A Hybrid Approach to Private Record Matching

Ali Inan, Murat Kantarcioglu, *Member*, *IEEE*,
Gabriel Ghinita, *Member*, *IEEE*, and Elisa Bertino, *Fellow*, *IEEE*

Abstract—Real-world entities are not always represented by the same set of features in different data sets. Therefore, matching records of the same real-world entity distributed across these data sets is a challenging task. If the data sets contain private information, the problem becomes even more difficult. Existing solutions to this problem generally follow two approaches: sanitization techniques and cryptographic techniques. We propose a hybrid technique that combines these two approaches and enables users to trade off between privacy, accuracy, and cost. Our main contribution is the use of a *blocking* phase that operates over sanitized data to filter out in a privacy-preserving manner pairs of records that do not satisfy the matching condition. We also provide a formal definition of privacy and prove that the participants of our protocols learn nothing other than their share of the result and what can be inferred from their share of the result, their input and sanitized views of the input data sets (which are considered public information). Our method incurs considerably lower costs than cryptographic techniques and yields significantly more accurate matching results compared to sanitization techniques, even when privacy requirements are high.

Index Terms—Privacy, security, record matching, anonymization, differential privacy.

1 Introduction

Analysis and integration of information maintained by distinct entities is critical for many applications. For instance, two competitor businesses may wish to share information about customers with similar demographics (e.g., age, zip code) to increase their revenues (e.g., to jointly support a location-customized service for young subscribers). However, to protect their customer base, both parties want to keep data that are not part of the join result private. The goal is to match similar records that represent *distinct* individuals, therefore matching based on unique identifiers (e.g., SSN) is not applicable. The problem described in this scenario is known as the *record matching* problem.

Since record matching is a key component of data integration methodologies, it has been investigated extensively [1]. However, especially after the introduction of powerful data mining techniques, privacy concerns related to sharing of individual information have pushed research toward the reformulation of the problem and the development of new solutions [2], [3], [4], [5], [6], therefore introducing the concept of private record matching.

Private matching is a challenging problem, as in many situations uniquely identifying information may not be available, and matching is performed based on attributes like age, occupation, etc. [1]. Furthermore, such information may not always be completely consistent across data sets (e.g., the weight of a patient may vary between two admissions to different hospitals). Therefore, it is important to devise methods that are capable of privately matching records through a *distance-based* condition, rather than simple equi-joins computed using cryptographic hashes [7].

Two main approaches have been proposed for private matching. These are *sanitization* methods that perturb private information to obscure individual identity [8], [9], [10], [11], [12] and *cryptographic* methods that rely on Secure Multi-party Computation (SMC) protocols [13]. Sanitization techniques such as k-anonymization [9] or random noise addition [11], [12], [14] involve a delicate tradeoff between accuracy and privacy: higher levels of protection translate into further deviation from the original data and consequently less accurate results. Cryptographic techniques do not sacrifice accuracy to achieve privacy. In the general setting [15], the algorithms applied to private data are converted to complex binary circuits with private inputs. Then, using SMC protocols, accurate results are obtained. SMC protocols guarantee that only the final result and any information that can be inferred from the final result and the input is revealed [13]. Such protocols have certain security parameters (e.g., encryption key sizes) that allow users to trade off between cost and privacy [13].

Compared to SMC, sanitization methods have two limitations: the level of privacy protection may not be sufficient and the matching result may exhibit false positives/negatives. On the other hand, SMC methods provide strong privacy and perfect accuracy, but incur prohibitive computational and communication cost. All existing SMC techniques that we are aware of [7], [16], [17], [18], [19] require $O(n * m)$ cryptographic operations where n (respectively m) is the number of records in the first (respectively second) data set. If $n = m = 10{,}000$, such an

- *A. Inan is with the Department of Computer Engineering, Isik University, Sile Kampusu, AMF-303, Universite Sok. No. 2 Mesrutiyet Koyu, 34980 Sile, Istanbul, Turkey. E-mail: ali.inan@isikun.edu.tr.*
- *M. Kantarcioglu is with the Department of Computer Science, University of Texas at Dallas, EC31, 2601 N. Floyd Rd Richardson, TX 75083. E-mail: muratk@utdallas.edu.*
- *G. Ghinita is with the Department of Computer Science, University of Massachusetts at Boston, 100 William T. Morrissey Blvd, Boston, MA 02125. E-mail: gabriel.ghinita@umb.edu.*
- *E. Bertino is with the Department of Computer Sciences, Purdue University, LWSN Bldg, West Lafayette, IN 47907. E-mail: bertino@cs.purdue.edu.*

Manuscript received 22 Sept. 2011; revised 3 Apr. 2012; accepted 2 May 2012; published online 10 May 2012.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-2011-09-0223. Digital Object Identifier no. 10.1109/TDSC.2012.46.

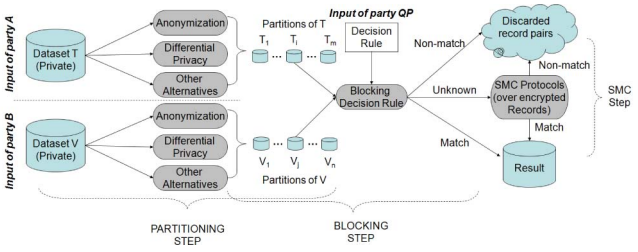

Fig. 1. Overview of the hybrid model.

integration task will require 10^8 cryptographic operations. The cost of each individual operation is very high, and grows with the number of compared attributes. Consequently, none of these techniques are able to provide a solution addressing all relevant application requirements with respect to privacy, cost, and accuracy.

We propose a novel method to address private record matching by combining cryptographic and sanitization techniques. Unlike existing solutions, the tradeoff in our solution is along three dimensions: privacy, cost, and accuracy. Our work is the first systematic approach in this direction.

We assume three participants in our method. These are two data holders, with the data sets to be matched, and the querying party, who provides the matching condition, also called the "decision rule". The proposed private matching technique, outlined in Fig. 1, consists of three phases:

1. *Partitioning.* Each data holder independently partitions its records according to some privacy-preserving mechanism (e.g., k-anonymization, ϵ-differential privacy). The outcome is a set of smaller partitions whose extents are hyper-rectangles in the multidimensional attribute space.
2. *Blocking.* All pairs of partitions from the data holders are input to a blocking decision rule. By looking at the regions covered by the partitions, the blocking decision rule outputs either *match*, *nonmatch*, or *unknown*. Only records within pairs of partitions labeled unknown are input to the costly SMC step. Since this step reduces the costs significantly, it is called the *blocking* step [20].
3. *SMC.* Pairs of records that are still not labeled are matched using cryptographic protocols. Matching record pairs are added to the result.

If the input data sets are too large, we may have to label significant amounts of record pairs using cryptographic techniques. In such cases, since cost of the private record matching process is not known in advance, data holder parties might be unwilling to participate. That is why we consider limiting the costs of cryptographic techniques. This also allows us to analyze the cost-accuracy and cost-privacy relationships. When the upper bound imposed on SMC costs is too low, some record pairs might remain unlabeled at the end of SMC. In order not to reveal irrelevant pairs, we label them as nonmatches. This precaution degrades recall (as defined in Section 4.3.2) since some of those unlabeled record pairs might actually be matching. Fortunately, based on the sanitized views output at the end of the *partitioning* step, pairs that are more likely to match can be given priority during the SMC step.

Our hybrid approach has several advantages over existing methods, which can be summarized as follows:

- Costs are usually much lower than, and at worst equal to the costs of existing cryptographic techniques.
- Record pairs marked as "nonmatch" by the decision rule are protected 100 percent against disclosure.
- Recall varies with the upper bound on SMC costs imposed by participants.
- Our method applies to any privacy preserving mechanism for partitioning a data set and any cryptographic technique for matching private records. During the partitioning step, we investigate several anonymization methods as well as differential privacy.

2 Related Work

Record matching has been studied for more than four decades [1]. However, few methods for *private* record matching have been investigated. Most studies in the field [21], [22], [23], [24] focus on private matching of string attributes (e.g., names and addresses). Our focus is rather on numerical and categorical attributes. Closely related to our work, Al-Lawati et al. [6], propose a secure blocking scheme to reduce costs. The approach has the disadvantage to work only for a specific comparison function. Also, as the focus is mainly on efficiency, the effectiveness of the approach has not been assessed. In contrast, our approach can be used with several comparison functions.

Several approaches investigated the secure set intersection problem [18]. Such methods deal with exact matching and are too expensive to be applied to large databases due to their heavy reliance on cryptography. Agrawal et al. [7] formalize a notion of private information sharing across databases that relies on commutative encryption techniques, leading to several protocols [19], [25], [26].

The hybrid approach discussed in this paper was first proposed in [27]. However, the privacy definition considered before was limited to k-anonymity. We extend the approach to novel privacy preserving partitioning methods. We show that any anonymization method based on generalization and suppression fits this description. The work in [28] extends the hybrid approach to differentially private databases. Anonymization within [27] is replaced by a partitioning step that generates sanitized views of the input data sets through aggregate querying. Even though [28] provides a privacy definition similar to the security definitions of SMC, adherence to this definition is not inspected in detail. In this paper, we show that the protocol as described in [28] has some limitations. Specifically, we describe an attack against differential privacy of the participants, explore a concrete solution to the problem, and study its advantages empirically through extensive experiments.

2.1 Anonymization

Prior to public release of a data set, to protect individual privacy, unique identifiers such as social security numbers are removed. Sweeney shows in [9] that this measure is not

sufficient because *quasi-identifier* attributes can be combined with public directories to accurately identify individuals. Anonymization is one popular solution against such attacks. By generalizing the values of quasi-identifying attributes and/or removing entire records from the data set, anonymization methods try to satisfy certain definitions of anonymity. The most well known of such definitions is k-anonymity [9], which requires every combination of quasi-identifier values (called an *equivalence class*) to appear at least k times in the anonymized data set, so that an individual is indistinguishable within a group of size at least k. This model has been extended by many related works in the area such as ℓ-diversity [10] and t-closeness [29].

2.2 Differential Privacy

Dwork proves in [12] that every privacy protection mechanism is vulnerable to some kind of background knowledge. Instead of tailoring privacy definitions against different types of background knowledge, one should minimize the risk of disclosure that arises from participation into a database. This notion is captured by the differential privacy protection mechanism [12], which addresses the case of statistical databases where users are only allowed to ask aggregate queries. Differential privacy requires random noise to be added to each query result. The magnitude of the noise depends on the privacy parameter ϵ, and sensitivity of the query set $\mathcal{Q}$. Denoting the response to query Q over data set T with Q^T, sensitivity is defined as follows:

Definition 1 (L_1-sensitivity [30]). *Over any two views T_1, T_2 such that $|T_1| = |T_2|$ and T_1, T_2 differ in only one record, the* L_1-sensitivity *of query set $\mathcal{Q} = \{Q_1, \ldots, Q_q\}$ is measured as $S_{L_1}(\mathcal{Q}) = \max_{\forall T_1, T_2} \sum_{i=1}^{q} |Q_i^{T_1} - Q_i^{T_2}|$.*

Theorem 1 gives a sufficient condition for a statistical database to satisfy ϵ-differential privacy [30]:

Theorem 1. *Let $\mathcal{Q}$ be a set of queries answered by a statistical database, and denote by $S_{L_1}(\mathcal{Q})$ the L_1-sensitivity of $\mathcal{Q}$. Then, differential privacy with parameter ϵ can be achieved by adding to each query result random noise X, i.e., $Q_i^T \leftarrow Q_i^T + X$, where X is a random, i.i.d. variable drawn from a Laplace distribution with mean 0 and magnitude $\lambda \geq S_{L_1}(\mathcal{Q})/\epsilon$.*

$S_{L_1}(\mathcal{Q})$ is independent of data set T, and can be determined based on query set $\mathcal{Q}$ alone [30]. However, computing sensitivity is NP-hard [31].

3 Private Record Matching

Record matching is the process of identifying record pairs, across two input data sets, that correspond to similar (or the same) real-world entities. In essence, the problem consists of building a classifier that accurately classifies pairs of records as "match" or "nonmatch." In the private record matching problem, an accurate classifier is assumed to be available [5]. Therefore, private record matching methods focus on classifying all record pairs within the input data sets privately, accurately, and efficiently.

We consider a matching scenario with three participants. These are data holders parties A and B with the data sets T and V, respectively, and a querying party QP that provides the classifier for identification of matching record pairs. In a real-world application, A and B could be hospitals and QP a researcher trying to match patients with similar characteristics such as geographical location, age and sex.

Without loss of generality, let T and V be represented as relations. Let us also assume that these relations have the same schema, $T(A_1, \ldots, A_d)$ and $V(A_1, \ldots, A_d)$. If not, schemas of T and V can be matched using private schema matching techniques such as [5].

Given matching thresholds $\theta_i \geq 0$ and distance functions $d_i : Dom(T.A_i) \times Dom(V.A_i) \rightarrow R^+$, defined over domains of corresponding attributes of T and V, record matching can be expressed as a join of T and V. For $t \in T$ and $v \in V$, the join condition is a decision rule DR that returns $true$ if $d_i(t.A_i, v.A_i) \leq \theta_i$ for all attributes $(1 \leq i \leq d)$ and $false$ otherwise. Formally,

$$DR(t, v) = \left\{ \begin{array}{ll} \text{true} & \text{if } \forall 1 \leq i \leq d, d_i(t.A_i, v.A_i) \leq \theta_i \\ \text{false} & \text{otherwise} \end{array} \right\}.$$

Our task is to identify $T \bowtie_{DR(t,v)} V$ in a privacy preserving manner such that the result will be available to the querying party QP and private records of the data holders that do not satisfy the join condition are not disclosed.

4 The Hybrid Approach

The hybrid approach combines sanitization methods with cryptographic methods in three steps. The first step, *partitioning*, produces sanitized views of the input data sets through perturbation. In Section 4.1, we describe the general characteristics of a partitioning algorithm that can be used in the hybrid approach. Various alternatives that fit this description are discussed in Section 5. The second step of the hybrid approach is the *blocking* step discussed in Section 4.2, where pairs of partitions produced in the *partitioning* step are compared against one another based on the regions covered by each partition. The third step, namely the *SMC* step, labels any pairs of records that were not classified as match or nonmatch in the *blocking* step. Details of the *SMC* step are discussed in Section 4.3. Finally in Section 4.4, we provide our privacy definition.

4.1 Partitioning Step

A partition p consists of a set of points, $Points(p)$ and a d-dimensional hyper-rectangle $Region(p)$ such that $\forall t \in Points(p) \Rightarrow t \in Region(p)$. In other words, every point in $Points(p)$ should be contained by the region of partition p. We denote the interval covered by a region r on dimension A_i as $[x_i, y_i]$, where x_i is the lower bound on attribute A_i and y_i is the upper bound.

Given data set D, a partitioning algorithm outputs a set of partitions $P^D = \{p_1, \ldots, p_k\}$. We focus primarily on space partitioning algorithms that cover the entire data space.[1]

4.2 Blocking Step

Given two regions R_1 and R_2, let $d_i^{inf}(R_1, R_2)$ denote the infimum distance between any pair of records within R_1 and R_2 over the ith dimension. Formally,

1. Inan et al. discuss data partitioning indexes such as R*-trees in [28]. However, their results indicate poor performance due to high sensitivity.

$$d_i^{inf}(R_1, R_2) = inf_{\substack{t \in R_1 \\ v \in R_2}}(d_i(t, v)).$$

By definition, $d_i^{inf}(R_1, R_2)$ is the greatest lower bound on the distance. If $d_i^{inf}(R_1, R_2) > \theta_i$ for some $1 \leq i \leq d$, then no two points from R_1 and R_2 can match.

We define the supremum distance similarly:

$$d_i^{sup}(R_1, R_2) = sup_{\substack{t \in R_1 \\ v \in R_2}}(d_i(t, v)).$$

By definition, $d_i^{sup}(R_1, R_2)$ limits from above the maximum distance between two arbitrary points of R_1 and R_2. If these distance values never exceed the threshold for any attribute, then all points within $R_1 \times R_2$ should match.

Based on infimum and supremum distance functions, the blocking decision rule $BDR(R_1, R_2)$ can be defined as

$$BDR(R_1, R_2) = \left\{ \begin{array}{ll} \text{N if } \exists\ 1 \leq i \leq i \leq d,\ d_i^{inf}(R_1, R_2) > \theta_i \\ \text{M if } \forall\ 1 \leq i \leq i \leq d,\ d_i^{sup}(R_1, R_2) \leq \theta_i \\ \text{U} \qquad\qquad \text{otherwise} \end{array} \right\}.$$

Here, the return values M, N, and U refer to *match*, *nonmatch*, and *unknown*, respectively. Not all pairs of regions can be classified as M and N. Whenever an accurate decision cannot be drawn, the pair is labeled U. Records in such regions will be labeled privately by SMC protocols.

4.2.1 Overall Protocol for Blocking

Let $\{T_i\}_{1 \leq i \leq m}$ (respectively $\{V_j\}_{1 \leq j \leq n}$) be the set of partitions extracted from data set T (respectively V). Algorithm 1 describes the overall protocol for the blocking step. For every partition $\{T_i\}_{1 \leq i \leq m}$ of T and $\{V_j\}_{1 \leq j \leq n}$ of V, the blocking decision rule BDR is evaluated. In step 3, record pairs that will be labeled with SMC protocols are identified. Step 6 inserts matching record pairs to the result set.

Algorithm 1. Protocol for the blocking step
Require: $T = \{T_i\}_{1 \leq i \leq m} \cup T$ and $V = \{V_j\}_{1 \leq j \leq n} \cup V$
1: **for all** Partitions $T_i \in T$ **do**
2: **for all** Partitions $V_j \in V$ **do**
3: **if** $BDR(Region(T_i), Region(V_j)) = U$ **then**
4: Privately match $Points(T_i) \times Points(V_j)$
5: **else if** $BDR(Region(T_i), Region(V_j)) = M$ **then**
6: Add $Points(T_i) \times Points(V_j)$ to the result
7: **end if**
8: **end for**
9: **end for**

Assuming that step 6 only marks the pair (T_i, V_j) as M and that step 4 is performed in the *SMC* step, Algorithm 1 terminates in $\Theta(m \times n)$ time.

4.3 SMC Step

Considering each partition as a small data set by itself, any existing solution for privacy preserving record matching can be applied to match the set of nonblocked partition pairs. In classical SMC protocols, using some cryptographic assumptions, it can be proven that only the final results and anything that could be inferred by looking at the final results are revealed. Our security guarantees are slightly different from the security guarantees provided by the generic SMC protocols [13]. We implicitly assume that disclosure of the output of our privacy preserving partitioning algorithms does not violate privacy. This is reflected in the privacy definition of Section 4.4, where the goal is to reveal *only* the final record matching result, the privacy preserving partitioning of the data sets and anything that can be inferred from the result and the partitioned data sets. Since the *blocking* step only depends on pairs of partitions, it satisfies the goal stated above. In other words, anything revealed during the *blocking* step could be inferred from the partitioned data sets.

4.3.1 Basic SMC Protocol for Record Matching

For each pair of records that is not blocked, we need to securely learn whether such a pair actually matches or not. In other words, for each possibly matching record pair and for each attribute, we need to securely calculate whether $d_i(t.A_i, v.A_i) \leq \theta_i$ is satisfied. Such a secure calculation is possible using generic SMC circuit evaluation techniques [13]. Also recently many protocols have been proposed using special encryption functions such as commutative encryption [7] and homomorphic encryption [18]. Either these protocols, or any other SMC technique that can securely compute $d(t, v)$ could be used in the *SMC* step.

4.3.2 Limited SMC Budget

Efficiency of a blocking scheme is measured by the *reduction ratio* (RR) metric [20]. Given a baseline comparison space $\mathcal{S}$, reduction ratio is the fraction of savings from the comparison space attained by the blocking scheme. We compare our results to a benchmark solution that privately evaluates DR over all record pairs in the Cartesian product $T \times V$. Therefore, $|\mathcal{S}| = |T \times V| = |T| \times |V|$. Then,

$$RR = 1 - \frac{\text{number of secure decision rule evaluations}}{|T| \times |V|}.$$

When the input data sets T and V are large, even after considerable reduction in comparison space, the cost of applying our solutions might be higher than the amount anticipated by the participants. In order to prevent concerns related to high costs from hampering the record matching process, we now discuss an extension to our methods where participants can determine an upper bound on the number of SMC protocol invocations, called the SMC_budget.

Similar to RR, we represent SMC_budget as a fraction of the Cartesian product size:

$$SMC\_budget = \frac{\text{max. number of secure } DR \text{ evaluations}}{|T| \times |V|}.$$

For example, if $|T| = |V| = 10^3$, then $SMC\_budget = 0.01$ implies DR will be evaluated at most $10^6 \times 0.01 = 10^4$ times using the SMC protocols of Section 4.3.1.

The number of record pairs that were not labeled after the *blocking* step, hence must be labeled in the SMC step, is $(1 - RR) \times |T| \times |V|$. If $1 - RR \leq SMC\_budget$, then there is no challenge in enforcing the limits over SMC operations because the budget meets or exceeds the need. However, when $1 - RR > SMC\_budget$, then some record pairs cannot be properly labeled.

In order to prevent disclosure of irrelevant record pairs, we assume that all such records are excluded from the result set (i.e., assumed to be nonmatching record pairs). Whenever SMC_budget is insufficient, record pairs should

be chosen carefully to maximize the number of matching record pairs found in the *SMC* step. This notion is captured by the *recall* measure. Let H be some heuristic that guides us in selecting the record pairs toward which the SMC_budget is spent. Then, the recall of H, denoted $Recall_H$, is the fraction of matching record pairs that H can identify in the *SMC* step. Formally, denoting matching record pairs by the set n_M, the recall of heuristic H is

$$Recall_H = \frac{\text{number of matching pairs found by } H}{|n_M|}.$$

A naive approach to enforce SMC_budget would be choosing a random subset of unlabeled record pairs. Yet, it makes more sense to use the information contained in partition regions. Below we discuss various heuristics that help identify possibly matching record pairs. Among these, the heuristic that has the maximum recall should be favored.

4.3.3 Selection Heuristics

Our heuristics rank pairs of partitions. In the SMC step, pairs are processed according to these ranks. If SMC_budget is low, then low-ranked pairs may be excluded and automatically labeled as "nonmatch."

The heuristics are outlined below. An empirical evaluation of these heuristics is provided in Section 7.5.

H_1—**Minimum comparison cost first.** In this heuristic, partitions of data set T are sorted with respect to the number of secure DR evaluations required to find all matching records of V. Then, the partitions are processed in ascending order. The idea is maximizing the fraction of records of T that are matched against V. H_1 would be advantageous if the partitions were weighted based on some criteria. For example, partitions that contain individuals of a certain age group may be given priority over others.

H_2—**Minimum volume partition first.** In this heuristic, partitions p of T are sorted with respect to the volume of their regions, $Region(p)$. Then, partitions are processed in ascending order. Considering records as random variables supported over their partition regions, this heuristic assumes that lower volumes imply less uncertainty in estimating the actual value of a record. Based on this idea, partitions with the smallest region are processed first.

H_3—**Partition pair (p_1, p_2) with maximum $Region(p_1) \cap Region(p_2)$ volume first.** This heuristic assumes the volume of the intersection between partition regions is an accurate indicator of possibly matching records. Therefore, pairs are ordered based on normalized intersection volumes and processed in descending order.

4.4 Privacy Definition

Privacy guarantees of SMC techniques can be proven under reasonable assumptions. We believe that a similar theoretical framework is needed for our hybrid approach. To this end, we extend the basic definitions and techniques used in SMC so that they apply to our hybrid framework. We will focus on security/privacy definitions of the semihonest model, where each party reveals some sanitized information about its data.[2]

2. The semihonest model is the most commonly used model in SMC-based privacy preserving data analysis algorithms [32].

In the semihonest model [13] a computation is secure if a party's view during protocol execution can be effectively simulated based on its input and output. This does not imply that all private information is protected. Under this definition, disclosure of any information that can be deduced from the final result is not a violation.

We extend the basic semihonest model [13] by including sanitized data in the form of anonymized data sets or statistical query results that satisfy differential privacy definitions. We assume that such sanitized data are public and accessible by all participants. Formally, let $\bar{a} = (a_1, \ldots, a_z)$ be the sanitized data of all the parties. Let $f : (\{0,1\}^*)^z \longmapsto (\{0,1\}^*)^z$ be a probabilistic, polynomial-time functionality, where $f_i(x_1, x_2, \ldots, x_z)$ denotes the ith component of $f(x_1, x_2, \ldots, x_z)$ and let Π be a z-party protocol for computing f. For $I = \{i_1, i_2, \ldots, i_t\} \subseteq [z]$ where $[z]$ denotes the set $\{1, 2, \ldots, z\}$, we let $f_I(x_1, x_2, \ldots, x_z)$ denote the subsequence $f_{i_1}(x_1, \ldots, x_z)$, $\ldots$, $f_{i_t}(x_1, \ldots, x_z)$. Let the view of the ith party during an execution of protocol Π on $\bar{x} = (x_1, x_2, \ldots, x_z)$, denoted by $view_i^{\Pi}(\bar{x})$, be $(x_i, r_i, m_i^1, \ldots, m_i^t)$ where r_i represents the outcome of the ith party's internal coin tosses, and m_i^j represents the jth message received by the ith party. Also, given $I = \{i_1, i_2, \ldots, i_t\}$, we let $view_I^{\Pi}(\bar{x})$ denote the subsequence $(I, view_{i_1}^{\Pi}(\bar{x}) \ldots view_{i_t}^{\Pi}(\bar{x}))$.

We say that protocol Π privately computes f if there exists a probabilistic polynomial time algorithm, denoted S, such that for every $I \subseteq [z]$, it holds that

$$\begin{aligned} &\{(S(I, (x_{i_1}, \ldots, x_{i_t}), \bar{a}, f_I(\bar{x})), f(\bar{x}))\}_{\bar{x} \in (\{0,1\}^*)^z} \\ &\quad \stackrel{C}{\equiv} \left\{ \left(view_I^{\Pi}(\bar{x}, \bar{a}), output^{\Pi}(\bar{x}) \right) \right\}_{\bar{x} \in (\{0,1\}^*)^z}, \end{aligned} \tag{1}$$

where $\stackrel{C}{\equiv}$ denotes computational indistinguishability [13].

In our context, three parties want to compute the record matching function $f(T, V, DR)$ where data set T (respectively V) is the input of the first party (respectively the second party) and DR is the input of the QP. Also, we define $f_1(T, V, DR) = f_2(T, V, DR) = \emptyset$ and $f_3(T, V, DR) = T \bowtie_{DR(t,v)} V$ (i.e., the set of matched records). In addition, let $\bar{a}$ be the union of the sanitized data released during the blocking step. We say that protocol Π privately computes record matching function $f(T, V, DR)$ if (1) above holds.

Compared to the existing privacy definitions in the semihonest model, we assume that all sanitized data (e.g., anonymized data or differentially private statistical query results) are available to any coalition of parties. The objective of the privacy preserving protocol is to reveal nothing more than what can be inferred by *all* sanitized information, original inputs of the colluding parties and the final function result (here, the set of matching record pairs).

In contrast to classic SMC models, our hybrid model can trade off privacy versus efficiency easily. If no sanitized data is revealed (i.e., $\bar{a} = \emptyset$), our model will be equivalent to SMC models. On the other hand, as we show in this study, by revealing sanitized data, it is possible to improve the efficiency of SMC protocols without sacrificing accuracy.

5 Privacy Preserving Partitioning

The general structure of a partition algorithm was outlined in Section 4.1. Now we discuss several alternatives of such

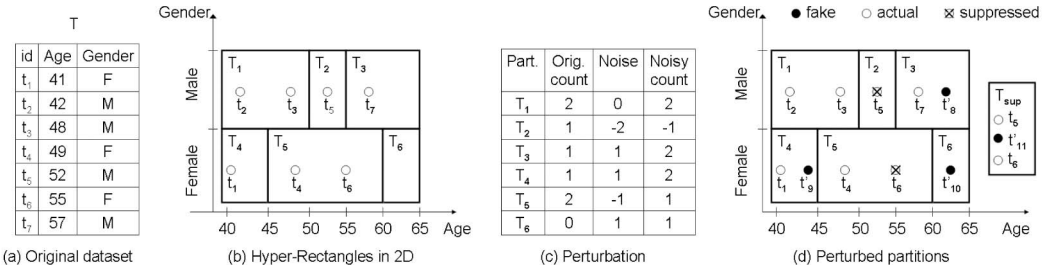

Fig. 2. Overview of differentially private partitioning.

methods that protect individual privacy. Section 5.1 presents anonymization methods as partitioning algorithms and provides a mapping from anonymized data sets to a set of partitions. Section 5.2 discusses construction of multidimensional index structures tailored to differential privacy. In Section 5.3, we discuss how encrypted records are generated according to the partitioning algorithms.

5.1 Using Anonymization for Partitioning

Most anonymization methods can be categorized as generalization algorithms.[3] Suppose all categorical quasi-identifier attribute domains are converted to discrete numeric domains. A deterministic process that maps equivalence classes to partitions is provided in Algorithm 2.

Algorithm 2. Mapping generalization-based anonymized data sets to a set of partitions
Require: Set E of equivalence classes
1: $P \leftarrow \emptyset$
2: **for all** Equivalence classes $e \in E$ **do**
3: **for all** Quasi-identifier attributes $A_i \in A^{qid}$ **do**
4: $min_i \leftarrow$ min. value over A_i in records of e
5: $max_i \leftarrow$ max. value over A_i in records of e
6: **end for**
7: Generate a partition p that represents e
8: $p.Region = [min_1, max_1] \times \ldots \times [min_{|A^{qid}|}, max_{|A^{qid}|}]$
9: $p.Points =$ all records of e
10: Insert p into P
11: **end for**
12: Return the set of partitions, P

The algorithm maps every equivalence class to a partition in the $|A^{qid}|$-dimensional space, where A^{qid} is the set of quasi-identifier attributes.[4] Each such partition covers the region that represents the minimum bounding rectangle of the records within the corresponding equivalence class. The points of the partition are simply the records of the equivalence class. Notice that Algorithm 2 operates on the anonymized version of the data set and need not access individual records. Therefore, the set of partitions output by the algorithm offers the same privacy guarantees as the underlying definition of anonymity (e.g., k-anonymity).

If the anonymized data set input to Algorithm 2 is a flat relation, then an initial iteration over all (anonymized) records should identify set E of equivalence classes and their respective extents. Then, partitions can be constructed and points assigned to each partition. Overall complexity of the algorithm is $O(n)$, where n is the number of records.

5.2 Using Differential Privacy for Partitioning

In this step, each party generates a set of d-dimensional hyper-rectangles and partitions its data set accordingly. A natural way of obtaining these hyper-rectangles is to perform an indexing of the record set. We explore partitioning based on BSP-trees and kd-trees. The structure of the index may also reveal sensitive information. For example, kd-trees reveal the median values along the split dimension. To protect against such disclosures, the index itself must be perturbed by the privacy mechanism.

Consider an indexing algorithm that accesses the data using range queries only. If the transcript of all queries issued is protected with differential privacy, then the outcome (i.e., the perturbed index) does not leak any private information.[5] This is exemplified in Section 5.2.2, where we present the *Adaptive-kd* algorithm that adaptively constructs a kd-tree based on noisy information about the data set.

Consider table T in Fig. 2a. Suppose the set of hyper-rectangles generated by party A holding data set T is as shown in Fig. 2b. There are six such hyper-rectangles, denoted $T_1, \ldots, T_6$. The records associated with each hyper-rectangle are shown as 2D points. The example considers disjoint ranges (i.e., nonoverlapping hyper-rectangles), so that L_1-sensitivity of the query set is easy to compute.

To facilitate blocking, data holders disclose: 1) the extents of the hyper-rectangles obtained from the *partitioning* step, and 2) the cardinality of each partition. Assuming that the partitioning step respects privacy, the extents can be directly disclosed. However, random noise must be injected into the record counts of every partition. The amount of noise depends on the privacy parameter ϵ as well as the sensitivity of the partitioning step. Specifically, each party determines sensitivity $S^T_{L_1}$ and generates noise drawn from a Laplace distribution with mean 0 and magnitude $S^T_{L_1}/\epsilon$. Continuing our example, since partitions $T_1, \ldots, T_6$ have disjoint extents, the resulting sensitivity has value two. Therefore, Laplace noise with magnitude $2/\epsilon$ is injected into the cardinality of each partition.

Random noise values as well as original and noisy counts for the partitions are provided in Fig. 2c. Positive noise is incorporated by adding fake records to the partition. In Fig. 2d, the noises for partitions T_3, T_4, and T_6 are $+1$, which requires insertion of fake records t'_8, t'_9, and t'_{10} to each partition, respectively. Every fake record is assigned a remote value (outside attribute domains[6]) so that the decision rule never evaluates to *true* when fake and

3. A notable exception is the *Anatomy* method [33], which uses permutation of records.
4. We expect every attribute of DR to be a quasi-identifier attribute.
5. In effect, this process is equivalent to perturbing the response to a complex query that requests a particular index of the data set.
6. Such values can be generated based on domain knowledge.

original records are matched in the *SMC* phase. Therefore, no information is disclosed about which partitions contain fake records and how many. We emphasize that fake records are never disclosed other than in *encrypted* form. If the underlying encryption function is semantically secure [13], this is equivalent to releasing only (noisy) counts.

Negative noise is more difficult to handle. Consider partition T_5 that contains records t_4 and t_6. Added noise is -1. For the perturbed partition to reflect the noisy response to the count over the region of T_5, the partitioning algorithm has to remove one of the records; both records cannot be accommodated. Here, t_6 is chosen uniformly at random and moved to a separate partition T_{sup} of suppressed records. Record t_4 still remains within the partition, such that the cardinality of T_5 after perturbation equals the noisy count.

The support of Laplace distribution is $(-\infty, \infty)$, therefore especially with undersized partitions, the negative noise may exceed the original, nonperturbed count. One example in Fig. 2c is partition T_2. Since added noise is -2, the partitioning algorithm needs to suppress exactly two records from T_2. Yet, there is only one eligible record: t_5. In such cases, we enforce the partitioning algorithm to add to T_{sup} as many fake records as the deficit. Here, the deficit is only 1 and t'_{11} is inserted to T_{sup}.

While this requirement increases the overall cost of matching, it also increases the security/privacy guarantees of the protocol. With this requirement in place, the number of suppressed records $|T_{sup}|$ equals the absolute value of the sum of all negative noises (here, $|(-2) + (-1)| = 3$). For a formal analysis please refer to Section 6.1.

Fig. 2d provides a visualization of the perturbed partitions. Fake, suppressed, and actual records are clearly marked. In addition to the six partitions we started with, an additional partition that contains suppressed records, T_{sup} is released. To facilitate correctness of the matching outcome, the records in T_{sup} will be compared against the *entire* data set of the other party (i.e., the *SMC* step is performed for $T_{sup} \times V$ and $T \times V_{sup}$, respectively).

5.2.1 Binary Space Partitioning Trees

In d-dimensional space, BSP-trees are constructed by recursively dividing the space into subspaces using $(d-1)$-dimensional axis-orthogonal hyperplanes. The axis is chosen based on the depth of the tree node which is being split: if the depth of node n is denoted by $Depth(n)$, the splitting hyperplane is orthogonal to axis $Depth(n)\%d + 1$ and splits the space into two equal halves.[7]

Fig. 3 exemplifies a BSP-tree of height three in the two-dimensional space. The data space is first partitioned into equal partitions on axis A_1 through the line $A_1 = 0.5$. For all intermediate nodes at level 1, the next splitting line is orthogonal to A_2, $A_2 = 0.5$. At level 2, the splitting line partitions on A_1 axis again. The resulting partitions are the leaf nodes of the tree: $p_1, \ldots, p_8$. Partitions cover the same volume and their regions do not depend on data distribution.

Let $\mathcal{Q}^h$ be the query set that asks for the BSP-tree index of height h on some data set D. $\mathcal{Q}^h$ contains exactly 2^h count queries: one for each leaf node's extent. The extents depend only on attribute domains and are disjoint regardless of h and D. Therefore, the sensitivity of $\mathcal{Q}^h$ is 2.

7. The "+1" term only maps dimension indices from $[0, d-1]$ to $[1, d]$.

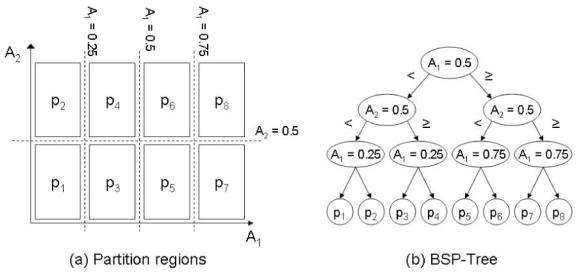

Fig. 3. BSP-tree example.

5.2.2 Adaptive kd-Tree Construction

The root node of a kd-tree covers the entire space. At each step, a splitting dimension and a split value from the range of the current node on that dimension are chosen heuristically to divide the space into subspaces. Similar to BSP-trees, we choose the splitting dimension dim based on node depth: $dim = Depth(n)\%d + 1$. The split value on the dimension dim is chosen based on the distribution of the points. Selecting the median value on dim-axis of the points distributes the points evenly among the left and right child nodes. In our analysis, we use the noisy median for categorical attributes and the noisy mean for numerical attributes,[8] both calculated from noisy counts and sums.

Algorithm 3 outlines this approach. Starting with the root node, until tree height reaches the maximum value h, the next available node is chosen for splitting. Theorems 2 and 3 analyze the sensitivity of Algorithm 3.

Algorithm 3. Adaptive kd-tree construction
Require: h, maximum height of the tree
1: $Region(root) \leftarrow A_1 \times \cdots \times A_d$
2: Stack $s \leftarrow \emptyset$
3: s.Push($root$)
4: **while** Stack is not empty and tree height $\leq h$ **do**
5: Node $n \leftarrow s$.Pop()
6: Dimension $dim \leftarrow$ Depth$(n)\%d + 1$
7: **if** dim is numerical **then**
8: $Sum \leftarrow$ query Sum(dim) over $Region(n)$
9: $Count \leftarrow$ query Count$(*)$ over $Region(n)$
10: $val \leftarrow Sum/Count$
11: **else**
12: **for all** Categories c in $Region(n)_{dim}$ **do**
13: $Counts[c] \leftarrow$ query Count$(*)$ over
14: $Region(n) \cap A_1 \times \cdots \times A_{dim-1} \times \{c\} \times \cdots \times A_d$
15: **end for**
16: $val \leftarrow$ median calculated from $Counts$
17: **end if**
18: $Left(n) \leftarrow$ left-split of $Region(n)$ on dim at val
19: $Right(n) \leftarrow$ right-split of $Region(n)$ on dim at val
20: s.Push($Left(n)$), s.Push($Right(n)$)
21: **end while**

Theorem 2. *Let $\mathcal{Q}^h = \{Q_1^h, \ldots, Q_q^h\}$ be the query set that adaptively constructs a kd-tree of height h. The largest subset $\mathcal{Q}_*^h \subseteq \mathcal{Q}^h$ of queries with overlapping query regions has size*

8. Querying the median is costlier in terms of sensitivity.

$$n_h = 2 \times \sum_{i=0}^{h-1} I(A_{i\%d+1} \text{ is num.}) + \sum_{i=0}^{h-1} I(A_{i\%d+1} \text{ is cat.}).$$

Proof. When $h = 1$, the tree contains two leaf nodes and the root. $\mathcal{Q}^1$ consists of the queries that determine the split value of the root on attribute A_1. If A_1 is categorical, then $\mathcal{Q}^1$ is a set of nonoverlapping count queries and $n_1 = 1$. If A_1 is numerical, $\mathcal{Q}^1$ consists of a sum and a count query over the entire data space and $n_1 = 2$. Using indicator functions $I(.)$, n_1 can be expressed as

$$n_1 = 2 \times I(A_1 \text{ is numerical}) + I(A_1 \text{ is categorical}).$$

The base case has been proven. Next, assume that $|\mathcal{Q}_*^{h-1}| = n_{h-1}$ holds for $h > 1$. To add another level to the tree, current leaf nodes should be split on attribute $A_{(h-1)\%d+1} = A^{\#}$.

The set of queries determining the split values ($\mathcal{Q}^h - \mathcal{Q}^{h-1}$) do not overlap. By definition, any two sibling nodes cover disjoint areas. Therefore, depending on whether $A_{\#}$ is categorical or numerical, either 1 or 2 new queries are added to $\mathcal{Q}_*^{h-1}$. Accordingly, n_h can be calculated as

$$\begin{aligned} n_h &= n_{h-1} + 2 \times I(A_{\#} \text{ is num.}) + I(A_{\#} \text{ is cat.}) \\ &= 2 \sum_{i=0}^{h-1} I(A_{i\%d+1} \text{ is num.}) + \sum_{i=0}^{h-2} I(A_{i\%d+1} \text{ is cat.}). \end{aligned}$$

□

Theorem 3. *L_1-sensitivity of $\mathcal{Q}^h$ is $2n_h$.*

Proof. The proof follows directly from the definition of L_1-sensitivity. Changing one record of the data set changes the result of count queries by at most two. For sum queries, assuming the numerical attribute in question is normalized to $[0, 1]$, the maximum distance in L_1-norm corresponds to changing some record's attribute value from 0 to 1. Here, the L_1-norm distance changes[9] by at most two. In total, adaptively building the kd-tree (or equivalently retrieving the partition regions) has sensitivity $2n_h$. □

Partition cardinalities are obtained with an additional set of count queries. Including these, sensitivity of Algorithm 3 becomes $2n_h + 2$.

5.3 Releasing Encrypted Records

As outlined in Section 5.2, the count queries over partition extents will be answered with not an integer value but a set of encrypted records so that records that are not blocked in the *blocking* step can be input to the *SMC* step. We now discuss how such encrypted records will be generated.

The process is rather straightforward with anonymization methods. Each equivalence class represents a partition and simply encrypting the original records is sufficient.

Differential privacy does not allow releasing the exact number of records within partitions. To accommodate the noise added to the original count, our strategy either adds fake records to the partition or suppresses existing records.

Algorithm 4 provides the details. First, a partition to hold suppressed records is created. The region of this partition covers the entire data space so that P_{sup} is never blocked by the blocking decision rule. Then, the algorithm iterates over all partitions, generating perturbed partitions. Step 6 adds r fake records to partition p so that the number of encrypted records will equal the noisy count at the end. Otherwise, r records are moved to P_{sup} in steps 7 to 15. As many of the suppressed records as possible are chosen from p. Any deficit is met by addition of fake records in step 14. Finally, in step 18, the records are encrypted.

Algorithm 4. Partition perturbation in statistical databases
Require: Set of partitions P, Laplace noise generator $Lap(\lambda)$ suitable for the partitioning algorithm

```
1: Create an empty partition P_sup for suppressed records
2: Set Region(P_sup) to the entire domain
3: for all Partitions p ∈ P do
4:   r ← round(Lap(λ))
5:   if r > 0 then
6:     Add r fake records to partition Points(p)
7:   else
8:     r ← r × (−1)
9:     for i = 1 to min(r, |Points(p)|) do
10:      Select t ∈ Points(p) uniformly at random
11:      Suppress t by moving it to Points(P_sup)
12:    end for
13:    if r > |Points(p)| then
14:      Add r − |Points(p)| fake rec.s to Points(P_sup)
15:    end if
16:  end if
17: end for
18: Encrypt all records in all partitions
```

Step 14 might appear unnecessary. Intuitively, one can only suppress existing records, not fake records. We formally explain the need for this step in Section 6.1.

6 Security Analysis

In order to prove security under the privacy definition of Section 4.4, each party's view should be simulated based on its input and output. We only simulate the partitioning algorithm's output in terms of encrypted records and partition extents (i.e., *Points* and *Region* of the partitions). The *blocking* step is performed entirely on partition extents and can be simulated easily. The *SMC* step should by itself be secure (not only according to the definition of Section 4.4 but in the strict SMC sense). Security of the entire protocol follows from secure composition of the three steps [13].

Simulations over anonymization based partitioning are rather straightforward. Therefore below, we discuss security of statistical databases protected with differential privacy.

6.1 Revealing Suppressed Records

Algorithm 4 of Section 5.3 suppresses records only when the noise for a particular partition is negative. For data set T, the total number of suppressed records is

9. Normalization can be performed as a preprocessing step. The ranges of continuous attributes may be determined during schema matching).

$$|T_{sup}| = \sum_{i=1}^{m} |\alpha_i| \times I(\alpha_i < 0),$$

where $\alpha_i \sim Lap(\lambda)$ is the noise added to the count for each partition T_i and $I(.)$ is an indicator function that returns 1 when α_i is negative and 0 otherwise.

Notice that $|T_{sup}|$ (similarly, $|V_{sup}|$ of data set V) is never queried. Therefore, $|T_{sup}|$ is not part of the sanitized data (in this case, differentially private query results, denoted $\bar{a}$ in the privacy definition) and revealing the number of suppressed records to the querying party in the *partitioning* step violates differential privacy. Theorem 4 provides a proof by contradiction.

Theorem 4 (Disclosure of $|T_{sup}|$). *Given query set $\mathcal{Q}$ and database T, let $\mathcal{K}$ be a mechanism that returns a transcript $\mathcal{TR}_{\mathcal{Q}}^{T} = \{(Q_1, a_1), \ldots, (Q_{|\mathcal{Q}|}, a_{|\mathcal{Q}|}), |T_{sup}|\}$ s.t.*

1. *Each a_i is the noisy response to query Q_i according to differential privacy. Formally, $a_i = Q_i^T + \alpha_i$, where Q_i^T is the response over T and α_i is random noise drawn from Laplace distribution $L(0, \lambda)$ with $\lambda \geq S_{L_1}(\mathcal{Q})/\epsilon$,*
2. *$|T_{sup}|$ is the number of records suppressed during partitioning. Specifically,*

$$|T_{sup}| = \sum_{i=1}^{|\mathcal{Q}|} |\alpha_i| \times I(\alpha_i < 0).$$

Mechanism $\mathcal{K}$ violates differential privacy.

Proof. Suppose releasing $|T_{sup}|$ does not violate differential privacy. Consider a BSP-tree of height 1. The domain will be partitioned into two partitions T_1 and T_2 and query set will be $\mathcal{Q} = \{Count(Region(T_1)), Count(Region(T_2))$. In response to $\mathcal{Q}$, the database returns noisy counts, and the number of suppressed records. The transcript is $\mathcal{TR}_{\mathcal{Q}} = \{nc(T_1), nc(T_2), |T_{sup}|\}$.

Let Ψ_i, Φ_i, Θ_i denote the random variables corresponding to $\mathcal{TR}_{\mathcal{Q}}^{D_i}$ over data set D_i. Since $\mathcal{K}$ satisfies differential privacy, $\mathcal{K}$ provides ϵ-indistinguishability [30] for all sibling data sets D_1, D_2 and all possible responses (ψ, ϕ, θ):

$$\left| \ln \frac{Pr[\Psi_1 = \psi, \Phi_1 = \phi, \Theta_1 = \theta]}{Pr[\Psi_2 = \psi, \Phi_2 = \phi, \Theta_2 = \theta]} \right| \leq \epsilon. \tag{2}$$

The first two elements of the transcript are constructed according to differential privacy. This implies

$$\left| \ln \frac{Pr[\Psi_1 = \psi, \Phi_1 = \phi]}{Pr[\Psi_2 = \psi, \Phi_2 = \phi]} \right| \leq \epsilon. \tag{3}$$

Let E_1 denote the event that $\Psi_1 = \psi, \Phi_1 = \phi$. Similarly, let E_2 be $\Psi_2 = \psi, \Phi_2 = \phi$. Equation (2) can be written as

$$\left| \ln \frac{Pr[\Theta_1 = \theta | E_1] \times Pr[E_1]}{Pr[\Theta_2 = \theta | E_2] \times Pr[E_2]} \right| \leq \epsilon, \tag{4}$$

$$\left| \ln \frac{Pr[\Theta_1 = \theta | E_1]}{Pr[\Theta_2 = \theta | E_2]} + \ln \frac{Pr[E_1]}{Pr[E_2]} \right| \leq \epsilon, \tag{5}$$

$$\left| \ln \frac{Pr[\Theta_1 = \theta | \Psi_1 = \psi, \Phi_1 = \phi]}{Pr[\Theta_2 = \theta | \Psi_2 = \psi, \Phi_2 = \phi]} \right| \leq 2\epsilon. \tag{6}$$

Here, (4) conditions both numerator and denominator on E_i. In (5), logarithm of multiplication is converted to a sum. Equation (6) uses (3) to further bound the probability.

Finding any configuration $(D_1, D_2, \psi, \phi, \theta)$ that violates (6) is sufficient to disprove ϵ-indistinguishability of $\mathcal{K}$. We first fix sibling data sets D_1 and D_2 such that $|D_1| = |D_2|$ and, D_1 and D_2 differ in only one record. The case where the update does not change the counts over $Region(T_1)$ and $Region(T_2)$ does not lead to a contradiction. That is why, we assume that the sibling data sets D_1 and D_2 are constructed in such a way that

$$\begin{aligned} \left|T_1^{D_1}\right| &= t_1, & \left|T_2^{D_1}\right| &= t_2, \\ \left|T_1^{D_2}\right| &= t_1 - 1, & \left|T_2^{D_2}\right| &= t_2 + 1. \end{aligned}$$

This implies, the update operation that yields D_2 moves one record of T_1 to T_2.

$\mathcal{K}$ perturbs the original counts with independent Laplace noise $\alpha_1^{D_i}, \alpha_2^{D_i} \sim L(0, \lambda)$ where, $\lambda = S(\mathcal{Q})/\epsilon$. Therefore, the transcript $\mathcal{TR}_{\mathcal{Q}}^{D_i}$ is constructed as

$$\Psi_i = \left|T_1^{D_i}\right| + \alpha_1^{D_i}, \tag{7}$$

$$\Phi_i = \left|T_2^{D_i}\right| + \alpha_2^{D_i}, \tag{8}$$

$$\Theta_i = \left|\alpha_1^{D_i}\right| \times I\left(\alpha_1^{D_i} < 0\right) + \left|\alpha_2^{D_i}\right| \times I\left(\alpha_2^{D_i} < 0\right). \tag{9}$$

Using (6) and (7) we can establish a relationship between $\alpha_1^{D_1}$ and $\alpha_1^{D_2}$ added to Ψ_1 and Ψ_2, respectively.

$$\begin{aligned} \psi = \Psi_1 = \Psi_2 &\Longrightarrow \left|T_1^{D_1}\right| + \alpha_1^{D_1} = \left|T_1^{D_2}\right| + \alpha_1^{D_2} \\ &\Longrightarrow t_1 + \alpha_1^{D_1} = t_1 - 1 + \alpha_1^{D_2} \\ &\Longrightarrow \alpha_1^{D_2} = \alpha_1^{D_1} + 1. \end{aligned} \tag{10}$$

The same analysis over Φ_1, Φ_2 and (8) yields

$$\alpha_2^{D_2} = \alpha_2^{D_1} - 1. \tag{11}$$

We are now ready to fix the values of ψ, ϕ to provide a contradiction. Suppose $\psi = t_1 - 2$ and $\phi = t_2 + 3$. Given $\Psi_1 = \psi$, $\Phi_1 = \phi$, $\Psi_2 = \psi$, $\Phi_2 = \phi$, these imply $\alpha_1^{D_1} = -2$, $\alpha_2^{D_1} = 3$, $\alpha_1^{D_2} = -1$ and $\alpha_2^{D_2} = 2$.

Then, $\Theta_1 = |-2| \cdot I(-2 < 0) + |3| \cdot I(3 < 0) = 2$ and similarly $\Theta_2 = |-1| \cdot I(-1 < 0) + |2| \cdot I(2 < 0)] = 1$.

Inserting these values into (9) is sufficient

$$\left| \ln \frac{Pr[\theta = 2 | \Psi_1 = \psi, \Phi_1 = \phi]}{Pr[\theta = 1 | \Psi_2 = \psi, \Phi_2 = \phi]} \right| = |\ln 1 - \ln 0| = \infty.$$

Since $\infty \not\leq 2\epsilon$ for any finite value of ϵ, we conclude based on this counter example that mechanism $\mathcal{K}$ does not provide ϵ-indistinguishability. □

In order to prevent possible inferences, data holder parties may choose not to release the partition that contains suppressed records. The resulting protocol will strictly adhere to differential privacy. However, as a side effect, any record in T_{sup} (or, respectively, V_{sup}) will not appear in the matching result regardless of the budget SMC_budget set

aside for the *SMC* step. The cost of increased security is reduced recall in matching. In this scenario, the matching function denoted f in the privacy definition becomes a randomized matching function.

On the other hand, $|T_{sup}|$ is completely random in nature. Every term α_i in the sum is a random variable, and $|T_{sup}|$ is independent from the data set T. Consequently, data holders might still be willing to release $|T_{sup}|$ to increase the recall.

Below, we consider both alternatives. If $|T_{sup}|$ is released, then it should be considered part of sanitized data $\bar{a}$ described in the privacy definition. When presenting experimental results over statistical databases in Section 7, the two alternatives will be treated separately.

6.1.1 Data Holder Parties

The roles of data holder parties A and B are symmetric. Without loss of generality, we discuss over party A.

The input of A is comprised of its data set T and a partitioning algorithm (e.g., a BSP-tree index or an adaptive kd-tree index) $Partition^{\epsilon}$ parameterized by the privacy measure, ϵ. We assume that all system parameters (i.e., ϵ, tree-height, etc.) are public information.

The view $view_A^{\Pi}$ of A contains the messages A sends and receives during execution of the protocol Π:

- Statistical queries A receives from QP,
- Noisy responses to these queries, and
- Encrypted records sent to QP at the end of the *partitioning* step.

The output $output_A^{\Pi}$ of A is the empty set. A does not learn any matching record pairs (i.e., the function outcome). Since $output_A^{\Pi}$ is completely independent of $view_A^{\Pi}$, we skip formal simulation.

6.1.2 Querying Party QP

QP provides as input the decision rule, $DR(t, v)$. The output of this party is the record matching outcome represented as a set of matching record pairs: $M = T \bowtie_{DR(t,v)} V$.

The following constitutes $view_{QP}^{\Pi}$:

- Statistical queries issued to A and B during partitioning and the noisy responses received from A and B,
- Partitioning of T in terms of partition regions and encrypted records belonging in each partition (possibly including the partition of suppressed records), and
- Partitioning of V, similar to that of T.

Privacy definition of Section 4.4 places all queries and their responses inside the sanitized data $\bar{a}$. Therefore, this portion of $view_{QP}^{\Pi}$ need not be simulated: the simulator can simply copy the statistical queries and the responses from $\bar{a}$ to its output. The main challenge for the simulation is rather the partitioning of data sets T and V, which has to be performed based on the output (i.e., the set M above), the input (i.e., the decision rule), and the sanitized view $\bar{a}$ (i.e., differentially private query responses).

Without loss of generality, we present a simulator for partitioning T. The extension to V is straightforward. Algorithm 5 gives the details. Depending on whether suppressed records are released or not, $countSupp \in \bar{a}$ will be a null value or the actual number of suppressed records.

Algorithm 5. Simulator S_{QP} simulating $view_{QP}^{\Pi}$ on T
Require: Differentially private query responses by A, set of matching record pairs M, number of suppressed records $countSupp$
Ensure: A computationally indisting. partitioning of T

1: Create a simulated partitioning set ST
2: **for all** Queries $Q_i, 1 \leq i \leq m$ **do**
3: Create new partition ST_i
4: $Region(ST_i) \leftarrow$ Query region of Q_i
5: $Points(ST_i) \leftarrow \emptyset$
6: $ST \leftarrow ST \cup ST_i$
7: **end for**
8: Create a partition ST_{sup} for suppressed records
9: Project M to T: $\pi_T^M = \{t | (t, v) \in M, t \in T\}$
10: **for all** $t \in \pi_T^M$ **do**
11: $ST_i = \{ST_i \in ST | t \in Region(T_i)\}$
12: $Points(ST_i) \leftarrow Points(ST_i) \cup t$
13: **end for**
14: **for** $i = 1$ to m **do**
15: **while** $|Points(ST_i)| < nc(T_i)$ **do**
16: Insert a fake encrypted record to $Points(ST_i)$
17: **end while**
18: **while** $|Points(ST_i)| > nc(T_i)$ **do**
19: Move an encrypted record from ST_i to ST_{sup}
20: **end while**
21: **end for**
22: **if** $countSupp$ is not null **then**
23: **while** $|Points(ST_{sup})| < countSupp$ **do**
24: Insert a fake encrypted record to $Points(ST_{sup})$
25: **end while**
26: Return $ST \cup ST_{sup}$
27: **else**
28: Return ST
29: **end if**

Steps 1 to 8 of the algorithm create a copy of the original partitioning based on the statistical queries answered by A. Initially, all partitions are empty. In steps 9 to 13, records of T in the matching result M are hashed to their corresponding partitions. The decision depends only on the region covered by a partition and the record's attribute values. Then, in steps 14 to 21, partition points are adjusted according to the noisy counts obtained from differentially private query responses. If the Laplace noise added by A to the count is positive or not every record in $Points(T_i)$ appears in M, step 16 adds fake records to bridge the gap. On the other hand, if the noise is negative and T_{sup} is released, excess records identified from M are moved to the partition of suppressed records, ST_{sup}. Finally, depending on whether T_{sup} is part of the partitioning or not, either ST or its union with suppressed records is returned.

Theorem 5 proves security with respect to the querying party according to our privacy definition.

Theorem 5 (Security of Π_{QP}). *The output of simulator S_{QP} is computationally indistinguishable with $view_{QP}^{\Pi}$, i.e., $S_{QP} \overset{C}{\equiv} view_{QP}^{\Pi}$.*

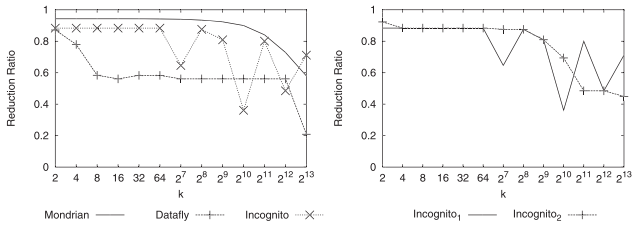

Fig. 4. (a) Reduction ratio of k-anonymity, (b) Comparison of two Incognito heuristics.

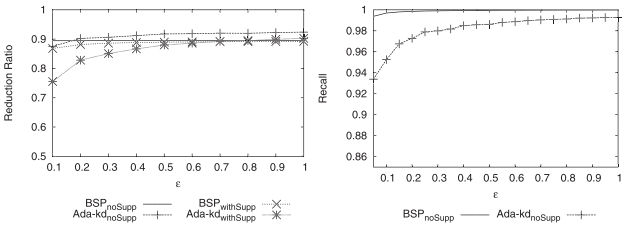

Fig. 5. (a) Reduction ratio, (b) Recall with varying differential privacy parameter, ϵ.

Proof. From Algorithm 5, it is easy to see that each partition ST_i has the same region and size as $T_i \in view_{QP}^{\Pi}$. Furthermore, under the assumption that a semantically secure encryption scheme is used, we arrive at the indistinguishability of encrypted records output by the simulator with those of the view (see [13] for details). □

7 Experimental Results

We performed experiments using the real-world $Census-income$ data set from the UCI Machine Learning Repository.[10] After combining training and test data sets, we removed records with missing attribute values. Remaining 142,521 records were shuffled and partitioned into d_1, d_2, d_3 containing 47,507 records each. We built data sets D_1 and D_2 as $d_1 \cup d_2$ and $d_2 \cup d_3$, respectively. Regardless of the matching thresholds, records in $D_1 \cap D_2 = d_2$ match.

The following quasi-identifying attributes were considered for record matching: *age, education, marital status, race*, and *sex*. By default, the decision rule contains only three attributes, $\{age, education, marital\ status\}$. For categorical attributes, we set the matching threshold θ to 0 and use Hamming distance. For numerical attributes, after normalization to $[0, 1]$, the matching threshold is set to 0.05 and the distance is measured by euclidean distance.

During the security analysis of differentially private hybrid solutions in Section 6.1, we showed that in certain settings, revealing the number of suppressed records could be problematic in terms of privacy protection. Consequently, we assume that the data holder parties decide whether or not to involve suppressed records into the *SMC* phase. When the decision is to release suppressed records, we refer to the corresponding partitioning method as $P_{withSupp}$. When the decision is to ignore, we label P as P_{noSupp}.

Among these two formulations, it is easy to see that $RR(P_{withSupp}) \leq RR(P_{noSupp})$ always holds. Because, in addition to all record pairs matched by P_{noSupp}, $P_{withSupp}$ also considers the suppressed records. On the other hand, P_{noSupp} suffers from reduced recall: $Recall(P_{withSupp}) \geq Recall(P_{noSupp})$. Any matching record pair that contains a suppressed record cannot be identified by P_{noSupp} regardless of the number of allowed SMC protocol executions. Therefore, we provide two sets of RR measurements over differentially private partitioning methods; with and without suppressed records. Also, for methods P_{noSupp} that ignore suppressed records, we measure the maximum loss in recall separately in each experiment scenario. Methods labeled as $P_{withSupp}$ are excluded from these results because when SMC_budget is not restricted or $SMC\_budget \geq RR(P_{withSupp})$ any method $P_{withSupp}$ attains 100 percent recall. Effects of SMC_budget are discussed in Section 7.5.

10. http://www.ics.uci.edu/~mlearn/MLRepository.html.

7.1 Privacy Parameters

In this scenario, we present the effects of privacy parameters on reduction ratio and recall. Our secondary aim is to determine the default values of privacy parameters. For each parameter, the value that offers the strongest privacy protection and attains around 85 percent reduction ratio is selected. This setting ensures fair comparison of different privacy definitions in the other experiments and requires the following configuration: $(k = 2,048)$-anonymity (see Fig. 4a), and $(\epsilon = 0.1)$-differential privacy (see Fig. 5a).

The default height of BSP and Adaptive-kd trees are set to eight. We used generalization hierarchies of equal-width leaf nodes. The hierarchies had depths 5, 4, 3, 2, and 2 for age, $education$, $marital\ status$, $race$, and sex, respectively.

7.1.1 k-Anonymity

Fig. 4a presents the results over Datafly [9], Incognito [34] and Mondrian [35]. Reduction ratio deteriorates as k increases because the input has to be further generalized. Mondrian is the least affected due to its flexibility in arranging generalized values dynamically based on distribution of records. Given an input data set and the parameter k, Incognito identifies all full-domain anonymizations and selects the "best" heuristically. The original description of the algorithm [34] favors an anonymization that is highest in the generalization lattice. When there are multiple valid anonymizations at the same level, we break ties by choosing the one that produces the largest number of equivalences.

As shown by the sharp downward spikes of Fig. 4a, this naive strategy has some drawbacks. Without any coordination between the data holders, Incognito may generalize entirely different quasi-identifier attributes of data sets T and V. For example, T would be generalized on age while V is generalized on $education$. Consequently, the blocking decision rule cannot operate efficiently and reduction ratio declines considerably. Below, we propose a simple heuristic that solves this problem:

1. Sort quasi-identifier attributes QI_i with respect to matching thresholds θ_i in descending order,
2. In this order, associate with each QI_i a generalization penalty $P_i = 2^i$,
3. Calculate the penalty for entry $e : \langle E_1, \ldots, E_d \rangle$ as

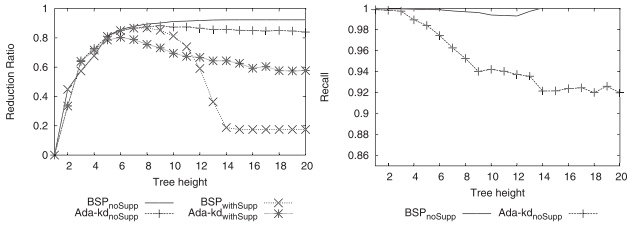

Fig. 6. (a) Reduction ratio, (b) Recall with varying statistical index height.

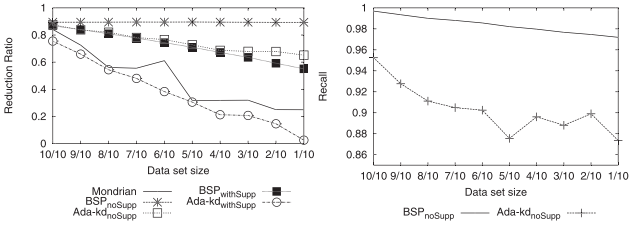

Fig. 7. (a) Reduction ratio, (b) Recall with varying data set size.

$$Penalty(e) = \sum_{i=1}^{d} P_i \times E_i,$$

4. Select the entry with minimum penalty.

Step 2 assigns smaller penalty to attributes with larger matching thresholds because they can tolerate higher levels of generalization. Step 3 allows building an ordering between various entries based on attribute-penalties. Through step 4, both parties are forced to select the "best" anonymization similarly and with respect to the ordering obtained at step 3. As seen in Fig. 4b, the spikes in reduction ratio are no longer observed.

7.1.2 ϵ-Differential Privacy

The parameter ϵ is inversely related to the privacy guarantees of differential privacy. In other words, at lower values of ϵ, Laplace noise of larger magnitude is added to query responses. Consequently, as ϵ increases, blocking becomes less efficient and both reduction ratio and recall increase.

RR measurements are depicted in Fig. 5a. Adaptive-kd trees initially perform worse than BSP-trees due to higher sensitivity. At larger values of ϵ, this disadvantage of Adaptive-kd trees is overcome by the flexible partitioning choices compared to BSP-trees.

Due to the additional cost of matching suppressed records, the reduction ratio of $BSP_{withSupp}$ (respectively $Ada - kd_{withSupp}$) is higher than BSP_{noSupp} (respectively $Ada - kd_{noSupp}$). However, the difference is more pronounced between $Ada - kd_{withSupp}$ and $Ada - kd_{noSupp}$ because of larger added noise implied by higher sensitivity.

Fig. 5b depicts the recall measurements. As ϵ increases, the number of suppressed records declines in both data sets. Consequently, more matching record pairs can be identified and recall improves. Again, the recall of $Ada - kd_{noSupp}$ is lower due to higher sensitivity.

7.2 Tree Height, h

Our reduction ratio measurements with varying height of BSP and Adaptive-kd trees are provided in Fig. 6a. For all partitioning methods, the reduction ratio initially improves as the trees grow deeper. But above some threshold value, building even deeper trees damages the efficiency of blocking over all methods except BSP_{noSupp}.

The number of partitions generated by space partitioning trees is bounded from above by 2^h. At small values of h, partitions cover a large volume of space and contain a significant number of records. Blocking at such low granularity is inefficient. Higher granularity partitions of deeper trees perform better in the blocking step. That is why up to a threshold reduction ratio improves.

Height h that maximizes reduction ratio depends on two parameters: sensitivity of the partitioning algorithm and granularity of the partition regions. As h increases, partitions cover smaller volumes and contain fewer records. However, sensitivity either does not change (i.e., BSP-trees) or increases (i.e., Adaptive-kd trees). In both cases, while partition sizes decline, magnitude of the added noise does not. Therefore, the ratio of fake and suppressed records to actual records increases, and reduction ratio deteriorates. Since sensitivity of the BSP-tree algorithm is less than the Adaptive-kd algorithm, this occurs at a larger value of h.

Notice that reduction ratio of BSP-trees does not change after $h = 14$. At $h = 15$, the tree height saturates, i.e., there is no allowable split. The same situation occurs with Adaptive-kd trees as well, but in this case h still affects the results: larger h implies higher sensitivity, which in turn translates to more noise and lower reduction ratio.

Negative effects of increasing tree height are always more pronounced with methods that consider suppressed records. Comparing $Ada - kd_{noSupp}$ and $Ada - kd_{withSupp}$, we observe that $Ada - kd_{noSupp}$ suffers only from increased positive noise (i.e., fake records) while $Ada - kd_{withSupp}$ suffers from both positive and negative noise.

Another interesting observation is the decrease in $RR(BSP_{withSupp})$ compared to $RR(Ada - kd_{withSupp})$. Adaptive-kd trees determine partition regions with respect to the distribution of records. Even for a large number of partitions, each partition contains a fair amount of records. Partitions of BSP-trees are independent of the data and most partitions contain very few records. Accordingly, suppressed records accumulate much faster in comparison to Adaptive-kd trees. Yet, most of these suppressed records are fake records derived according to Step 14 of Algorithm 4.

This is also supported by recall measurements of Fig. 6b. BSP_{noSupp} attains higher recall because less original records are suppressed in comparison to $Ada - kd_{noSupp}$. Partitions of Adaptive-kd trees contain more records and are perturbed with larger noise, therefore more matching record pairs are lost to the suppression process.

7.3 Data Set Size

In this scenario, sizes of input data sets are varied. Fractions on the x-axis of Fig. 7 indicate the proportion of the preprocessed data set from which T and V are built. Thus, $10/10$ implies the entire *Census-income* data set was used.

Decreasing the data set size without changing the privacy parameters decreases the reduction ratio. Smaller data set sizes force k-anonymity to generalize the input data set much more harshly. The reduction ratio of Mondrian keeps declining as the data set gets smaller. We exclude

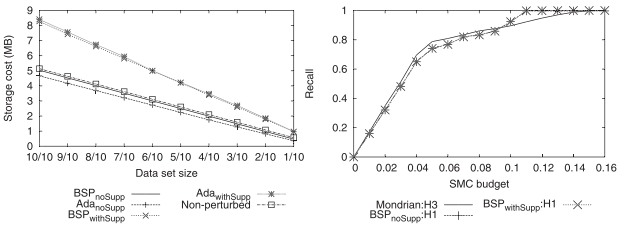

Fig. 8. (a) Index storage costs with varying data set size, (b) Recall of best heuristics.

Datafly and Incognito because Mondrian performs consistently better as discussed in Section 7.1.1.

In this scenario, the height of BSP and Adaptive-kd trees, and ϵ are fixed. As the data set shrinks in size, the magnitude of the added noise does not change. Yet, partitions contain less records and reduction ratio deteriorates.

Similarly, the recall of BSP_{noSupp} and $Ada-kd_{noSupp}$ decreases with data set size as shown in Fig. 7b. $Ada-kd_{noSupp}$ performs worse in terms of recall due to higher sensitivity and larger noise.

Next, we measure the storage costs of the proposed technique with varying data set size. The results provided in Fig. 8a indicate that as input data sets grow linearly in size, the space required for an individual index increases almost linearly. This is expected, since the number of data records is much larger than the number of index nodes. Also note that including suppressed records increases storage costs. We observe that $Ada-kd_{noSupp}$ is more storage-efficient than BSP_{noSupp} because $Ada-kd_{noSupp}$ suppresses more records (as also indicated by recall measurements of Fig. 7b). On the other hand, $Ada-kd_{withSupp}$ is slightly less efficient than $BSP_{withSupp}$, because $Ada-kd_{withSupp}$ introduces more fake records due to higher sensitivity of the query set. A nonperturbed version of either index at the specified height consumes around the same space as BSP_{noSupp} because the number of suppressed records almost equal that of fake records and sensitivity of a BSP-tree is very low.

7.4 Number of Attributes

In a separate experiment scenario, number of attributes were varied. According to our results, using an attribute set of size $d: 1 \leq d \leq 5$ that consists of the first d elements of $\{age, education, marital\ status, race, sex\}$ does not affect reduction ratio and recall significantly. Therefore, we omit these results due to space limitations.

7.5 Selection Heuristics

Next, we provide the results with varying SMC_budget and different selection heuristics. At $SMC\_budget = 0$, SMC operations are not allowed and recall is close to 0. Increasing the budget allows more private evaluations of the decision rule and recall improves. Except BSP_{noSupp} and $Ada-kd_{noSupp}$, which cannot match suppressed records, every method under every heuristic reaches 100 percent recall when $SMC\_budget \geq 1-RR$.

For each of the three selection heuristics proposed in Section 4.3.3, we experimented with k-anonymity and differential privacy methods ($BSP_{withSupp}$, $Ada-kd_{withSupp}$, BSP_{noSupp}, and $Ada-kd_{noSupp}$). This setup yields 15 series of measurements. To prevent overcrowded presentation, only the results of the best performing (method, heuristic) pairs within each of these groups are reported in Fig. 8b. We assumed that a method performs better if the area under its recall-SMC budget curve is larger.

Among various anonymization methods, heuristic H_3 (i.e., max. intersection volume first) coupled with Mondrian outperforms the other anonymization methods. In the figure, recall improves almost linearly with the number of SMC protocol invocations. BSP-trees attain higher recall than Adaptive-kd because they achieve higher reduction ratio. Among the three heuristics, the highest recall by BSP-trees is achieved through H_1 (i.e., min. comparison cost first). Since all partition regions are of the same volume, with BSP-trees, H_2 (i.e., min. volume partition first) cannot distinguish one partition pair from another and performs poorly. H_3 has similar problems: two partitions either entirely overlap or do not intersect at all.

In the comparison of the best selection heuristics over each group of partitioning methods (see Fig. 8b), we observe that H_3 with Mondrian initially performs better than H_1 with BSP-trees. However, at around $SMC\_budget = 0.095$, the recall of the BSP-tree method exceeds the anonymization-based Mondrian. Notice that BSP_{noSupp} and $BSP_{withSupp}$ are aligned due to low sensitivity of the partitioning method.

8 Conclusion and Future Work

In this work, we proposed a novel approach that combines sanitization methods and cryptographic methods to solve the private record matching problem. Our method allows participants to tradeoff between accuracy, privacy, and costs. Empirical analysis of the proposed methods performed on real-world data indicates that the hybrid approach attains significant savings in costs even at considerably high levels of privacy protection.

As future work, we will extend our existing solution to handle alphanumeric attributes (e.g., address information). Another promising area of future research might be extending the idea of hybrid approaches to other privacy preserving data mining tasks. We believe that the hybrid approach could provide substantial performance improvements for privacy preserving distributed data mining protocols.

Acknowledgments

This work was partially supported by Air Force Office of Scientific Research MURI Grant FA9550-08-1-0265, National Institutes of Health Grant 1R01LM009989, US National Science Foundation (NSF) Grant Career-CNS-0845803, and NSF Grants CNS-0964350, CNS-1016343, CNS-1016722.

References

[1] A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios, "Duplicate Record Detection: A Survey," *IEEE Trans. Knowledge and Database Eng.*, vol. 19, no. 1, pp. 1-16, Jan. 2007.

[2] C. Clifton, M. Kantarciolu, A. Doan, G. Schadow, J. Vaidya, A. Elmagarmid, and D. Suciu, "Privacy-Preserving Data Integration and Sharing," *Proc. Ninth ACM SIGMOD Workshop Research Issues in Data Mining and Knowledge Discovery (DMKD '04)*, pp. 19-26, 2004.

[3] C. Quantin, H. Bouzelat, F. Allaert, A. Benhamiche, J. Faivre, and L. Dusserre, "How to Ensure Data Security of an Epidemiological Follow-Up: Quality Assessment of an Anonymous Record Linkage Procedure," *Int'l J. Medical Informatics,* vol. 49, no. 1, pp. 117-122, 1998.

[4] T. Churces and P. Christen, "Some Methods for Blindfolded Record Linkage," *Medical Informatics and Decision Making,* vol. 4, no. 9, 2004.

[5] M. Scannapieco, I. Figotin, E. Bertino, and A.K. Elmagarmid, "Privacy Preserving Schema and Data Matching," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 653-664, 2007.

[6] A. Al-Lawati, D. Lee, and P. McDaniel, "Blocking-Aware Private Record Linkage," *Proc. Second Int'l Workshop Information Quality in Information Systems (IQIS '05),* pp. 59-68, 2005.

[7] R. Agrawal, A. Evfimievski, and R. Srikant, "Information Sharing across Private Databases," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 86-97, 2003.

[8] B.C.M. Fung, K. Wang, and P.S. Yu, "Top-Down Specialization for Information and Privacy Preservation," *Proc. 21st Int'l Conf. Data Eng. (ICDE '05),* pp. 205-216, 2005.

[9] L. Sweeney, "k-Anonymity: A Model for Protecting Privacy," *Int'l J. Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 10, no. 5, pp. 557-570, 2002.

[10] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-Diversity: Privacy Beyond K-Anonymity," *Proc. 22nd Int'l Conf. Data Eng. (ICDE '06),* p. 24, 2006.

[11] R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 439-450, 2000.

[12] C. Dwork, "Differential Privacy," *Proc. Int'l Colloquium Automata, Languages and Programming (ICALP '02),* pp. 1-12, 2006.

[13] O. Goldreich, "General Cryptographic Protocols," *The Foundations of Cryptography,* vol. 2, Cambridge Univ. Press, 2004.

[14] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the Privacy Preserving Properties of Random Data Perturbation Techniques," *Proc. IEEE Third Int'l Conf. Data Mining (ICDM '03),* pp. 99-106, 2003.

[15] A.C. Yao, "Protocols for Secure Computation," *Proc. IEEE Symp. Foundations of Computer Science (CS),* pp. 160-164, 1982.

[16] J. Vaidya and C. Clifton, "Secure Set Intersection Cardinality with Application to Association Rule Mining," *J. Computer Security,* vol. 13, no. 4, pp. 593-622, 2005.

[17] M.J. Freedman, K. Nissim, and B. Pinkas, "Efficient Private Matching and Set Intersection," *Proc. Eurocrypt,* 2004.

[18] L. Kissner and D. Song, "Privacy-Preserving Set Operations," *Proc. CRYPTO,* pp. 241-257, 2005.

[19] R. Agrawal, D. Asonov, M. Kantarcioglu, and Y. Li, "Sovereign Joins," *Proc. 22nd Int'l Conf. Data Eng. (ICDE '06),* pp. 26-37, 2006.

[20] M.G. Elfeky, A.K. Elmagarmid, and V.S. Verykios, "TAILOR: A Record Linkage Tool Box," *Proc. 18th Int'l Conf. Data Eng. (ICDE '02),* pp. 17-28, 2002.

[21] R. Schnell, T. Bachteler, and J. Reiher, "Privacy-Preserving Record Linkage Using Bloom Filters," *BMC Medical Informatics and Decision Making,* vol. 9, no. 1, p. 41, Aug. 2009.

[22] C.M. O'Keefe, M. Yung, L. Gu, and R. Baxter, "Privacy-Preserving Data Linkage Protocols," *Proc. ACM Workshop Privacy in the Electronic Soc. (WPES '04),* pp. 94-102, 2004.

[23] V.S. Verykios, A. Karakasidis, and V.K. Mitrogiannis, "Privacy Preserving Record Linkage Approaches," *Int'l J. Data Mining, Modelling and Management,* vol. 1, pp. 206-221, 2009.

[24] M.J. Atallah, F. Kerschbaum, and W. Du, "Secure and Private Sequence Comparisons," *Proc. ACM Workshop Privacy in the Electronic Soc. (WPES '03),* pp. 39-44, 2003.

[25] F. Emekci, D. Agrawal, A.E. Abbadi, and A. Gulbeden, "Privacy Preserving Query Processing Using Third Parties," *Proc. 22nd Int'l Conf. Data Eng. (ICDE '06),* 2006.

[26] M. Kantarcioglu, A. Inan, W. Jiang, and B. Malin, "Formal Anonymity Models for Efficient Privacy-Preserving Joins," *Data and Knowledge Eng.,* vol. 68, no. 11, pp. 1206-1223, 2009.

[27] A. Inan, M. Kantarcioglu, E. Bertino, and M. Scannapieco, "A Hybrid Approach to Private Record Linkage," *Proc. IEEE 24th Int'l Conf. Data Eng. (ICDE '08),* pp. 496-505, 2008.

[28] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino, "Private Record Matching Using Differential Privacy," *Proc. 13th Int'l Conf. Extending Database Technology (EDBT '10),* pp. 123-134, 2010.

[29] N. Li, T. Li, and S. Venkatasubramanian, "T-Closeness: Privacy Beyond K-Anonymity and L-Diversity," *Proc. IEEE 23rd Int'l Conf. Data Eng. (ICDE '07),* pp. 106-115, 2007.

[30] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating Noise to Sensitivity in Private Data Analysis," *Proc. Third Theory Computing Conf. (TCC),* pp. 265-284, 2006.

[31] X. Xiao and Y. Tao, "Output Perturbation with Query Relaxation," *Proc. VLDB Endowment,* vol. 1, no. 1, pp. 857-869, 2008.

[32] C. Clifton, M. Kantarcoğlu, X. Lin, J. Vaidya, and M. Zhu, "Tools for Privacy Preserving Distributed Data Mining," *SIGKDD Explorations,* vol. 4, no. 2, pp. 28-34, Jan. 2003.

[33] X. Xiao and Y. Tao, "Anatomy: Simple and Effective Privacy Preservation," *Proc. 32nd Int'l Conf. Very Large Databases,* pp. 139-150, 2006.

[34] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan, "Incognito: Efficient Full-Domain k-Anonymity," *Proc. SIGMOD Int'l Conf. Management of Data,* pp. 49-60, 2005.

[35] K. Le Fevre, D.J. DeWitt, and R. Ramakrishnan, "Mondrian Multidimensional k-Anonymity," *Proc. 22nd Int'l Conf. Data Eng. (ICDE '06),* pp. 25-35, 2006.

Ali Inan received the BS and MS degrees in computer science from Sabanci University, Istanbul, Turkey and the PhD degree in computer science from the University of Texas at Dallas. He is an assistant professor of the Computer Engineering Department at Isik University, Istanbul, Turkey. His research interests include database systems, data mining and security, and privacy issues related to the management of data. He is particularly interested in privacy preserving data mining and data integration. He has published various papers in international journals and proceedings of international conferences.

Murat Kantarcioglu received the BS degree in computer engineering from Middle East Technical University, and the MS and PhD degrees in computer science from Purdue University. He is an associate professor in the Computer Science Department and the director of the Data Security and Privacy Lab at the University of Texas at Dallas. He is a recipient of US National Science Foundation (NSF) Career award and Purdue CERIAS Diamond award for academic excellence. His research focuses on creating technologies that can efficiently extract useful information from any data without sacrificing privacy or security. He has published more than 75 papers in peer reviewed Journals and conferences. His research has been supported by grants from NSF, AFOSR, ONR, NSA, and NIH. Some of his research work has been covered by the media outlets such as Boston Globe, ABC News, etc., and has received two best paper awards. He is a member of the IEEE.

Gabriel Ghinita is an assistant professor with the Department of Computer Science, University of Massachusetts, Boston. His research interests lie in the area of data security and privacy, with focus on privacy-preserving transformation of microdata, private queries in location-based services and privacy-preserving sharing of sensitive data sets. Prior to joining University of Massachusetts, he was a research associate with the Cyber Center at Purdue University, and a member of the Center for Education and Research in Information Assurance and Security (CERIAS). He served as reviewer for top Journals and conferences such as *IEEE TPDS*, *IEEE TKDE*, *IEEE TMC*, *VLDBJ*, *VLDB*, WWW, and ICDE. He is a member of the IEEE.

Elisa Bertino is a professor of computer science at Purdue University, and serves as a research director of the Center for Education and Research in Information Assurance and Security (CERIAS) and interim director of Cyber Center (Discovery Park). Previously, she was a faculty member and department head at the Department of Computer Science and Communication of the University of Milan. Her main research interests include security, privacy, digital identity management systems, database systems, distributed systems, and multimedia systems. She is currently serving as chair of the ACM SIGSAC and as a member of the editorial board of the following International Journals: *IEEE Security & Privacy, IEEE Transactions on Service Computing, ACM Transactions on Web*. She also served as editor in chief of the *VLDB J.* and editorial board member of *ACM TISSEC* and *IEEE TDSC*. She coauthored the book "*Identity Management - Concepts, Technologies, and Systems*." She received the 2002 IEEE Computer Society Technical Achievement Award for outstanding contributions to database systems and database security and advanced data management systems and the 2005 IEEE Computer Society Tsutomu Kanai Award for pioneering and innovative research contributions to secure distributed systems. She is a fellow of the IEEE and the ACM.