Scalable privacy-preserving linking of multiple databases using counting Bloom filters

Dinusha Vatsalan
Research School of Computer Science
The Australian National University
Canberra ACT 0200, Australia
Email: dinusha.vatsalan@anu.edu.au

Peter Christen
Research School of Computer Science,
The Australian National University
Canberra ACT 0200, Australia
Email: peter.christen@anu.edu.au

Erhard Rahm
Universität Leipzig,
Institut für Informatik
04109, Leipzig, Germany
Email: rahm@informatik.uni-leipzig.de

***Abstract*—The integration, mining, and analysis of person-specific data can provide enormous opportunities for organizations, governments, and researchers to leverage today's massive data collections. However, the use of personal or otherwise sensitive data also raises concerns about the privacy, confidentiality, and potential discrimination of people. Privacy-preserving record linkage (PPRL) is a growing research area that aims at integrating sensitive information from multiple disparate databases held by different organizations while preserving the privacy of the individuals in these databases by not revealing their identities and thereby preventing re-identification and discrimination. PPRL approaches are increasingly required in many real-world application areas ranging from healthcare to national security. Previous approaches to PPRL have mostly focused on linking only two databases. Scaling PPRL to several databases is an open challenge since privacy threats as well as the computation and communication costs increase significantly with the number of databases involved. We thus propose a new encoding method of sensitive data based on *Counting Bloom Filters* (CBF) to improve privacy for multi-party PPRL (MP-PPRL). We investigate optimizations to reduce computation and communication costs for CBF-based MP-PPRL. Our empirical evaluation with real datasets demonstrates the viability of our approach in terms of scalability, linkage quality, and privacy.**

I. Introduction

A wide range of real-world applications, ranging from government services, healthcare, crime and fraud detection, to national security, require person-related data from multiple sources held by different organizations to be integrated or linked. Integrated data can then be used for data mining and analytics to empower efficient and quality decision making with rich data [1]. For example, health outbreak systems that allow the early detection of infectious diseases before they spread widely around a country or even worldwide require the integration of health data, travel data, and consumed drug data. National security agencies require to integrate data from law enforcement agencies, Internet service providers, and financial institutions to enable the accurate identification of crime and fraud, or terrorism suspects.

In the absence of unique entity identifiers in the databases to be linked, personal identifying attributes (such as names and addresses) need to be used for the linkage. Known as quasi-identifiers (QIDs) [2], such values are in general well correlated with entities to allow accurate linkage. Using such personal information across organizations, however, often leads to privacy and confidentiality concerns. This problem has been addressed through the development of 'privacy-preserving record linkage' (PPRL) [3] techniques.

PPRL aims to conduct linkage using only masked (encoded) QIDs without requiring any sensitive or confidential information to be exchanged and revealed between the organizations involved in a linkage. Generally, masking transforms the original QID values such that a specific functional relationship exists between the original and the masked QID values [3].

While there have been many different approaches proposed for PPRL, most work thus far has concentrated on linking data from only two sources [3]. As the two examples described above show, linking data from multiple sources (MP-PPRL) is however commonly required in practical applications.

A challenge with MP-PPRL is that the computation and communication complexities increase significantly with multiple parties. The number of naïve (all-to-all) comparisons required across p databases of n records each is n^p. Addressing this complexity challenge, private indexing/blocking techniques are used to reduce the number of candidate record sets, which are then compared in detail and classified into matches and non-matches [1], [3]. However, with increasing p existing private blocking techniques are not sufficient to reduce the number of comparisons, as has been empirically studied [4], [5]. Moreover, with multiple parties the risk of collusion increases, where a sub-set of parties collude in order to learn about other parties' sensitive data.

We propose the use of *Counting Bloom Filter* (CBF), which is a variation of Bloom Filter (BF) [3], to enable efficient and approximate matching for MP-PPRL. BFs are bit vectors into which values are hash-mapped (as we will describe in Section III). CBFs, on the other hand, are integer vectors that contain a count value in each position. Multiple BFs can be summarized as a single CBF by applying vector addition between BFs using a secure summation protocol [6].

Calculating the similarity of a set of records using a single CBF instead of multiple BFs provides increased privacy without compromising the linkage quality, because a CBF contains

	Categorical data	**String data**
Blocking / Indexing	[7]	[4], [8]
Exact matching	[9], [10], [11],	[12], [13]
Approximate matching	-	**[5]**

TABLE I
EXISTING MP-PPRL APPROACHES.

only the summary information of multiple records rather than the bit values of individual records as in BFs. This privacy aspect of CBFs has so far not been utilized in PPRL.

To overcome the scalability and privacy challenges of MP-PPRL, we introduce an efficient CBF-based communication pattern that significantly reduces the number of record comparisons in contrast to naïve all-to-all comparisons. It thereby improves the scalability while also improving the privacy (reducing the likelihood of collusions) by arranging parties into several groups (rings) and distributing computations among parties. We propose two variations of extended secure summation protocols for improved privacy against collusion among the database owners: (a) homomorphic encryption-based and (b) salting-based (using random seed integers). We theoretically analyze our protocol in terms of complexity and privacy, and empirically evaluate and compare with two baseline approaches using large voter registration datasets.

Outline: In the following section we review the literature and in Section III we describe the preliminaries. In Section IV we propose our protocol for MP-PPRL based on CBFs, where in Section IV-A we propose two extended secure summation protocols to improve privacy of our approach, and in Section IV-B we introduce a communication pattern to improve scalability (as well as privacy) of our approach. We analyze our protocol in terms of complexity, linkage quality, and privacy in Section V, and in Section VI we validate these analyses through an empirical study. Finally, we summarize and discuss future research directions in Section VII.

II. RELATED WORK

Various techniques have been proposed in the literature to tackle the problem of PPRL [3]. However, most of them consider linking two databases only, and only few approaches have been proposed for PPRL on multiple databases. As categorized in Table I, the main drawbacks of the small number of existing MP-PPRL approaches is that either they (1) only allow blocking (not matching), (2) only support exact matching (which classifies record sets as matches if their masked QIDs are exactly the same), or (3) are applicable to QIDs of categorical data only (however, linkage using QIDs of string data, such as names and addresses, is required in many real applications [3], [5]).

A Secure Multi-party Computation (SMC)-based approach using an oblivious transfer protocol was proposed by O'Keefe et al. [12] for PPRL on multiple databases. While provably secure, the approach can only perform *exact matching* of masked values (i.e. variations and errors in the QIDs are not considered) and it is *computationally expensive* compared to efficient perturbation-based privacy techniques, such as BFs and k-anonymity [3]. Kantarcioglu et al. [10] introduced a multi-party approach to perform secure equi-join (*exact matching*) on k-anonymous databases.

A MP-PPRL approach for *categorical values* was proposed by Mohammed et al. [11], where a top-down generalization is performed to provide k-anonymous privacy and the generalized blocks are then classified into matches and non-matches using the C4.5 decision tree classifier. Another efficient multi-party approach for *categorical data* was recently proposed by Karapiperis et al. [9] using a Count-Min sketch data structure. Sketches are used to summarize the local set of elements which are then intersected to provide a global synopsis using homomorphic operations, secure summation, and symmetric noise addition privacy techniques.

As we will describe in detail in Section VI, Lai et al. [13] developed an efficient BF-based approach for MP-PPRL. However, the approach performs only *exact matching*. This approach has been adapted by Vatsalan and Christen [5] for *approximate matching* in MP-PPRL.

Several approximate comparison functions for calculating similarities of *pairs* of string and other data types have been proposed for PPRL by adapting existing comparison functions. A secure version of the Levenshtein edit distance was proposed by Karakasidis and Verykios [14] using CBFs. However, developing privacy-preserving comparison functions for multiple (more than two) records has only recently been considered by Vatsalan and Christen [5] using the Dice coefficient similarity.

The scalability problem in MP-PPRL has been addressed recently by Ranbaduge et al. who developed a family of multi-party private blocking approaches [4], [8]. However, the *number of comparisons* required for multi-party linkage *remains very large* ($O(b \times (n/b)^p)$ for linking p databases each containing n records and b blocks) even with existing private blocking approaches employed [4], [5]. Therefore, efficient and advanced multi-party filtering and communication patterns need to be developed in order to make PPRL scalable and viable in practical applications.

III. DEFINITION AND PRELIMINARIES

In this section, we define the problem and describe preliminaries required for our approach. We assume p database owners P_1, P_2, $\cdots$, P_p with their respective databases $\mathbf{D}_1$, $\mathbf{D}_2$, $\cdots$, $\mathbf{D}_p$ (containing sensitive or confidential identifying information) participate in the PPRL process under the honest-but-curious (HBC) adversary model [3]. In this model, parties are assumed to follow the protocol without deviating or sending false information while being curious to learn about other parties' data. Note that the HBC model does not prevent parties from colluding among them to learn about other parties' data [15]. We quantify the risk of collusion in MP-PPRL and the reduction of risk by our extended secure summation protocols and proposed communication pattern in Section V.

A dedicated trusted linkage unit (honest broker) is commonly used in PPRL approaches in the literature [3]. A linkage

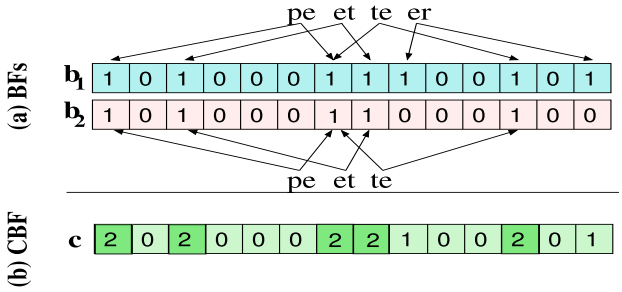

Fig. 1. Similarity calculation of two QID values ('peter' and 'pete') masked using (a) BFs and (b) CBF (with $l = 14$, $k = 2$ and $q = 2$).

unit (LU) is an external party that conducts linkage of the masked records sent to it by the database owners [3]. For our protocol we assume such a LU is available to conduct the linkage. We also assume a set of QID attributes A, which will be used for the linkage, is common to databases $\mathbf{D}_1, \cdots, \mathbf{D}_p$. We formally define the problem of PPRL on multiple databases as follows.

Definition 3.1: **MP-PPRL**: Assume $P_1, \ldots, P_p$ are the p owners (parties) of the databases $\mathbf{D}_1, \ldots, \mathbf{D}_p$, respectively. They wish to determine which of their records $R_{1,i} \in \mathbf{D}_1$, $R_{2,j} \in \mathbf{D}_2$, ..., $R_{p,k} \in \mathbf{D}_p$ match (correspond to the same entity) based on the (masked) QIDs of these records according to a decision model $C(\cdot)$ that classifies record sets ($R_{1,i}$, $R_{2,j}$, ..., $R_{p,k}$) into one of the two classes $\mathbf{M}$ of matches and $\mathbf{U}$ of non-matches. A party does not wish to reveal its actual records with any other party. The parties are however prepared to disclose to each other, or to an external party, the actual values of some selected attributes of the record sets that are classified into $\mathbf{M}$ to allow analysis.

A. Protocol building blocks

Bloom filter (BF) encoding: A BF $\mathbf{b}_i$ is a bit array data structure of length l bits where all bits are initially set to 0. k independent hash functions, $h_1, h_2, \ldots, h_k$, each with range $1, \ldots l$, are used to map the elements s in a set $\mathbf{S}$ into the BF by setting the bit positions $h_j(s)$, with $1 \leq j \leq k$, to 1. BF encoding has been used as an efficient masking technique in several PPRL solutions [5], [13], [16].

The set $\mathbf{S}$ of q-grams (sub-strings of length q) of QID values of each record in the databases to be linked can be hash-mapped into a BF. These BFs are then sent to a LU that calculates the similarity of BFs [16] in order to classify them as matches or non-matches. The BF encoding of two QID values 'peter' and 'pete' into $l = 14$ bits long BFs using $k = 2$ hash functions is shown in Figure 1(a).

Dice coefficient similarity: Any set-based similarity function (such as Overlap, Jaccard, and Dice coefficient) can be used to calculate the similarity of pairs or sets of (multiple) BFs. The Dice coefficient has been used for comparing BFs since it is insensitive to many matching zeros (bit positions to which no q-grams are hash-mapped) in long BFs [16].

Definition 3.2: **Dice similarity**: The Dice coefficient similarity of p BFs ($\mathbf{b}_1, \cdots, \mathbf{b}_p$) is:

$$Dice\_sim(\mathbf{b}_1, \cdots, \mathbf{b}_p) = \frac{p \times z}{\sum_{i=1}^{p} x_i}, \quad (1)$$

where z is the number of common bit positions that are set to 1 in all p BFs (common 1-bits), and x_i is the number of bit positions set to 1 in $\mathbf{b}_i$ (1-bits), $1 \leq i \leq p$.

Figure 1(a) illustrates the Dice coefficient similarity calculation of two QID values 'peter' and 'pete' masked into BFs.

Secure summation: A secure summation protocol [6] can be used to securely sum the input values of multiple parties ($p \geq 3$), such that no party learns the individual values of other parties, but only the summed value.

The input can either be a single numerical value or a vector of numerical values. For example, p numerical vectors ($\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p$) from p parties ($P_1, P_2, \cdots, P_p$), respectively, can be securely summed by using a random numerical vector $\mathbf{r}$ which is generated by the LU (and kept secret) and then sent to the first party P_1. The first party P_1 that receives $\mathbf{r}$ calculates the vector summation of $x_1 = \mathbf{r} + \mathbf{v}_1$ and sends this to P_2, which adds its own $\mathbf{v}_2$ to the received sum and sends $x_2 = x_1 + \mathbf{v}_2$ to P_3. This process is repeated until the last party sends $x_p = x_{p-1} + \mathbf{v}_p$ back to LU, which then subtracts $\mathbf{r}$ from the summed value x_p to calculate the summation of the p vectors. The protocol employs a ring-based communication pattern over all parties to learn $\sum_{i=1}^{p} \mathbf{v}_i$ but no party P_i learns the values $\mathbf{v}_j$ (with $i \neq j$) of the other parties.

IV. CBF-based MP-PPRL algorithm

In this section, we describe how Counting Bloom filters (CBFs) can be used for efficiently calculating similarities (approximate matching) of QID values between a set of multiple (two or more) records in MP-PPRL. A CBF $\mathbf{c}$ of p ($p > 1$) BFs is an integer array data structure of the same length (l) as the BFs. It contains in each position β, with $1 \leq \beta \leq l$, the count of values in the position β over a set of p BFs, such that $\mathbf{c}[\beta] = \sum_{i=1}^{p} \mathbf{b}_i[\beta]$, where $\mathbf{c}[\beta]$ is the count value in bit position β of the CBF $\mathbf{c}$ and $\mathbf{b}_i[\beta] \in [0, 1]$ is the value in the bit position β of BF $\mathbf{b}_i$. Given p BFs (bit vectors) $\mathbf{b}_i$ with $1 \leq i \leq p$, the CBF $\mathbf{c}$ can be generated by applying a vector addition operation between the bit vectors such that $\mathbf{c} = \sum_i \mathbf{b}_i$.

Proposition 4.1: The Dice coefficient similarity of p BFs can be calculated given only a CBF $\mathbf{c}$ as:

$$Dice\_sim(\mathbf{c}) = \frac{p \times |\{\beta : \mathbf{c}[\beta] = p, 1 \leq \beta \leq l\}|}{\sum_{\beta=1}^{l} \mathbf{c}[\beta]}. \quad (2)$$

Proof: The Dice coefficient similarity of p BFs ($\mathbf{b}_1, \cdots, \mathbf{b}_p$) is determined by the sum of 1-bits ($\sum_{i=1}^{p} x_i$) in the denominator and the number of common 1-bits (z) in all p BFs in the nominator of Eq. (1). The number of 1-bits in a BF $\mathbf{b}_i$ is $x_i = \mathbf{b}_i[1] + \mathbf{b}_i[2] + \cdots + \mathbf{b}_i[l]$, with $1 \leq i \leq p$. Therefore $\sum_{i=1}^{p} x_i = \sum_{i=1}^{p} \mathbf{b}_i[1] + \mathbf{b}_i[2] + \cdots + \mathbf{b}_i[l]$. The sum of values in all bit positions of the CBF is $\sum_{\beta=1}^{l} \mathbf{c}[\beta] =$

$\sum_{\beta=1}^{l} \mathbf{b}_1[\beta] + \mathbf{b}_2[\beta] + \cdots + \mathbf{b}_p[\beta]$ which is equal to $\sum_{i=1}^{p} x_i$. Further, if a bit position β ($1 \leq \beta \leq l$) contains 1 in all p BFs, i.e. $\forall_{i=1}^{p} \mathbf{b}_i[\beta] = 1$, then $\mathbf{c}[\beta] = \sum_{i=1}^{p} \mathbf{b}_i[\beta] = p$. Therefore, $z = |\{\beta \in \mathbf{c} : \mathbf{c}[\beta] = p|\}$. ■

If the LU only receives the CBF $\mathbf{c}$ that contains the summed values in the bit positions of p BFs instead of the actual p BFs, then the LU can calculate the Dice coefficient of p BFs using Eq. (2) without learning any information about the individual bit positions of each party. Figure 1(b) shows an example of similarity calculation of two BFs $\mathbf{b}_1$ and $\mathbf{b}_2$ using CBF $\mathbf{c}$. The information gained from a CBF is less than what can be gained from p BFs (i.e. CBFs provide increased privacy compared to BFs), as we theoretically and empirically validate in Sections V and VI, respectively.

The vector summation of individual BFs from p different parties to construct a CBF $\mathbf{c}$ needs to be calculated securely among these parties, such that no party learns the individual values in the other parties' bit positions, and the LU only learns the summed values $\forall_{\beta=1}^{l} \mathbf{c}[\beta]$. However, the basic secure summation (*BSS*), as described in Section III-A, is susceptible to collusion among the parties to learn the values of another party. For example, parties P_1 and P_3 can collude to learn party P_2's values. In Section IV-A, we propose two variations of extended secure summation protocols for improved privacy against the risk of collusion among parties.

Our protocol allows efficient, approximate, and private linking of multiple databases from p (≥ 3) parties. We first describe a naïve protocol (which we refer as *NAI*) based on CBFs, and in Section IV-B we propose an improved communication pattern to make the protocol more scalable with increasing number of large databases.

- **Step 1:** The parties agree upon the following parameter values: the BF length l; the k hashing functions $h_1, \ldots, h_k$ to be used; the length (in characters) of grams q; a minimum similarity threshold value s_t ($0 \leq s_t \leq 1$), above which a set of records is classified as a match; a private blocking function $block(\cdot)$; the blocking keys [1] B used for blocking; and a set of QID attributes A used for the linkage.
- **Step 2:** Each party P_i ($1 \leq i \leq p$) individually applies a private blocking function [3] $block(\cdot)$ to reduce the number of candidate sets of records $\mathbf{C}$ (from $\prod_i^p n_i$, where $n_i = |\mathbf{D}_i|$ is the number of records in $\mathbf{D}_i$ held by party P_i), which otherwise becomes prohibitive even for moderate n_i and p. The $block(\cdot)$ function groups records according to their blocking key values (BKVs) [7] and only records with the same BKV (i.e. records in the same block) from different parties are then compared and classified using our protocol. Any private blocking technique [4], [8] can be used as the $block(\cdot)$ function.
- **Step 3:** Each party P_i hash-maps the q-gram values of QIDs A of each of its n_i records in its respective database $\mathbf{D}_i$ into n_i BFs (one BF per record in $\mathbf{D}_i$) of length l using the hash functions $h_1, \ldots, h_k$ to generate $\mathbf{D}_i^M$. It is crucial to set the BF related parameters in an optimal way that balances all three properties of PPRL (complexity, quality, and privacy) [5], [16]. We further discuss parameter setting for BFs used in our protocol in Section V.
- **Step 4:** In the next step, the parties initiate a secure summation protocol to securely perform vector addition between their BFs in order to generate a CBF for each set of candidate records $\mathbf{cs} \in \mathbf{C}$. A candidate record set $\mathbf{cs}$ contains p masked records (BFs), one from each of the p parties, and $\mathbf{C}$ contains in total n_i^p such cs (exponential complexity). The secure summation is initiated by an external linkage unit LU that provides a vector (of length l) of random values.

  The LU sends a random vector $\mathbf{R}[\mathbf{cs}]$ to a party (assume P_1) for each candidate record set $\mathbf{cs} \in \mathbf{C}$ to sum with the party's BF vector $\mathbf{b}_i \in \mathbf{D}_i^M$ ($i = 1$) by applying a vector addition operation. The summed vectors $\forall_{\mathbf{cs} \in \mathbf{C}} \mathbf{R}[\mathbf{cs}] + \mathbf{b}_i$ are then sent to party P_{i+1}. Party P_i, $2 \leq i \leq p$, receives the summed vectors from P_{i-1} and adds its BF vector $\mathbf{b}_i \in \mathbf{D}_i^M$ to each candidate set $\mathbf{cs} \in \mathbf{C}$ and sends the summed vectors to the next party P_{i+1}. This process is repeated until the last party (i.e. P_p) adds its $\mathbf{b}_p$ vector to each received summed vector $\mathbf{R}[\mathbf{cs}] + \sum_{i=1}^{p-1} \mathbf{b}_i$ from party P_{p-1} and sends the final summed vector back to the LU for each $\mathbf{cs}$.
- **Step 5:** Finally, the LU that has provided the random vectors $\mathbf{R}$ generates the CBF $\mathbf{c} = (\mathbf{R}[\mathbf{cs}] + \sum_{i=1}^{p} \mathbf{b}_i) - \mathbf{R}[\mathbf{cs}]$ for each candidate set $\mathbf{cs} \in \mathbf{C}$. The LU then calculates the Dice coefficient similarity of each resulting CBF $\mathbf{c}$ following Eq. (2) to classify the compared sets of records (BFs) within a block into matches and non-matches based on the similarity threshold s_t. The final matching sets of records will then be used by the database owners or an external party for further analysis using certain set of attribute values.

A. Extended secure summation

The basic secure summation protocol (*BSS*) described in Section III-A is susceptible to collusion risk by the parties involved, as we will discuss in detail in Section V. In order to overcome this risk (to improve privacy), we propose two extended secure summation protocols.

- Homomorphic-based secure summation (*HSS*): The partially homomorphic Paillier cryptosystem [17] is a reliable secure multi-party computation (SMC) technique for performing secure joint computation among several parties. In the *HSS* approach a pair of private and public keys is used for encrypting and decrypting the individual BF values. Successive encryption of the same value using the same public key generates different encrypted values with high probability, and decrypting the encrypted values using a private key returns the correct original value. The public key is known to all parties while the private key is known only to the LU. Each party P_i receives summed vectors containing encrypted values to which P_i adds its encrypted $\mathbf{b}_i$ vector (using the public key)

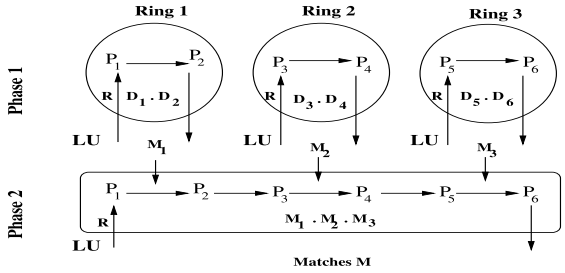

Fig. 2. Ring-by-ring (*RBR*) communication pattern for comparison of CBFs from $p = 6$ parties, as described in Section IV-B. In this example, the number of parties per ring is $r = 2$.

and sends the encrypted summed vectors to the next party. Without knowing the private key a party P_i cannot decrypt the received vectors and therefore colluding with a party to learn another party's $\mathbf{b}_j$ (with $i \neq j$) would be impossible.

- Salting-based (using random seed integers) secure summation (*SSS*): The *HSS* approach provides a secure solution compared to the *BSS* approach against collusion attacks at the cost of an excessive computation overhead, making it not scalable to linking multiple large databases. Therefore, we propose the *SSS* approach to provide security against collusion attacks in an efficient way. Salting has been used to defend against dictionary attacks on one-way hash functions where an additional string is concatenated with a value to be encrypted [18]. We adopt a similar concept in the *SSS* approach where the salting key is an additional random integer used by each party P_i individually when performing the secure summation. The salting key generated and used by each party is sent only to the LU and therefore a party P_i's BF values cannot be identified by means of collusion among the parties, as without knowing the salting key of P_i its BF values cannot be learned. Since the salting keys are random integer values, performing secure summation of BFs with the salting keys does not add any additional computation and communication overhead, except the communication of salting keys from parties to the LU.

B. Improved communication

The number of candidate record sets to be compared for multi-party linkage in the naïve method (*NAI*), as described above, is exponential in the number of parties and their dataset sizes, which is prohibitively large to be practically feasible (even when using existing private blocking and filtering approaches) [5]. In this section, we propose a ring-by-ring (*RBR*) communication pattern, as illustrated in Figure 2, for improved scalability and privacy of our approach based on CBFs for multi-party scenarios.

The main idea of the improved communication pattern is to exploit the facts that most candidate sets are true non-matches (due to the class imbalance problem of record linkage [7]), and that a true matching set must have a high similarity between any sub-set of records in that set. Hence for MP-PPRL applications with many parties it is promising to determine partial matches for a sub-set of parties and consider additional parties only for these partial matches.

Algorithm 1: Comparison of CBFs using ring-by-ring (*RBR*) communication.

Input:
- $\mathbf{D}_i^M$: Party P_i's BFs, $1 \leq i \leq p$
- r: Ring sizes, with $2 \leq r \leq p/2$
- s_t: Minimum similarity threshold to classify record sets

Output:
- $\mathbf{M}$: Matching record sets

1: $rings = group([P_1, P_2, \cdots, P_p], r)$ // Group parties
2: **for** $ring \in rings$ **do**:
3: $\quad \mathbf{C}_{ring} = sec\_sum([\forall_{P_i \in ring}\ \mathbf{D}_i^M])$ // Secure summation
4: $\quad$ **for** $\mathbf{cs} \in \mathbf{C}_{ring}$ **do**:
5: $\quad\quad$ **if** $Dice\_sim(\mathbf{cs}) \geq s_t$ **then**: // Eq. (2)
6: $\quad\quad\quad \mathbf{M}_{ring}.append(\mathbf{cs})$ // $\mathbf{M}$ in this ring
7: $\mathbf{C} = sec\_sum([\forall_{ring \in rings}\ \mathbf{M}_{ring}])$ // Secure summation
8: **for** $\mathbf{cs} \in \mathbf{C}$ **do**:
9: $\quad$ **if** $Dice\_sim(\mathbf{cs}) \geq s_t$ **then**: // Eq. (2)
10: $\quad\quad \mathbf{M}.append(\mathbf{cs})$ // $\mathbf{M}$ in all rings

The parties are first arranged into rings of size r, with at least 2 rings and a minimum size of each ring being $r = 2$ (i.e. $2 \leq r \leq p/2$). The value of r needs to be carefully chosen, as it has a trade-off between scalability (complexity) and privacy. The higher the value for r is, the better the privacy of the protocol because the resulting CBFs are more difficult to exploit with an inference attack, as will be discussed in Section V. On the other hand, higher values of r results in lower scalability for larger datasets across many parties because the number of comparisons required per ring exponentially increases with the ring size r.

The *NAI* approach can be considered as a special case of *RBR* with only one ring of size $r = p$. With several smaller rings ($r \leq p/2$), each ring reduces the number of candidates to matches only and thereby enables a significant filtering in the number of candidate record sets.

The *RBR* method, as detailed in Algorithm IV-B, consists of two phases. In the first phase (lines 2-6), the parties in each ring individually perform secure summation on their sets of masked records (BFs) using the $sec\_sum(\cdot)$ function to generate the CBFs $\mathbf{C}_{ring}$ (line 3). The similarity of each CBF $\mathbf{c} \in \mathbf{C}_{ring}$ is calculated using Eq. (2) to identify matches $\mathbf{M}_{ring}$ in each ring (lines 4-6). In the second phase (lines 7-10), the rings perform secure summation among them on the matches identified in each ring $\mathbf{M}_{ring}$ in order to identify matches $\mathbf{M}$ from all p parties.

Every ring in the first phase can employ a different set of BF parameters to reduce the possibility of collusion between a set of parties in different rings. In the second phase, all parties then have to agree on another set of parameters for BF encodings of the matches identified in the different rings in the first phase. In addition, the rings in the first phase can be processed independently and in parallel in a distributed environment making the *RBR* scalable to larger dataset sizes.

V. Analysis of the protocol

In this section we analyze our MP-PPRL protocol in terms of complexity, privacy, and linkage quality.

Complexity: We assume p parties participate in the protocol, each having a database of n records. In step 1 of our protocol, the agreement of parameters has a constant communication complexity. Blocking the databases in step 2 has $O(n)$ computation and $O(p \cdot b)$ communication complexity (assuming $b \leq n$ blocks) at each party. The masking of records (with g average q-grams) into BFs of length l using k hash functions in step 3 is $O(n \cdot g \cdot k)$ at each party.

Steps 4 and 5 consist of the secure summation of BFs to calculate the CBFs of candidate sets. With the simplified assumption that all blocks are of equal size n/b, using the *NAI* method requires a total of $O(\sum_{i=1}^{p} b(n/b)^i)$ vector summations. Our extended secure summation protocols *HSS* and *SSS* aim to improve privacy at the cost of complexity overhead. The extended *HSS* protocol requires $n \cdot l \cdot p$ encrypted values (long integers of 4 bytes each) to be exchanged among the parties, while the basic secure summation (*BSS*) and *SSS* require exchanging $n \cdot l \cdot p$ short integer values (of 2 bytes each), which is more efficient compared to *HSS*. Further, the homomorphic encryption and decryption functions used in *HSS* are computationally expensive compared to simple vector addition and subtraction operations [15].

The exponential complexity limits the *NAI* multi-party linkage to a small p or a large number of small blocks (small n/b). (i.e. p or n/b has to be small). Our improved *RBR* method reduces the complexity significantly (depending on the number of parties per ring, r). In general, the complexity is reduced from exponential growth with p down to $max(r, p/r)$. Assuming each ring has r parties, the first phase of *RBR* requires $b(n/b)^r$ candidate sets to be processed in each of the p/r rings, and the second phase compares the $b(n/b)$ matching sets from each ring, leading to $O(\sum_{i=1}^{r} b(n/b)^i \cdot (p/r) + \sum_{i=1}^{p/r} b(n/b)^i)$. Overall, the complexity of the *NAI* method is $O(b(n/b)^p)$, and the *RBR* method is $O(max(b(n/b)^r \cdot p/r, b(n/b)^{p/r})$. This theoretical analysis shows that the proposed communication method *RBR* is computationally more efficient compared to the *NAI* method.

Privacy: As with most of the existing PPRL approaches, we assume that all parties follow the honest-but-curious (semi-honest) adversary model [3], where the parties follow the protocol while being curious to learn the other parties' data by means of inference attacks on input data [2] or collusion [3]. To analyze the privacy against inference attacks, we discuss what the parties can learn from the data exchanged among them during the protocol.

Communication occurs among the parties in step 1 of our protocol (as described in Section IV) where they agree on parameter settings, and in steps 4 and 5 where they participate in a secure summation protocol. The agreement of parameters (in step 1) does not reveal any sensitive information about the underlying data. Secure summation (steps 4 and 5) involves the exchange of masked data where the parties communicate the partial and full summations of their BFs among them in step 4 and to the LU in step 5, respectively, to calculate the CBFs of the candidate sets and their similarities. The secure summation protocol is secure against inference attacks by the LU because the random value used by a party is unknown to the LU [6]. We assume a trusted LU is available, which does not collude with any parties, as is commonly used in many practical PPRL applications [19].

However, collusion among the database owners is a privacy risk in the basic secure summation protocol where a set of parties can collude to learn the BF of another party using their received summation values. To overcome this problem, in Section IV-A we have proposed two extended secure summation protocols, *HSS* and *SSS*. The *HSS* protocol uses homomorphic encryptions for secure summation which makes the protocol more secure because without knowing the private key (which is only known to the LU) identifying a party's BF values by means of collusion will not be successful. However, this protocol encrypts each integer value in a BF (in total l values for each BF) into a hash key (long integers) and thus incurs a very large communication overhead making the protocol not viable for linkage of multiple large databases. The *SSS* protocol similarly makes the protocol more secure by adding additional integer values as salting keys by each party individually (known only to the LU) in the secure summation, such that without knowing the salting key value collusion among parties to learn a party's BF is not possible. Compared to *HSS*, the *SSS* approach does not incur any expensive communication overhead as the salting keys are small integer values.

Compared to calculating similarities of records using multiple (p) BFs, a single CBF improves privacy by making the inference of individual BFs and thus their q-grams mapped into them more difficult. An inference attack allows an adversary to map a list of known values from a global dataset (e.g. q-grams or attribute values from a public telephone directory) to the encoded values (BFs or CBF) using background information (such as frequency) [2], [20]. The only information that can be learned from such an inference attack using a CBF $\mathbf{c}$ of a set of x BFs (summed over x parties, where either $x = p$ in *NAI* and $x = r$ in *RBR*) is if a bit position in $\mathbf{c}$ is either 0 or x which means it is set to 0 or 1 in the BFs from all x parties, respectively.

Proposition 5.1: The probability of re-identifying the original (unencoded) values of x ($x > 1$) records R_1, R_2, $\cdots$, R_x given a single CBF $\mathbf{c}$ is smaller than the probability of re-identifying the original values of R_i given their x individual BFs $\mathbf{b}_i$ (with $1 \leq i \leq x$),

$$\forall_{i=1}^{x} Pr(R_i|\mathbf{c}) < Pr(R_i|\mathbf{b}_i). \tag{3}$$

Proof: Assume the number of matching (unencoded) values to a masked BF pattern $\mathbf{b}_i$ from an inference attack is n_g, where $n_g = 1$ in the worst case (i.e. a one-to-one mapping exists between the masked BF $\mathbf{b}_i$ and the original unencoded value of R_i [2]). The probability of re-identifying the original value R_i given a BF in the worst case is therefore $Pr(R_i|\mathbf{b}_i) = 1/n_g = 1.0$. However, a CBF $\mathbf{c}$ represents x BFs and thus in the worst case $n_g = x$, which leads to a

	Runtime (sec)	Size (MBytes)	F-measure
BSS	34.03	47.11	1.0
HSS	60557.93	1257.06	1.0
SSS	34.03	47.15	1.0

TABLE II
RESULTS OF SECURE SUMMATION PROTOCOLS ($p = 3$, $n = 5000$).

maximum of $Pr(R_i|\mathbf{c}) = 1/x$ with $x > 1$ (when $x = 1$, $\mathbf{c} \equiv \mathbf{b}_1$). Hence, $\forall_{i=1}^{x} Pr(R_i|\mathbf{c}) < Pr(R_i|\mathbf{b}_i)$. ■

We will empirically evaluate and compare the privacy of CBFs and BFs in Section VI. A larger r in the *RBR* method (i.e. a larger x) results in a smaller probability of re-identification of $1/x$ (improved privacy) by the LU at the cost of more record comparisons. Further, using different BF encodings by different rings in our *RBR* method improves privacy by reducing the collusion possibilities compared to the *NAI* method. More specifically, when p parties are involved in the linkage, the maximum number of possible combinations for collusion in the *NAI* method is $\theta_N = p \cdot (p-1)$, while in the *RBR* it is $\theta_R = p/r \cdot r \cdot (r-1) = p \cdot (r-1)$. For example, $p = 9$ results in $\theta_N = 72$ and $\theta_R = 18$ collusion possibilities for the *NAI* and *RBR* (with $r = 3$) methods, respectively. The extended secure summation protocols (both *HSS* and *SSS*) have zero collusion risk among the parties (i.e. $\theta_H = 0$ and $\theta_S = 0$. In addition, hardening BF techniques [18], such as hash-mapping of several QID values from each record into one compound BF [21], [16], can be applied to make an inference attack more difficult.

Linkage quality: Our protocol supports approximate matching of QID values, in that data errors and variations are taken into account depending upon the minimum similarity threshold s_t used. The Dice coefficient similarity of p records calculated using their respective p BFs is the same as the similarity calculated using a single CBF $\mathbf{c}$, as we have proven in Proposition 4.1.

The quality of BF-based masking depends on the BF parameterization [5]. For a given BF length, l, and the number of elements (e.g. q-grams) to be inserted into the BF, g, the optimal number of hash functions, k, that minimizes the false positive rate f (of a collision of two different q-grams being mapped to the same bit position), is $k = l/g \cdot ln(2)$, leading to a false positive rate of $f = (1/2^{ln(2)})^{l/g}$.

While k and l determine the computational aspects of BF masking, linkage quality and privacy will be determined by the false positive rate f. A higher value for f will mean a larger number of false matches and thus lower linkage quality [16]. The false positive rate f in a CBF $\mathbf{c}$ is the summation of the false positive rates f_i introduced in the corresponding p BFs $\mathbf{b}_i$, with $1 \leq i \leq p$. Therefore, a CBF provides increased privacy compared to a BF while resulting in the same number of false matches as with its corresponding p BFs (i.e. results in the same similarity with increased privacy).

VI. EXPERIMENTAL EVALUATION

We have conducted experiments on the datasets used in [5] that were extracted from the North Carolina Voters Registration (NCVR) database (from ftp://alt.ncsbe.gov/data/). These datasets contain 5,000 to 1,000,000 records for 3, 5, 7, and 10 parties, with 50% of records occurring in all parties (i.e. 50% of records are true matches). We used datasets with 0% and 20% (No-mod and 20% -mod in figures) corrupted values in the matching records.

We used an exact matching BF-based PPRL approach (*EM-BF*) [13] and an approximate matching BF-based PPRL approach (*AM-BF*) [5] as baseline methods. Since other existing approaches for MP-PPRL (as reviewed in Section II and categorized in Table I) either allow blocking only, support exact matching only, or are applicable to categorical data only, we cannot directly compare them with our approach.

In the *EM-BF* approach, a conjuncted BF (that contains common 1-bits) is jointly calculated such that each party processes a particular segment in the BF. Each party then checks its BFs with the conjuncted BF to classify them as matches or not. The *AM-BF* approach adapts the *EM-BF* approach for approximate matching in PPRL [5]. Once the conjuncted BF segments are computed by the respective parties, a secure summation protocol is used by the parties to securely sum the number of common as well as all 1-bits to calculate the Dice similarity (Eq. (1)).

We implemented all approaches in Python (version 2.7), and ran all experiments on a server with 2.4 GHz CPUs, 128 GBytes of main memory and running Ubuntu 14.04. The programs and test datasets are available from the authors. We used first name, last name, city, and zipcode as the QIDs. We set the parameters for all three approaches as $l = 500$, $k = 20$, $q = 2$, $s_t = 0.8$, $block(\cdot) = Soundex(\cdot)$ [1] on first and last names, and $p = [3, 5, 7, 10]$. For *RBR*, we set $r \geq 3$.

We evaluated the scalability using *runtime* and the *number of comparisons* required for the linkage, and the linkage quality using the *F-measure* [3]. In line with other work in PPRL [4], [5], we evaluated privacy using the average disclosure risk measure, DR_{Mean}, based on the probability of suspicion P_s, i.e. the likelihood a masked database record in $\mathbf{D}^M$ can be matched with one or several record(s) in a large public database [22] ($DR_{Mean} = \sum(P_s)/|\mathbf{D}^M|$).

Table II shows the runtime and memory size required and the F-measure results achieved with the three secure summation protocols. As can be seen, the *HSS* requires significantly higher runtime and memory (which is not practical in real applications) to improve privacy against collusion attacks on the *BSS* without compromising the F-measure results. However, the *SSS* approach requires similar runtime and memory as the *BSS* for improving privacy against collusions with no loss in linkage quality. We therefore use *SSS* in all following experiments.

Figure 3 (a) shows the number of comparisons required between records by the *RBR* and *NAI* methods. For the *NAI* method, this number grows exponentially with the size of the

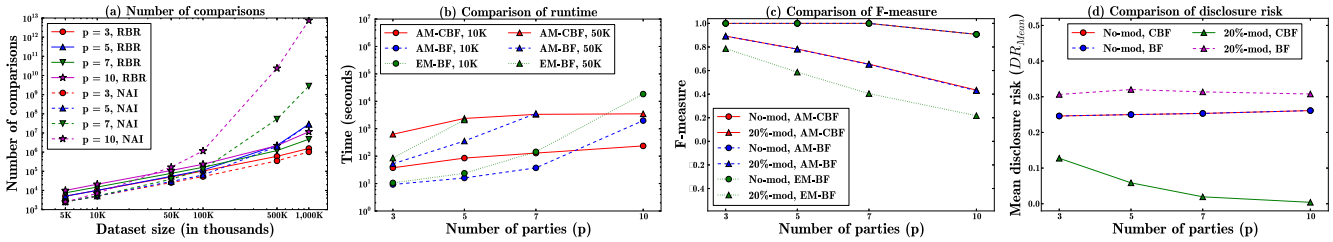

Fig. 3. (a) Number of comparisons between records, (b) runtime, (c) F-measure, and (d) privacy results for all three approaches.

datasets for increasing p. As shown in Figure 3 (b), although the baseline approaches require lower runtime for linking $p \leq 5$ datasets than our approach they are not scalable to larger p and datasets. We were unable to conduct experiments for the baseline methods with $p \geq 5$ on the 50K datasets due to the exponential increase in the number of comparisons. Our approach is scalable and requires significantly lower runtime for linking datasets from multiple parties.

As can be seen in Figure 3 (c), our approach achieves similar F-measure as *AM-BF* and outperforms *EM-BF* on modified datasets. Finally, we compared the privacy results of our CBF-based approach with the BF-based baseline approaches in Figure 3 (d). The results show that the DR_{Mean} of CBF is consistently lower (and thus privacy is better) than BF, and as expected it decreases with larger p.

VII. Conclusions and Future Work

We have presented a scalable protocol for PPRL on multiple databases with an improved communication pattern. Our approach performs approximate matching on the QID values (of string data) masked using efficient and simple privacy techniques, namely counting BFs (CBFs) and secure summation. Experiments conducted on real datasets of up-to one million records showed the scalability and improved privacy of our approach compared to two baseline approaches while achieving superior or similar linkage quality results.

In future work, we plan to investigate other improved collision resistant secure summation protocols and different approximate string comparison functions [1] to be incorporated. We also aim to develop efficient PPRL techniques for identifying matching record sets within sub-sets of parties, which is an important research problem. Another research direction would be to develop MP-PPRL approaches under other adversary models such as the covert model or accountable computing [2] (where honest parties can verify fake data from dishonest parties with high probability) to overcome the limitations of the semi-honest (HBC) adversary model. Finally, we plan to investigate improved classification techniques for MP-PPRL including relational clustering and graph-based approaches [1] which are successfully used in non-PPRL applications.

Acknowledgment

This work was partially funded by the Australian Research Council (ARC) under Discovery Projects DP130101801 and DP160101934, and Universities Australia and the German Academic Exchange Service (DAAD).

References

[1] P. Christen, *Data Matching - Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, 2012.
[2] D. Vatsalan, P. Christen, C. M. O'Keefe, and V. S. Verykios, "An evaluation framework for privacy-preserving record linkage," *JPC*, 2014.
[3] D. Vatsalan, P. Christen, and V. S. Verykios, "A taxonomy of privacy-preserving record linkage techniques," *Elsevier JIS*, vol. 38, no. 6, pp. 946–969, 2013.
[4] T. Ranbaduge, P. Christen, and D. Vatsalan, "Clustering-based scalable indexing for multi-party privacy-preserving record linkage," in *PAKDD*, 2015.
[5] D. Vatsalan and P. Christen, "Scalable privacy-preserving record linkage for multiple databases," in *ACM CIKM*, 2014.
[6] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Zhu, "Tools for privacy preserving distributed data mining," *SIGKDD Explorations*, vol. 4, no. 2, pp. 28–34, 2002.
[7] P. Christen, "A survey of indexing techniques for scalable record linkage and deduplication," *IEEE TKDE*, vol. 24, no. 9, pp. 1537–1555, 2012.
[8] T. Ranbaduge, D. Vatsalan, and P. Christen, "Hashing-based distributed multi-party blocking for privacy-preserving record linkage," in *PAKDD, Springer LNAI*, Auckland, 2016.
[9] D. Karapiperis, D. Vatsalan, V. S. Verykios, and P. Christen, "Large-scale multi-party counting set intersection using a space efficient global synopsis," in *DASFAA*, Hanoi, 2015.
[10] M. Kantarcioglu, W. Jiang, and B. Malin, "A privacy-preserving framework for integrating person-specific databases," in *PSD*, Istanbul, 2008.
[11] N. Mohammed, B. Fung, and M. Debbabi, "Anonymity meets game theory: secure data integration with malicious participants," *VLDB*, vol. 20, no. 4, pp. 567–588, 2011.
[12] C. O'Keefe, M. Yung, L. Gu, and R. Baxter, "Privacy-preserving data linkage protocols," in *ACM WPES*, Washington, 2004, pp. 94–102.
[13] P. Lai, S. Yiu, K. Chow, C. Chong, and L. Hui, "An Efficient Bloom filter based Solution for Multiparty Private Matching," in *SAM*, 2006.
[14] A. Karakasidis and V. S. Verykios, "Secure blocking+secure matching = secure record linkage," *JCSE*, vol. 5, pp. 223–235, 2011.
[15] Y. Lindell and B. Pinkas, "Secure multiparty computation for privacy-preserving data mining," *Journal of Privacy and Confidentiality*, vol. 1, no. 1, p. 5, 2009.
[16] R. Schnell, T. Bachteler, and J. Reiher, "A novel error-tolerant anonymous linking code," *German Record Linkage Center, Working Paper Series No. WP-GRLC-2011-02*, 2011.
[17] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT*. Springer, 1999, pp. 223–238.
[18] R. Schnell, "Privacy-preserving record linkage," in *Methodological Developments in Data Linkage*, H. G. Katie Harron and C. Dibben, Eds. John Wiley & Sons, 2015, pp. 201–225.
[19] S. M. Randall, A. M. Ferrante, J. H. Boyd, and J. B. Semmens, "Privacy-preserving record linkage on large real world datasets," *Journal of Biomedical Informatics*, 2013.
[20] M. Kuzu, M. Kantarcioglu, E. Durham, and B. Malin, "A constraint satisfaction cryptanalysis of Bloom filters in private record linkage," in *PETS, Springer LNCS*, vol. 6794, Waterloo, Canada, 2011, pp. 226–245.
[21] E. Durham, "A framework for accurate, efficient private record linkage," Ph.D. dissertation, Faculty of the Graduate School of Vanderbilt University, Nashville, TN, 2012.
[22] P. Christen, D. Vatsalan, and V. S. Verykios, "Challenges for privacy preservation in data integration," *ACM Journal of Data and Information Quality*, vol. 5, no. 1-2, p. 4, 2014.