Scalable Privacy-Preserving Record Linkage for Multiple Databases [*]

Dinusha Vatsalan and Peter Christen
Research School of Computer Science, College of Engineering and Computer Science,
The Australian National University, Canberra ACT 0200, Australia
{dinusha.vatsalan, peter.christen}@anu.edu.au

ABSTRACT

Privacy-preserving record linkage (PPRL) is the process of identifying records that correspond to the same real-world entities across several databases without revealing any sensitive information about these entities. Various techniques have been developed to tackle the problem of PPRL, with the majority of them only considering linking two databases. However, in many real-world applications data from more than two sources need to be linked. In this paper we consider the problem of linking data from three or more sources in an efficient and secure way. We propose a protocol that combines the use of Bloom filters, secure summation, and Dice coefficient similarity calculation with the aim to identify all records held by the different data sources that have a similarity above a certain threshold. Our protocol is secure in that no party learns any sensitive information about the other parties' data, but all parties learn which of their records have a high similarity with records held by the other parties. We evaluate our protocol on a large dataset showing the scalability, linkage quality, and privacy of our protocol.

Categories and Subject Descriptors

H2.7 [**Database Management**]: Database Administration-*Security, integrity, and protection*

General Terms

Algorithms, Experimentation, Security

Keywords

Record linkage; privacy; security; Bloom filter; multi-party.

1. INTRODUCTION

Linking records from different databases with the aim to improve data quality or enrich data for further analysis and mining is occurring in an increasing number of application areas including healthcare, government services, crime and fraud detection, and business applications [1]. The analysis of data linked across organizations can, for example, facilitate the detection of infectious diseases early before they spread widely around a country or worldwide, or enable the accurate identification of fraud, crime, or terrorism suspects [13]. These applications require data from several organizations, such as human health data, consumed drug data, and animal health data for the first of the above examples [3]; while the second above example requires data from law enforcement agencies, Internet service providers, the police, as well as financial institutions.

Today, record linkage not only faces computational challenges due to the increasing size of datasets and quality challenges due to the presence of real-world data errors, but also privacy and confidentiality challenges due to growing privacy concerns by the public. In the absence of unique entity identifiers in the databases that are linked, personal identifying attributes (such as names, addresses, gender, and dates of birth) need to be used for the linkage. Known as quasi-identifiers (QIDs) [12], values in such attributes are in general sufficiently well correlated with entities to allow accurate linkage. Using such personal information however often leads to privacy and confidentiality concerns.

The privacy challenges posed in the record linkage process led to the development of techniques that facilitate 'privacy-preserving record linkage' (PPRL) [13]. PPRL tackles the problem of how to identify records that refer to the same entities in different databases such that only masked (encoded) QIDs have to be revealed. Generally, the original data are transformed (using some encoding function) such that a specific functional relationship exists between the original and the masked data [12], without compromising the privacy and confidentiality of the entities represented by these data.

Many different approaches have been proposed for PPRL [13], but most of these are limited to linking data from two sources. As the example applications described above have shown, linking data from several sources is however commonly required. We propose an efficient solution for PPRL across multiple parties. While existing multi-party PPRL techniques [5, 7, 8, 9, 10] either perform only exact matching or use computationally expensive privacy techniques, the novelty of our solution is that it supports approximate matching based on two efficient privacy techniques: Bloom filters [11] and secure summation [6]. We conduct an empirical study on a large real dataset to validate the scalability, linkage quality, and privacy of our solution.

[*]Funded by the Australian Research Council under Discovery Project DP130101801.

CIKM'14, November 3–7, 2014, Shanghai, China.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2598-1/14/11 ...$15.00.
http://dx.doi.org/10.1145/2661829.2661875.

2. RELATED WORK

Various techniques have been developed to address the PPRL research problem [13], but few among these have considered PPRL on multiple databases. The first approach to PPRL [10] links multiple databases by comparing the hash-encoded values (using one-way secure hash algorithms) from all data sources by using a third party. However, this approach only performs exact matching (i.e. a single variation in a QID results in a completely different hash-encoded value). A secure equi-join protocol for multiple database tables was proposed in [5] for exact matching, and a secure multi-party computation based approach using an oblivious transfer protocol was presented in [9] for PPRL on multiple databases. While provably secure, the approach is computationally expensive compared to perturbation-based privacy techniques. Recently, a multi-party PPRL approach for approximate matching of categorical values based on k-anonymity and game-theoretic concepts was proposed [8].

An efficient multi-party PPRL approach for exact matching using Bloom filters was introduced by Lai et al. [7]. In this approach, database values are first converted into a Bloom filter bit array. Each party then partitions its Bloom filter into segments according to the number of parties involved in the linkage, and sends these segments to the corresponding other parties. The segments received by a party are combined using a conjunction (logical AND) operation. The resulting combined Bloom filter segments are then exchanged between the parties. Each party checks its own full Bloom filter with the final result, and if the membership test is successful then the value is considered to be a match. Though the cost of this approach is low since the computation is completely distributed between the parties and the processing of Bloom filters is very fast, the approach can only perform exact matching. As we describe next, we use Lai et al.'s [7] multi-party Bloom filter based approach as a building block for our approximate matching solution.

3. MULTI-PARTY LINKAGE PROTOCOL

We now describe our approach to securely and efficiently link databases from three or more parties. We use the following notation: P is the number of parties involved in our protocol, where each party p_i holds a database D_i containing sensitive or confidential identifying information. Database D_i contains $N_i = |D_i|$ records. We assume a set of QID attributes A, which will be used for the linkage, is common to all these databases. Our protocol will calculate the similarity between sets of records using the values in A. We next describe the building blocks of our protocol, then explain each of the steps in our protocol, and finally analyze the complexity and privacy characteristics of our protocol.

3.1 Protocol Building Blocks

Bloom filter encoding: A Bloom filter 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 each of the elements in a set S into the Bloom filter by setting k corresponding bit positions to 1. Bloom filters are one efficient perturbation-based privacy technique that has successfully been used in several privacy-preserving solutions [4, 11, 12].

Schnell et al. [11] were the first to propose a method for approximate matching in PPRL of two databases using Bloom filters. In their work, as in our protocol, the q-grams (substrings of length q) of attribute values in A of each record in the databases to be linked are hash-mapped into Bloom filters using k independent hash functions. The Bloom filters are then sent to a third party that calculates the Dice coefficient [1] similarity of pairs of Bloom filters.

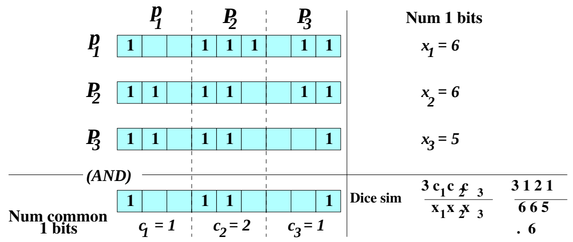

Figure 1: Dice similarity calculation of three Bloom filters (BFs) across three parties. Rows illustrate the BFs generated by the three parties, while columns show which party holds which BF segments.

Dice coefficient: Any set-based similarity function can be used to calculate the similarity of pairs or sets of Bloom filters. The Dice coefficient has been used for matching of Bloom filters, since it is insensitive to many matching zeros in long Bloom filters [11]. We calculate the Dice coefficient similarity of P Bloom filters $b_1, \cdots, b_P$ as:

$$Dice\_sim(b_1, \cdots, b_P) = \frac{P \times c}{\sum_{i=1}^{P} x_i} \qquad (1)$$

where c is the number of common bit positions that are set to 1 in all P Bloom filters (common 1-bits), and x_i is the number of bit positions set to 1 in b_i (1-bits), $1 \leq i \leq P$.

Multi-party Bloom filter matching: In our protocol the calculation of the number of common 1-bits (c) is distributed among the parties, such that $c = \sum_{i=1}^{P} c_i$. Bloom filters are split into P segments and each party sends its segments to the corresponding other parties. The parties then individually calculate the number of common 1-bits c_i in their respective segments of the Bloom filters they receive from the other parties for all sets of records. As example, the distributed Dice coefficient calculation of three Bloom filters from three parties is illustrated in Figure 1.

Secure summation: Once each of the P parties has calculated its c_i and x_i values for each set of Bloom filters, a secure summation protocol [6] can be applied to calculate $c = \sum_{i=1}^{P} c_i$ and $x = \sum_{i=1}^{P} x_i$ in a secure way (in order to calculate the Dice similarity of a set of Bloom filters). This protocol uses two vectors R_c and R_x (of length equal to the number of sets) of large random numbers (values larger than l) to hide the actual sensitive values c_i and x_i, and employs a ring-based communication pattern over all parties which allows each party to learn the final values (c and x) but no party to learn the sensitive values of the other parties.

3.2 Protocol Steps

We divide the steps of our protocol into three phases: (1) data preparation, (2) distributed matching, and (3) similarity calculation. In the initial data preparation phase,

1. the parties agree upon a bit array length l; k hashing functions $h_1, \ldots, h_k$; the length (in characters) of grams q; a minimum Dice similarity threshold value,

s_t, above which a set of records is classified as a match; and a set of blocking keys [1] and QID attributes A;

2. each party p_i individually applies a private blocking function [13] to reduce the number of candidate sets of records (from $\prod_i^P N_i$); and
3. each party p_i hash-maps the q-gram values of A of each of its N_i records into N_i Bloom filters of length l using the hash functions $h_1, \ldots, h_k$.

In the distributed matching phase, for all records and their Bloom filters in each block, each party p_i:

4. segments its Bloom filters into P equal sized segments and sends the j^{th} segment to party p_j, with $1 \leq j \leq P$ and $j \neq i$;
5. receives the i^{th} segment of Bloom filters from all other parties p_j, with $1 \leq j \leq P$ and $j \neq i$;
6. applies a logical conjunction (AND) on each set of Bloom filter segments ($b_1^i \wedge b_2^i \wedge \cdots \wedge b_P^i$) for each record set combination within the block; and
7. calculates the number of common 1-bits (c_i) and the total number of 1-bits in its own Bloom filter (x_i) for each set of Bloom filter segments.

Finally, in the similarity calculation phase, the parties:

8. use the secure summation protocol to exchange the values of c_i and x_i for each set of Bloom filters for the calculation of the sums c and x, respectively; and
9. calculate the Dice coefficient similarity of each set of Bloom filters using c and x following Equation 1 to classify the compared sets of records into matches and non-matches based on the similarity threshold s_t.

3.3 Complexity and Privacy Analysis

We assume P parties participate in the protocol, each having a database of N records, and we assume $B \leq N$ blocks are being formed by each party [1]. In the first phase, agreement of parameters has a constant communication complexity, and blocking the databases has $O(N)$ computation complexity. Finding the intersection of blocks from all parties has a communication complexity of $O(P\,B)$ and a computation complexity of $O(B \; log \; B)$ at each party. The creation of Bloom filters using k hash functions for N records is $O(kN)$.

In the distributed matching phase, each party sends its Bloom filter segments (each of length l/P) to the other parties. If we assume direct communication, $P(P-1)$ messages are required in this step, each of these of size $N \times l/P$ ($O(N\,l\,P)$ total communication). With the simplified assumption that all blocks are of equal size (N/B), then in each block $(N/B)^P$ sets of Bloom filters (i.e. all candidate sets of records in a block) have to be generated and their logical conjunctions calculated, leading to a total of $O(B(N/B)^P)$ calculations. This combinatorial complexity currently limits our protocol to a small number of parties, or a large number of small blocks (i.e. N/B is small). Our main future research focus is to improve this step of our protocol by efficiently filtering non-matching record sets.

The similarity calculation phase consists of the secure summation of the calculated number of common 1-bits (c_i) and total 1-bits (x_i). This requires for each Bloom filter set two integer numbers to be sent in a ring communication (P messages) over all parties with a total communication of $O(P\,B(N/B)^P)$, followed by the distribution of the final results which is again $O(P\,B(N/B)^P)$.

To assess the privacy of our protocol, we assume all parties follow the honest-but-curious adversary model [13], in that they are curious and try to find out as much as possible about the other parties' data while following the protocol. Since calculations are distributed among the parties, each party only learns l/P bits of each other party's Bloom filters, which reduces with increasing P (and thus privacy improves with increasing P).

The values for the number of hash functions used (k) and the length of the Bloom filter (l) provide a trade-off between the linkage quality and privacy [11]. The higher the value for k/l, the higher the privacy and the lower the quality of linkage, because the number of q-grams mapped to a single bit increases, which leads to lower linkage quality but makes it more difficult for an adversary to learn the possible q-gram combinations. Hash-mapping several attribute values from each record into one compound Bloom filter [4] makes it even more difficult for an adversary to learn individual attribute values that correspond to a revealed bit pattern.

4. EXPERIMENTS AND DISCUSSION

We have implemented our proposed approach in Python (version 2.7.3), and ran all experiments on a server with 2.4 GHz CPUs, 128 GBytes of main memory and running Ubuntu 12.04. The programs and test datasets are available from the authors. Following other work in PPRL [4, 11], we set the parameters as $l = 1,000$, $k = 30$, $s_t = [0.8, 0.9]$, and $P = [3, 5, 7, 10]$. We apply a Soundex based phonetic blocking [1] to improve the scalability of our protocol.

We evaluate the scalability of our protocol measured by runtime, and the quality of the achieved linkage measured by precision and recall [1]. In line with other work in PPRL [12], we evaluate privacy using disclosure risk (DR) measures based on the probability of suspicion, i.e. the likelihood a masked database record can be matched with one or several (masked) record(s) in a publicly available global database. We show mean DR values, as well as marketeer DR values calculated as the proportion of records that match to exactly one record in the global database.

For all experiments we used the large real-world North Carolina Voter Registration database (named 'NC') as available from `ftp://alt.ncsbe.gov/data/`. We have downloaded this database every second month since October 2011 and built a compound temporal dataset that contains over 8 million records of voter's names and addresses.

To allow evaluation of our protocol with data of different sizes, different quality, and for different number of parties, we used a recently proposed data corruptor [2] to create a variety of datasets with different characteristics. From the full NC dataset we extracted sub-sets of 5,000, 10,000, 50,000, 100,000, 500,000, and 1,000,00 records for each party where the number of matching records is set to 50% (i.e. half of all selected records occur in the datasets of all parties).

To investigate how our protocol deals with dirty data (where attribute values contain errors and variations), we generated several series of datasets with one, two, or three modifications (corruptions) applied to randomly selected attribute values. These corruptions consisted of character edit operations (insert, delete, substitute, or transposition), as well as optical character recognition and phonetic modifications based on look-up tables and rules [2]. Because we generated our different datasets we know the true matching records which allows us to calculate linkage accuracy.

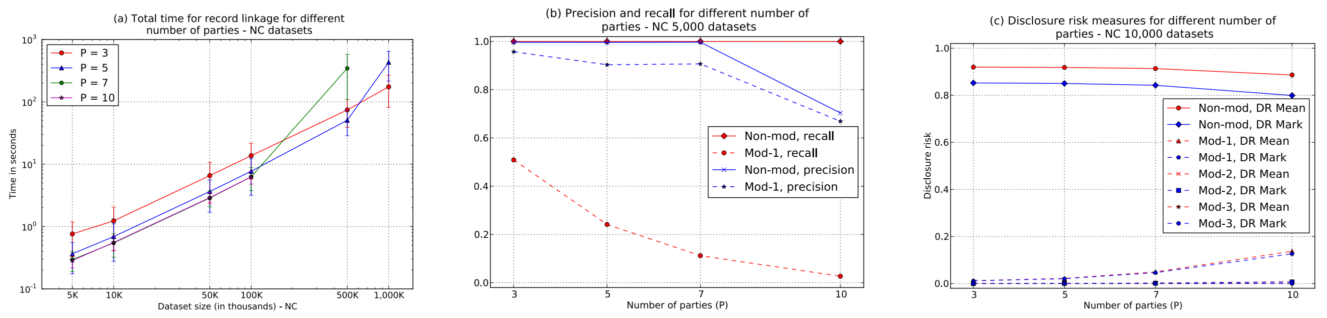

Figure 2: (a) Total time required for linkage (for one party, averaged over all parties), (b) precision and recall of the linkage of datasets with no and one modification, and (c) disclosure risk (DR) measures of privacy [12].

Figure 2 (a) shows the scalability of our approach, measured by runtime as averaged over all parties and over all variations of each dataset. Interestingly, runtime decreases with larger number of parties (P) because the Bloom filter segments at each party become shorter (l/P) and the similarity calculations are distributed among the parties.

The quality of linkage measured by precision and recall is presented in Figure 2 (b) on the NC 5,000 modified and non-modified datasets. As can be seen, precision and recall are high on the non-modified datasets. On the modified datasets the recall drops quite drastically with the number of parties. This is because when records are modified in each dataset the number of missed true matching record sets increases. In future work we will investigate similarity techniques that allow for matching records in sub-sets of parties only.

Finally, the privacy of our protocol, as measured by disclosure risk (DR) [12] of an exact matching attack using the full NC dataset as the global dataset, is shown in Figure 2 (c). As discussed in Section 3.3, DR decreases (i.e. privacy increases) with an increasing number of parties for the non-modified datasets as the Bloom filter segments become shorter and are therefore matched to more global records. Since all the records in these datasets are non-modified (an unlikely real-world situation) there exist exact matchings of records in the global datasets which leads to higher DR values. For the modified datasets (more likely in real applications), the DR values are lower and most Bloom filter segments match to no global record at all, but as these segments become shorter with more parties an increasing number of segments do match, leading to a slight increase in DR.

5. CONCLUSIONS AND FUTURE WORK

We have presented a secure protocol for PPRL across multiple parties based on Bloom filters. Our protocol identifies sets of records that have a high Dice similarity across all parties. The protocol has a communication complexity that is linear in the number of parties and the size of the databases that are linked, making the protocol scalable to applications where data from multiple parties need to be linked.

In future work, we plan to improve the scalability of our protocol by using improved private blocking or filtering approaches, and by investigating different communication patterns. A second avenue of future work will be to study linkage attacks with approximation of error bounds for privacy evaluation of our protocol. Finally, we plan to investigate improved classification techniques including relational clustering and graph-based approaches [1] which are successfully used in non-PPRL applications. Our ultimate aim is to develop techniques that allow for large databases to be linked in secure, accurate, and scalable ways across many parties, thereby facilitating data analysis and mining that currently are not feasible due to privacy and confidentiality concerns.

6. REFERENCES

[1] P. Christen. *Data Matching*. Springer, 2012.

[2] P. Christen and D. Vatsalan. Flexible and extensible generation and corruption of personal data. In *ACM CIKM*, San Francisco, 2013.

[3] C. Clifton, M. Kantarcioglu, A. Doan, G. Schadow, J. Vaidya, A. Elmagarmid, and D. Suciu. Privacy-preserving data integration and sharing. In *ACM SIGMOD Workshop DMKD*, Paris, 2004.

[4] E. A. Durham, C. Toth, M. Kuzu, M. Kantarcioglu, Y. Xue, and B. Malin. Composite Bloom filters for secure record linkage. *TKDE*, 99(PrePrints), 2013.

[5] M. Kantarcioglu, W. Jiang, and B. Malin. A privacy-preserving framework for integrating person-specific databases. In *PSD*, Istanbul, 2008.

[6] A. F. Karr, X. Lin, A. P. Sanil, and J. P. Reiter. Analysis of integrated data without data integration. *Chance*, 17(3):26–29, 2004.

[7] P. Lai, S. Yiu, K. Chow, C. Chong, and L. Hui. An Efficient Bloom filter based Solution for Multiparty Private Matching. In *SAM*, Las Vegas, 2006.

[8] N. Mohammed, B. Fung, and M. Debbabi. Anonymity meets game theory: secure data integration with malicious participants. *VLDB*, 20(4):567–588, 2011.

[9] C. M. O'Keefe, M. Yung, L. Gu, and R. Baxter. Privacy-preserving data linkage protocols. In *ACM WPES*, Washington DC, 2004.

[10] C. Quantin, H. Bouzelat, F. Allaert, and et al. How to ensure data security of an epidemiological follow-up: quality assessment of an anonymous record linkage procedure. *IJMI*, 49(1):117–122, 1998.

[11] R. Schnell, T. Bachteler, and J. Reiher. Privacy-preserving record linkage using Bloom filters. *BMC Med Inform Decis Mak*, 9(1), 2009.

[12] D. Vatsalan, P. Christen, C. M. O'Keefe, and V. S. Verykios. An evaluation framework for privacy-preserving record linkage. *JPC*, 6(1), 2014.

[13] D. Vatsalan, P. Christen, and V. S. Verykios. A taxonomy of privacy-preserving record linkage techniques. *JIS*, 38(6):946–969, 2013.