differences. First the $l - site - diversity$ is related to patients and, provided that the information on origin of the record is treated as a sensitive attribute of patient, it protects the patients. Second, it also protects the ownership anonymity for data providers guaranteeing that multiple data providers contribute records to each equivalence class. The QID set for data providers could be completely different from the QID set for data subjects. *l-site-diversity* is only relevant when there are multiple data sources and it adds another check when data is being anonymized so that the resulting data will not reveal the ownership of the data. It is worth mentioning that we could also exploit much stronger definitions of l-diversity such as *entropy l-diversity* or *recursive (c,l)-diversity* as defined in [14].

## 4 Distributed Anonymization Protocol

In this section we describe our distributed anonymization approach. We first describe the general protocol structure, then present the distributed anonymization protocol, followed by the distributed querying protocol.

We assume that the data are partitioned horizontally among n sites ($n > 2$) and each site owns a private database $d_i$. The union of all local databases, denoted $d$, gives a complete view of all data ($d = \bigcup d_i$). In addition, the *quasi-identifier* of each local database is uniform among all the sites. The sites engage in a distributed anonymization protocol where each site produces a local anonymized dataset $a_i$ and their union forms a virtual database that is guaranteed to be $k$-anonymous. Note that $a_i$ is not required to be $k$-anonymous by itself. When users query the virtual database, each individual database executes the query on $a_i$ and then engage in a distributed querying protocol to assemble the results that are guaranteed to be $k$-anonymous.

### 4.1 Selection of anonymization algorithm

Given our privacy models, we need to carefully adapt or design new anonymization algorithms with additional check for site-diversity and implement the algorithm using multi-party distributed protocols. Given a centralized version of anonymization algorithm, we can decompose it and utilize secure SMC protocols for sub-routines which are provably secure in order to build a secure distributed anonymization protocol. However, performing one secure computation, and using those results to perform another, may reveal intermediate information that are not part of the final results even if each step is secure. An important consideration for designing such protocols is to minimize the disclosure of intermediate information.

There are a large number of algorithms proposed to achieve $k$-anonymity. These $k$-anonymity algorithms can be also easily extended to support *l-diversity* check [14]. However, given our design goal above, not all anonymization algorithms are equally suitable for secure multi-party computation. Considering the two main strategies, top-down partitioning and bottom-up generalization, we discovered that top-down partitioning approaches have significant advantages over bottom-up generalization ones in multi-party computation setting because anything revealed during the protocol as intermediate results will in fact have a coarser view than the final result.

Based on the rationale above, our distributed anonymization protocol is based on the multi-dimensional top-down Mondrian algorithm [12]. The Mondrian algorithm uses a greedy top-down approach to recursively partition the (multidimensional) quasi-identifer domain space. It recursively chooses the split attribute with the largest normalized range of values, and (for continuous or ordinal attributes) partitions the data around the median value of the split attribute. This process is repeated until no allowable split remains, meaning that the data points in a particular region cannot be divided without violating the anonymity constraint, or constraints imposed by value generalization hierarchies.

### 4.2 Distributed anonymization protocol

The key idea for the distributed anonymization protocol is to use a set of secure multi-party computation protocols to realize the Mondrian method for the

**Algorithm 1** Distributed anonymization algorithm - leading site ($i = 0$)

```
1: function split(set $d_0$, ranges of QID attributes)
2: Phase 1: Determine split attribute and split point
3: Select best split attribute $a$ (see text)
4: If split is possible, send split attribute to node 1. Otherwise, send finish splitting to
   node 1 and finish.
5: Compute median of chosen $a$ for splitting (using secure k-th element algorithm).
6: Phase 2: Split current dataset
7: Send $a$ and $m$ to node 1
8: Split set $d_0$, create two sets, $s_0$ containing items smaller than $m$ and $g_0$ containing
   items greater than $m$. Distribute median items among $s_i$ and $g_i$.
9: Send finished to node 1
10: Wait for finished from last node (synchronization)
11: Phase 3: Recursively split sub datasets
12: Find $size_{left} = |\bigcup s_i|$ and $size_{right} = |\bigcup g_i|$ (using secure sum protocol)
13: If further split of left (right) subgroup is possible, send split_left=true
    (split_right=true) to node 1 and call the split function recursively (updating ranges
    of QID attributes). Otherwise send split_left=false (split_right=false) to node 1.
14: end function split
```

distributed setting so that each database produces a local anonymized dataset which may not be $k$-anonymous itself, but their union forms a virtual database that is guaranteed to be $k$-anonymous. We present the main protocol first, followed by important heuristics that is used in the protocol.

We assume a leading site is selected for the protocol. The protocols for the leading and other sites are presented in Algorithm 1 and 2. The steps performed at the leading site are similar to the centralized Mondrian method. Before the computation starts, range of values for each quasi-identifier in set $d = \bigcup d_i$ need to be calculated and the total number of data points. A *secure kth element* protocol can be used to securely compute the minimum ($k$=1) and maximum ($k = n$ where $n$ is the total number of tuples in the current partition) value of each attribute across the databases [1]. Note that the total number of data points is already computed before the split function is called (see next paragraph).

In Phase 1, the leading site selects the best split attribute and determines the split point for splitting the current partition. In order to select the best split attribute, the leading site uses a heuristic rule that is described in details below. If required, all the potential split attributes (e.g., the attributes that produce subgroups satisfying $l-site-diversity$) are evaluated and the best one is chosen. In order to determine the split medians, a *secure kth element* protocol is used ($k = \lceil \frac{n}{2} \rceil$) with respect to the data across the databases. To test whether given attribute can be used for splitting, we calculate a number of distinct sites in subgroups resulting from splitting on this attribute using the secure sum algorithm. The split is possible, if records in both subgroups are provided by at least $l$ sites. In Phase 2, the algorithm performs split and waits for all the nodes to finish splitting. Finally in Phase 3, the node recursively checks whether

**Algorithm 2** Distributed anonymization algorithm - non-leading node ($i > 0$)

```
1: function split(set c)
2:   Read split attribute $a$ and median $m$ from node $(i-1)$; pass them to node $(i+1)$
3:   if finish splitting received then return
4:   Split set $c$ into $s_i$ containing items *smaller than* $m$ and $g_i$ containing items
     *greater than* $m$. Distribute median items among $s_i$ and $g_i$.
5:   Read finished from node $i-1$ (sychronization); Send finished to node $i+1$
6:   Read *split_left* from node $i-1$ and pass it to node $i+1$
7:   if split_left then call split($s_i$)
8:   Read *split_right* from node $i-1$, Send *split_right* to node $i+1$
9:   if split_right then call split($g_i$)
10: end function split
```

| original data | | | anonymized data | | |
|---|---|---|---|---|---|
| **ID** | **ZIP** | **Age** | **ID** | **ZIP** | **Age** |
| 1 | 30030 | 31 | 1 | 30030-36 | 31-32 |
| 2 | 30033 | 32 | 2 | 30030-36 | 31-32 |

node 0

| original data | | | anonymized data | | |
|---|---|---|---|---|---|
| **ID** | **ZIP** | **Age** | **ID** | **ZIP** | **Age** |
| 3 | 30045 | 45 | 3 | 30037-56 | 32-45 |
| 4 | 30056 | 32 | 4 | 30037-56 | 32-45 |

node 1

| **ID** | **ZIP** | **Age** | **ID** | **ZIP** | **Age** |
|---|---|---|---|---|---|
| 5 | 30030 | 22 | 5 | 30030-36 | 22-30 |
| 6 | 30053 | 22 | 6 | 30037-56 | 22-31 |

node 2

| **ID** | **ZIP** | **Age** | **ID** | **ZIP** | **Age** |
|---|---|---|---|---|---|
| 7 | 30038 | 31 | 7 | 30037-56 | 22-31 |
| 8 | 30033 | 30 | 8 | 30030-36 | 22-30 |

node 3

**Fig. 2.** Distributed anonymization illustration

further split of new subsets is possible. In order to determine whether a partition can be further split, a *secure sum* protocol [16] is used to securely compute the total number of tuples of the partition across the databases.

We illustrate the overall protocol with an example scenario shown in Figure 2 where we have 4 nodes and we use $k = 2$ for $k$-anonymization and $l = 1$ for *l-site-diversity.* Note that the anonymized databases at node 2 and node 3 are not 2-anonymous by themselves. However the union of all the anonymized databases is guaranteed to be 2-anonymous.

**Selection of split attribute**. One key issue in the above protocol is the selection of split attribute. The goal is to split the data as much as possible while satisfying the privacy constraints so as to maximize discernibility or utility of anonymized data. The basic Mondrian method uses the range of an attribute as a goodness indicator. Intuitively, the larger the spread, the easier the good split point can be found and more likely the data can be further split. In our setting, we also need to take into account the site diversity requirement and adapt the selection heuristic. The importance of doing so is demonstrated in Figure 3. Let's assume that we want to achieve 2-anonymity and 2-site-diversity. In the first scenario, the attribute for splitting is chosen only based on range of attribute. The protocol finishes with 2 groups of 5 and 4 records (further split is impossible due to 2-site-diversity requirement). The second scenario exploits information on records distribution when the decision on split attribute is made (the more evenly records are distributed across sites in resulting subgroups, the better). This rule yields better results, namely three groups of 3 records each.

Based on the illustration, intuition suggests that we need to select a split attribute that results in partitions with even distribution of records from dif-

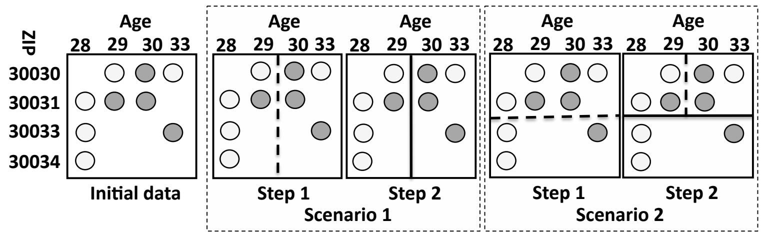

**Fig. 3.** Impact of split attribute selection when *l-site-diversity* ($n = 2$) is considered. Different shades represent different owners of records.

ferent data providers. This makes further split more likely while meeting the $l$-site-diversity constraint. Similar to decision tree classifier construction [8], information gain can be used as a scoring metric for selecting attribute that results in partitions with most diverse distribution of data providers. Note that this is used in the complete opposite sense from decision tree where the goal is to partition the data into partitions with homogeneous classes. The information gain of a potential splitting attribute $a_k$ is computed through the information entropy of resulting partitions:

$$e(a_k) = -\sum_{i=0}^{n-1} p_{(i,l_k)} log(p_{(i,l_k)}) - \sum_{i=0}^{n-1} p_{(i,r_k)} log(p_{(i,r_k)}) \qquad (2)$$

where $l_k$ and $r_k$ are partitions created after splitting the input set using attribute $a_k$ (and its median value) and $p(i, g)$ is the portion of records that belong to node $i$ in group $g$. It is important to note that the calculations need to take into account data on distributed sites and thus secure SMC protocol need to be used (e.g. secure sum).

Our final scoring metric combines the original range value based metric and the new diversity-aware metrics using a linear combination as follows:

$$\forall_{a_i \in Q} s_i = \alpha \frac{range(a_i)}{\max\limits_{a_j \in Q}(range(a_j))} + (1-\alpha)\frac{e(a_i)}{\max\limits_{a_j \in Q}(e(a_j))} \qquad (3)$$

where *range* function returns range of attribute, $e(a_i)$ returns values of information entropy as defined above when attribute $a_i$ is used for splitting and $\alpha$ is a weighting parameter.

Important to note is that if *l-site-diversity* is not required (e.g., *l=1*), then the evaluation of the heuristic rule above is limited to checking only the range of attributes, and choosing the attribute with the widest range.

## 4.3 Analysis

Having presented the distributed anonymization protocol, we analyze the protocol in terms of its security and overhead.

**Security**. We will now analyze the security of our distributed k-anonymity protocol. Before we get started, we will first define notation we will use. We will assume that there are $p$ attributes in QID ($P_1 ... P_p$). We will also assume that the final result of the distributed anonymization is denoted $T^*$. $T^*$ is a table where each record has the following form:

$$R_1, R_2, ..., R_p, s$$

with $R_i$ being range of attribute $P_i$ and s being secure attribute. Each range $R_i$ has a form $[r_i^F - r_i^T]$.

Our proof will show that, given the result, the leaked information (if any), and site's own input, any site can simulate the protocol and everything that was seen during the execution. Since the simulation generates everything seen during execution of the protocol, clearly no one learns anything new from the protocol when it is executed.

Our proof will also use a general composition theorem [7] that covers algorithms implemented by running many invocation of secure computations of simpler functionalities. Note that in our case the simpler functionalities are secure sum and secure $k - th$ element calculation. Let's assume a *hybrid model* where the protocol uses a trusted third-party to compute the result of such smaller functionalities $f_1 ... f_n$. The composition theorem states that if a protocol in hybrid model is secure in terms of comparing the real computation to the ideal model, then if a protocol is changed in such a way that calls to trusted third-party are replaced with secure protocols, the resulting protocol is still secure.

We have analyzed the distributed k-anonymity protocol in terms of security and present the following two theorems.

**Theorem 1**. Distributed k-anonymity protocol privately computes k-anonymous view of horizontally partitioned data in semi-honest model when $l = 1$.

**Proof**. Let's first consider a computation in a hybrid model. We will show that any given node can simulate the algorithm and all what was seen during its execution given only the final result and its local data. As the distributed k-anonymity algorithm is recursive, to efficiently simulate the execution of the algorithm node will use a special data structure that will help to identify group of records being considered at every step of recursion. The data structure will have the following form: ($P_i \rightarrow \textit{relevant range}_i$) and can be understood as constraint on values of attributes from QID. The initial values for relevant ranges can be simply identified by scanning the table $T^*$, and setting the ranges to the following values:

$$\textit{relevant range}_i = [\min_{t \in T^*}(r_i^F), \max_{t \in T^*}(r_i^T)]$$

At this point recursive algorithm starts. The arguments of the algorithm are relevant ranges for all the QID attributes, and the initial relevant ranges computed above are used for the first time. The algorithm first analyzes relevant ranges to identify the attribute $P_i$ that was used for splitting. As no l-site-diversity is required, the attribute that was used for splitting is actually the attribute with the largest relevant range. Let's assume that attribute $P_j$ has the widest range. Now the site knows that this attribute was used for splitting. The next step requires identification of the split point. It turns out that such a point of splitting can be easily identified. First, using the relevant ranges the site identifies a set of relevant records from $T^*$. Relevant record is a record that has all the ranges overlapping with the current relevant ranges:

$$F = \{(R_1, ..., R_p, s) \in T^* : \forall_{P_i} R_i \in \mathit{relevant\ range}_i\}$$

Now the node analyzes all relevant records. The next step is to identify all possible splitting points that could be used when the algorithm was executed. Note that as the same QID attribute could be used for splitting on different steps of the recursion, there can be few possible points of splitting. Formally the points of potential splits are distinct values of $r_j^T$ that appear in the set of relevant items $F$ (assuming that $P_j$ was the attribute with the largest range). To identify the median value (or the value that was used for split) the node checks which of the potential splitting points is actually a median. This can be easily done by choosing value that divides the set of relevant records into two sets with the sizes closest to $|F|/2$ (note that the subsets resulting from spitting might not have equal sizes - for instance if number of records is odd).

With the splitting attribute and point identified, the site is ready to simulate the split. If size of any of the two groups that result from splitting is greater than or equal to $2 * k$, this group can be further split. In such a case the node updates relevant ranges for that group and calls the recursive function.

Since we showed that the execution of the protocol in a hybrid model can be efficiently simulated by any node with only knowledge of the final result (we even did not have to use site's local dataset), such an execution is secure. From composition theorem follows that if we change the hybrid model and replace calls to trusted third-party with secure algorithms, the resulting protocol will still be secure. This finishes the proof.

**Theorem 2**. Distributed k-anonymity algorithm privately computes k-anonymous view of horizontally partitioned data in semi-honest model and when $l > 1$ it reveals at most statistics of the data that include:

1. Median values of each attribute from QID for groups of records of size $\geq 2*k$,
2. Entropy of distribution of records for groups resulting from potential splits,
3. Number of distinct sites that provide data to groups resulting from potential splits, keeping the identity of those sites confidential.

**Proof**. Let's consider a computation in a hybrid model. We will show that any given node can simulate the the algorithm given only the final result, its local data and the statistical data that is revealed in points 1, 2 and 3 defined above. The proof strictly follows the proof of Theorem 1. The same data structure is used to help the node identify a group of records being considered at every step. The recursive algorithm is also implemented in the same way as above. The only difference is the decision step when a node decides on the split attribute. This time, not only the range of attribute has to be considered, but also the distribution of records in groups resulting from splitting. Using the final result, information in points 1, 2 and 3, and the current relevant ranges of QID attributes, any node can decide on split attribute in the following way. First, the node identifies set of relevant records:

$$F = \{(R_1, ..., R_p, s) \in T^* : \forall_{P_i} R_i \in relevant\ range_i\}$$

Using the information from point 2 and the relevant ranges, the node can now calculate score defined in equation 3 for each attribute $P_i$. Next, using the information from points 1 and 3 the node can decide which of the attributes $P_i$ can be used for splitting (first node uses knowledge from point 1 to find the ranges of attributes of groups resulting from potential split and then the node uses knowledge from point 3 to decide whether such a split is possible). Finally, the node chooses an attribute with the largest possible score that satisfies $l{-}site{-}diversity$. Once the attribute is chosen, the node can continue simulation in the same way as in the proof of Theorem 1.

We again showed that the execution of the protocol in a hybrid model can be efficiently simulated by any node with only knowledge of the final result and the revealed information. Therefore such an execution is secure. From the composition theorem follows that if we change the hybrid model and replace calls to trusted third-party with secure algorithms, the resulting protocol will still be secure.

**Overhead.** Our protocol introduces additional overhead due to the fact that the nodes have to use additional protocols in each step of computation. The time complexity of the original Mondrian algorithm is $O(nlogn)$ where $n$ is the number of items in the anonymized dataset [12]. As we presented in Algorithm 1, each iteration of the distributed anonymization algorithm requires calculation of the heuristic decision rule, median value of an attribute, and the count of tuples of a partition. The secure sum protocol does not depend on the number of tuples in the database. The secure $k - th$ element algorithm is logarithmic in number of input items (assuming the worst - case scenario that all the input items are distinct). As a consequence, the time complexity of our protocol can be estimated as $O(nlog^2n)$ in terms of number of records in a database.

The communication overhead of the protocol is determined by two factors. The first is the cost for a single round. This depends on the number of nodes involved in the system and the topology which is used and in our case it is proportional to the number of nodes on the ring. As the future work, we are considering alternative topologies (such as trees) in order to optimize the communication cost for each round. The second factor is the number of rounds and is determined by the number of iterations and the sub-protocols used by each iteration of the anonymization protocol. The secure sum protocol involves one round of communication. In the secure $k$th element protocol, the number of rounds is $logM$ ($M$ being the range of attribute values) and each round requires secure computations twice. It is important to note that the distributed anonymization protocol is expected to be run offline on an infrequent basis. As a result, the overhead of the protocol will not be a major issue.

## 5 Experimental evaluation

We have implemented the distributed anonymization protocol in Java within the DObjects framework [10] which provides a platform for querying data across distributed and heterogeneous data sources. To be able to test a large variety of configurations, we also implemented the distributed anonymization protocol using a simulation environment. In this section we present a set of experimental evaluations of the proposed protocols.

The questions we attempt to answer are: 1) What is the advantage of using distributed anonymization algorithm over centralized or independent anonymization? 2) What is the impact of the *l-site-diversity* constraint on anonymization protocol? 3) What are the optimal values for $\alpha$ parameter in our heuristic rules presented in equation 3?

### 5.1 Distributed Anonymization vs. Centralized and Independent Anonymization

We first present an evaluation of the distributed anonymization protocol compared to the centralized and independent anonymization approaches in terms of the quality of the anonymized data.

**Dataset and setup.** We used the Adult dataset from UC Irvine Machine Learning Repository. The dataset contained 30161 records and was configured as in [12]. We used 3 distributed nodes (30161 records were split among those nodes using round-robin protocol). We report results for the following scenarios: 1) the data is located in one centralized database and classical Mondrian k-anonymity algorithm was run (centralized approach), 2) data are distributed among the three nodes and Mondrian k-anonymity algorithm was run at each site independently (independent or naive approach) and 3) data are distributed among the three nodes and we use the distributed anonymization approach presented in section 4. We ran each experiment for different $k$ values. All the experiments in this subsection used *1-site-diversity*.

**Results.** Figure 4 shows the average equivalence class size with respect to different values of $k$. We observe that our distributed anonymization protocol performs the same as the non-distributed version. Also as expected, the naive approach (independent anonymization of each local database) suffers in data utility because the anonymization is performed before the integration of the data.

### 5.2 Achieving Anonymity for Data Providers

The experiments in this section again use the Adult dataset. The data is distributed across $n = 100$ sites unless otherwise specified. We experimented with distribution pattern that we will describe in detail below.

**Metric**. The average equivalence group size as shown in previous subsection provides a general data utility metric. The query imprecision metric provides an application-specific metric that is of particular relevance to our problem setting.

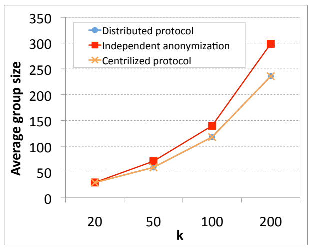

**Fig. 4.** Average equivalence class size vs. $k$

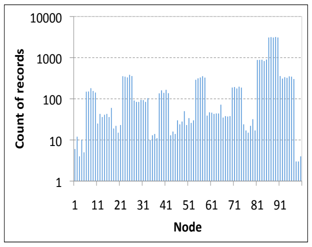

**Fig. 5.** Histogram for partitioning using City and Age

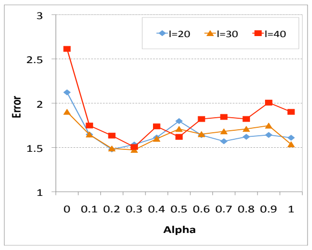

**Fig. 6.** Average error vs. $\alpha$ (information gain-based)

Given a query, since the attribute values are generalized, it is possible only to return the tuples from the anonymized dataset that are contained in any generalized ranges overlapping with selection predicate. This will often produce a larger result set than evaluating the predicate over the original table. For this set of experiments, we use summary queries (queries that return count of records) and we use an algorithm similar to the approach introduced in [19] that returns more accurate results. We report *relative error* of the query results. Specifically, given *act* as an exact answer to query and *est* as an answer computed according to algorithm defined above, the relative error is defined as $|act - est|/act$. For each of the tested configurations, we submit 10,000 randomly generated queries, and for each query we calculate relative error. We report average value of the error. Each query uses predicates on two randomly chosen attributes from *quasi-identifier*. For boolean attributes that can have only two values (e.g. sex), the predicate has a form of $a_i = value$. For other attributes we use predicate in the form $a_i \in R$. $R$ is a random range and has a length of $0.3 * |a_i|$, where $|a_i|$ denotes the domain size of attribute.

**Data partitioning**. In realistic scenario, data is often split according to some attributes. For instance, consider the patient data scenario, one can assume a partitioning based on the city attribute where the majority records from a hospital located in New York would have a New York address while those from a hospital located in Boston would have a Boston address. Therefore, to test a realistic scenario, we distributed records across sites using partitioning based on attribute values. The rules of partitioning were specified using two attributes, City and Age. The dataset contained data from 6 different cities, and every 1/6th of available nodes were assigned to a different city. Next, records within each group of nodes responsible for a given city were distributed using Age attribute by the following rule: records with age less than 25 were assigned to the first 1/3rd of nodes, records with age between 25 and 55 years were assigned to the second 1/3rd of nodes, and the remaining records were assigned to the remaining nodes. The histogram of count of records per node in this setup is presented in Figure 5 (please note the logarithmic scale of the plot).

**Results**. We now present the results evaluating the impact of $\alpha$ value under this setup. Figure 6 present average query error for different $\alpha$ and $l$ values for the heuristic rule we used. We can observe a significant impact of $\alpha$ value on the average error. The smallest average error value is observed for $\alpha = 0.3$ and

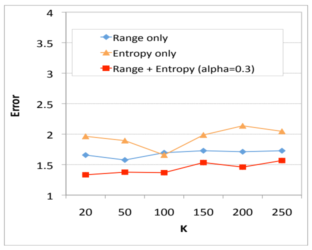

**Fig. 7.** Average error vs. $k$ ($l = 30$)

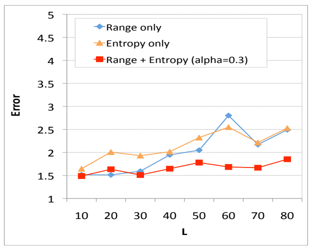

**Fig. 8.** Average error vs. $l$ ($k = 200$)

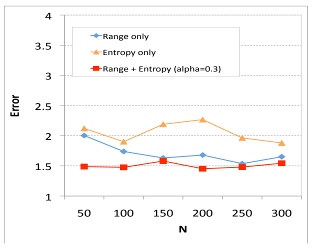

**Fig. 9.** Error vs. $n$ ($k = 200$ and $l = 30$)

this seems to be an optimal choice for all tested $l$ values. One can observe 30% decrease in error when compared to using only range as in original Mondrian ($\alpha = 1.0$) or using only diversity-aware metrics ($\alpha = 0.0$). It is worth of mentioning that we have also experimented with different distributions of records, and the results were consistent with what we presented above. We do not provide these results, however, due to space limitations.

The next experiment was focused on the impact of $k$ parameter on average error. We present results for $l$=*30* in Figure 7. The plot presents the results for three different split heuristic rules: using range only, information gain only, and combining range with information gain with $\alpha = 0.3$. We observe that the heuristic rule that takes into account range and information gain gives consistently the best results and a reduction of error around 30%. These results do not depend on the value of $k$.

Next, we tested the impact of the $l$ parameter for $l - site - diversity$. The Figure 8 shows average error for varying $l$ and $k = 200$ using the same three split heuristic rules as in previous experiment. Similarly to the result above, the rule that takes into account range and information gain gives the best results. With increasing $l$ we observe an increasing error rate because the data needs to be more generalized in order to satisfy the diversity constraints.

So far we have tested only scenarios with 100 nodes ($n = 100$). To complete the picture in Figure 9, we plot the average size of equivalence class with varying $n$, $k = 200$ and $l = 30$. One can notice that previous trends are maintained - the rules do not appear to be dependent on the number of nodes in the system. Also similar to above, the rule that takes into account range and information gain is superior to other methods and the query error is in average 30% smaller than that for others.

## 6 Conclusion

We have presented a distributed and decentralized anonymization approach for privacy-preserving data publishing for horizontally partitioned databases. Our work addresses two important issues, namely privacy of data subjects together with privacy of data providers. We presented a new notion, *l-site-diversity*, to achieve anonymity for data providers in anonymized dataset.

Our work continues along several directions. First, we are interested in developing a protocol toolkit incorporating more privacy principles and anonymization algorithms. In particular, an important point on our future research agenda is to support serial releases of data with data updates. Such concepts as *m-invariance* [19] or *l-scarsity* [3] are promising ideas and we plan to work on extending our research in this direction. Second, we are also interested in developing specialized multi-party protocols such as set union that offers a tradeoff between efficiency and privacy. Our current set union protocol is based on cryptographic approaches that are computationally expensive. We plan to look into different approaches such as algorithms based on probabilistic concepts.

## References

1. G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the $k$th-ranked element. In *In Avdances in Cryptology - Proc. of Eurocyrpt 04*, pages 40–55. Springer-Verlag, 2004.
2. S. Böttcher and S. Obermeier. Secure set union and bag union computation for guaranteeing anonymity of distrustful participants. *JSW*, 3(1):9–17, 2008.
3. Y. Bu, A. W. C. Fu, R. C. W. Wong, L. Chen, and J. Li. Privacy preserving serial data publishing by role composition. *Proc. VLDB Endow.*, 1(1):845–856, 2008.
4. C. Clifton, M. Kantarcioglu, and J. Vaidya. Tools for privacy preserving distributed data mining. *ACM SIGKDD Explorations*, 4:2003, 2003.
5. W. Du and M. J. Atallah. Secure multi-party computation problems and their applications: a review and open problems. In *NSPW '01: Proceedings of the 2001 workshop on New security paradigms*, pages 13–22, New York, NY, USA, 2001. ACM.
6. B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey on recent developments. *ACM Computing Surveys*, in press.
7. O. Goldreich. Secure multi-party computation, 2001. Working Draft, Version 1.3.
8. J. Han and M. Kamber. *Data Mining: Concepts and Techniques.* Morgan Kaufmann, 2nd edition, 2006.
9. W. Jiang and C. Clifton. A secure distributed framework for achieving k-anonymity. *VLDB Journal*, 15(4):316–333, 2006.
10. P. Jurczyk and L. Xiong. Dobjects: Enabling distributed data services for metacomputing platforms. In *Proc. of the ICCS*, 2008.
11. M. Kantarcioglu and C. Clifton. Privacy preserving data mining of association rules on horizontally partitioned data. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(9), 2004.
12. K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proceedings of the International Conference on Data Engineering (ICDE'06)*, 2006.
13. Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. Cryptology ePrint Archive, Report 2008/197, 2008. http://eprint.iacr.org/.
14. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the International Conference on Data Engineering (ICDE'06)*, page 24, 2006.
15. P. Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. Knowl. Data Eng.*, 13(6):1010–1027, 2001.

16. B. Schneier. *Applied Cryptography.* John Wiley & Sons, 2nd edition, 1996.
17. L. Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.
18. J. Vaidya and C. Clifton. Privacy-preserving data mining: Why, how, and when. *IEEE Security & Privacy*, 2(6):19–27, 2004.
19. X. Xiao and Y. Tao. M-invariance: towards privacy preserving re-publication of dynamic datasets. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 689–700, 2007.
20. S. Zhong, Z. Yang, and R. N. Wright. Privacy-enhancing k-anonymization of customer data. In *Proc. of the Principles of Database Systems (PODS)*, 2005.