Cluster Computing (2022) 25:3641–3652
https://doi.org/10.1007/s10586-022-03590-7

An enhanced privacy-preserving record linkage approach for multiple databases

Shumin Han[1] · Derong Shen[1] · Tiezheng Nie[1] · Yue Kou[1] · Ge Yu[1]

Received: 26 May 2021 / Revised: 28 January 2022 / Accepted: 20 March 2022 / Published online: 22 April 2022

Abstract
For the purpose of research, organizations often need to share and link data that belongs to a single individual while protecting the privacy, which is referred to as privacy preserving record linkage (PPRL). Various approaches have been developed to tackle this problem, however, it is still a challenging task due to the massive amount of data, multiple data sources, and 'dirty' data. Therefore, in this paper, an enhanced approximate multi-party PPRL (MP-PPRL) approach is proposed to improve privacy, scalability, and linkage quality. For privacy, bloom filter (BF) is a better and more efficient masking techniques than others so far. Thus, the records are encoded into BFs to ensure privacy. However, BFs may be compromised through frequency-based attacks. To enhance privacy, a distributed protocol that introduces multiple linkage units (Multi-LUs) to resist frequency-based attacks is proposed. In scalability, we develop a blocking technique based on sorted nearest neighborhood (SNN) approach for clustering similar BFs across multiple databases, called *BF-SNN*, which dramatically reduces complexity. In linkage quality, a personalized threshold that varies with different levels of 'dirty' data is introduced, which provides a more accurate error-tolerance for 'dirty' data and consequently improves linkage quality. An analysis and an empirical study are conducted on large real-world datasets to show the benefit of the proposed approach.

Keywords Record linkage · Privacy · Bloom filter · Multi-LUs · Blocking

1 Introduction

Large amounts of data from several domains like businesses, government agencies and research projects have been generated, collected, and stored. Integrating such data from different sources to identify and match records that relate to the same real-world entities is known as record linkage, entity matching, or entity resolution [1], which is an important data pre-processing step in many data mining applications. Since unique entity identifiers are not always available in all the databases to be linked, the linkage can only be achieved by comparing available identifying attributes known as quasi-identifiers (QIDs), such as names, addresses and dates of birth. Values in QIDs are in general sufficiently well correlated with entities to allow accurate linkage. However, using such personal information often leads to privacy and confidentiality concerns [2, 3]. For instance, a patient's medical information may be stored in a fragmented form across multiple health care providers. A more complete view of a patient's medical information, afforded by record linkage, allows better effectiveness of treatment and services. However, medical information of patients are highly sensitive and originated from multiple health care providers. To overcome such concerns, techniques are required to integrate records from different data sources while the privacy of represented entities is maintained.

✉ Derong Shen
shenderong@ise.neu.edu.cn

Shumin Han
hanshumin_summer@yeah.net

Tiezheng Nie
nietiezheng@ise.neu.edu.cn

Yue Kou
kouyue@ise.neu.edu.cn

Ge Yu
yuge@ise.neu.edu.cn

[1] School of Computer Science and Engineering, Northeastern University, Hunnan, Shenyang 110169, Liaoning, China

Privacy preserving record linkage, PPRL [4, 5] is the process of identifying records from two or more data sources that refer to the same individuals, without revealing other information besides the matched records. Various approaches [6–8] have been proposed to achieve this goal, with the majority of them only considering linking two sources. However, linking data from several sources is commonly required. For the small number of existing multi-party PPRL (MP-PPRL) techniques, the main drawbacks are that either they only support exact matching [9–11] or they are applicable to QIDs of categorical data [12] only. However, approximate linkage using QIDs of string data, such as names and addresses, is required in many real-world applications. Therefore, in view of the practical significance of PPRL, proposing a MP-PPRL solution that supports approximate matching for strings is necessary. Recently, two PPRL approaches that satisfy the requirements above are proposed in [13] and [14]. They are both based on two efficient privacy techniques bloom filter, BF [15] and secure summation [16]. However, there are still some limitations on approximate MP-PPRL approaches: (1) in privacy, BFs may be vulnerable to frequency-based attacks [17, 18], which would induce privacy concerns; (2) in scalability, with the increasing of the size of databases and the number of parties, the complexity grows exponentially; (3) in linkage quality, when input data contains typographical errors or variations ('dirty' data), the linkage quality is poor.

To overcome the mentioned problems above, in this paper, we propose an enhanced approximate MP-PPRL approach, which uses a distributed multiple linkage units (Multi-LUs) protocol, a blocking technique based on sorted nearest neighborhood (SNN) approach for clustering BFs, called *BF-SNN*, and personalized threshold to offer (1) improved privacy against new frequency-based attacks [17, 18] proposed by Anushka Vidanage, (2) better scalability with multiple parties, and (3) higher linkage quality for 'dirty' data. To the best of our knowledge, no such approximate MP-PPRL technique has been developed in the literature so far. In summary, our major contributions are as follows:

- We propose a novel distributed Multi-LUs protocol to enhance privacy. We first partition each BF into several non-overlapping segments based on a bit positions partition method, then the segments are assigned to corresponding LUs to resist new frequency-based attacks. Moreover, we theoretically prove the security of our distributed Multi-LUs protocol.
- We develop a *BF-SNN* blocking method to reduce complexity. Both twice sorting algorithm and sliding window algorithm are designed for supporting better scalability on multiple databases.
- We introduce a personalized threshold varying with different levels of 'dirty' data, which provides a more accurate error-tolerance for 'dirty' data and consequently improves linkage quality.
- We analyze our approach and conduct an empirical study on large real-world datasets. Empirical results manifest that our approach outperforms previous techniques in scalability, linkage quality and privacy.

The rest of paper is organized as follows. We shortly discuss the related work in Sect. 2. We introduce the definitions and background of the study in Sect. 3. We describe our approach in Sect. 4 and analyze the approach in Sect. 5. Then in Sect. 6 we validate this analysis through an empirical study. Finally, we conclude this paper in Sect. 7.

2 Related work

Over recent years, several techniques have been developed to address the PPRL research problem, but few among these have considered PPRL on multiple databases. The existing MP-PPRL approaches can be classified into two categories, exact MP-PPRL which only matches records when the matching attributes are exactly identical, and approximate MP-PPRL which matches several records if they are very similar. The exact MP-PPRL work includes various approaches, an exact MP-PPRL approach was introduced in [9] to perform secure equi-join of masked records from multiple k-anonymous databases by using a LU. And an exact approach based on secure multi-party computation (SMC) using an oblivious transfer protocol was proposed in [10] for PPRL on multiple databases. Another efficient MP-PPRL approach of categorical data was proposed [12] using a Count-Min sketch data structure. Lai et al. proposed a MP-PPRL [11] for matching of masked records using BF, the approach also only performs exact matching.

Different from the study of exact MP-PPRL, the approximate MP-PPRL approaches are few. In 2014, an approximate MP-PPRL approach [13] based on two efficient privacy techniques BF and secure summation was proposed by Dinusha Vatsalan. In this approach, records are first converted into BFs. Each party then partitions its BFs into segments according to the number of parties p, and sends these segments to the corresponding other parties. The segments received by a party are calculated the number of common 1-bits and the total number of 1-bits. At last, they use the secure summation protocol to calculate the similarity of each set of BFs and classify the compared sets of records into matches and non-matches based on the similarity threshold s_t. Although the cost of this approach is

low since the computation is completely distributed among the parties and the processing of BF is very fast, the privacy of this approach is weak. This approach exposes partial information of BFs in each party to the other participants. Even with only partial information of BFs, it is still possible for the participants to employ a frequency-based attack against those BFs. And when data contains typographical errors or variations, the linkage quality of this approach rapidly descends, which indicates its poor error-tolerance. The other approximate MP-PPRL approach was proposed in [14] using counting BF (CBF), a variation of BF. Comparing with the approach in [13], this approach provides increased privacy without compromising linkage quality, however, at the sacrifice of scalability. Therefore, in this paper we propose a novel distributed Multi-LUs protocol to enhance privacy, develop a *BF-SNN* blocking method to reduce complexity and introduce a personalized threshold to improve linkage quality.

3 Preliminaries

In this section, we define the problem and introduce the related technologies investigated in the paper.

3.1 Problem formulation

Definition 1 (MP-PPRL). Assume P_1, P_2,...,P_p are p parties owning the datasets $D_1, D_2, \ldots, D_p$ respectively. They wish to identify the matched records among $D_1, D_2, \ldots, D_p$ according to a matching function in a privacy preserving manner, such that at the end of the process P_1, P_2,...,P_p will know only a set of matched records respectively and no information will be revealed about the non-matched records.

3.2 Bloom filter

BF is an efficient and accurate masking technique in a variety of PPRL approaches [11, 13, 14]. A BF is a bit vector of length l where initially all bit positions are set to 0. At first, a set of QIDs values are selected as matching attribute values (MAVs), such as names, dates of birth, and addresses. These selected MAVs are then converted into a set $S = \{s_1, s_2, \ldots, s_n\}$ of sub-strings of length q (known as q-grams). Each $s_x \in S$ is encoded into a given BF by using k independent hash functions $h_1, h_2, \ldots, h_k$ and all bits having index positions $h_y(s_x)$ for $1 \leq y \leq k$ in the BF are set to 1. As shown in Fig. 1 (left), the BF encodes two QIDs values 'sarah' and 'sara' into $l = 14$ BFs using $k = 2$ hash functions.

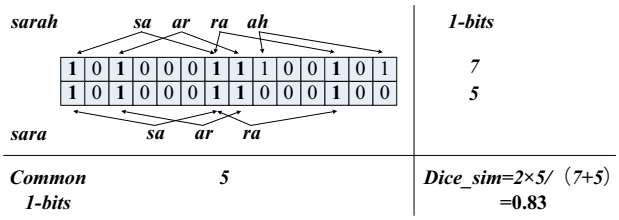

Fig. 1 Dice similarity calculation of two QID values ('sarah' and 'sara') masked using BFs

3.3 Dice coefficient

Any set-based similarity function can be used to calculate the similarity of pairs or sets of BFs. The Dice coefficient has been used for matching of BFs, since it is insensitive to many matching zeros in long BFs. We calculate the Dice coefficient similarity [13] of p BFs $b_1, b_2, \ldots, b_p$ as:

$$Dice\_sim(b_1, b_2, \ldots, b_p) = \frac{p \times c}{\sum_{i=1}^{p} x_i}, \quad (1)$$

where c 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 b_i (1-bits), $1 \leq i \leq p$. Figure 1 (right) illustrates the Dice coefficient similarity calculation of two QID values 'sarah' and 'sara' masked into BFs.

4 Approximate multi-party PPRL approach

In this section, we present our approximate MP-PPRL approach improving on privacy, scalability, and linkage quality. Firstly, to ensure privacy, each party encodes its records into BFs. However, BFs may be vulnerable to frequency-based attacks. So to enhance privacy, we propose a distributed Multi-LUs protocol to reduce the possibility of information leakage. Then, in that the computation complexity increases significantly with multiple parties, we propose a *BF-SNN* blocking method with two efficient algorithms twice sorting algorithm and sliding window algorithm to reduce the generation of candidate records on multiple databases. Finally, for the reason that the data in real-world often contains typographical errors or variations, a personalized threshold varying with different levels of 'dirty' data is introduced to improve linkage quality.

In addition, we study our approach in detail from Sects. 4.1 to 4.3 and illustrate our approach with an example as shown in Fig. 2. The symbols used in our approach are summarized in Table 1.

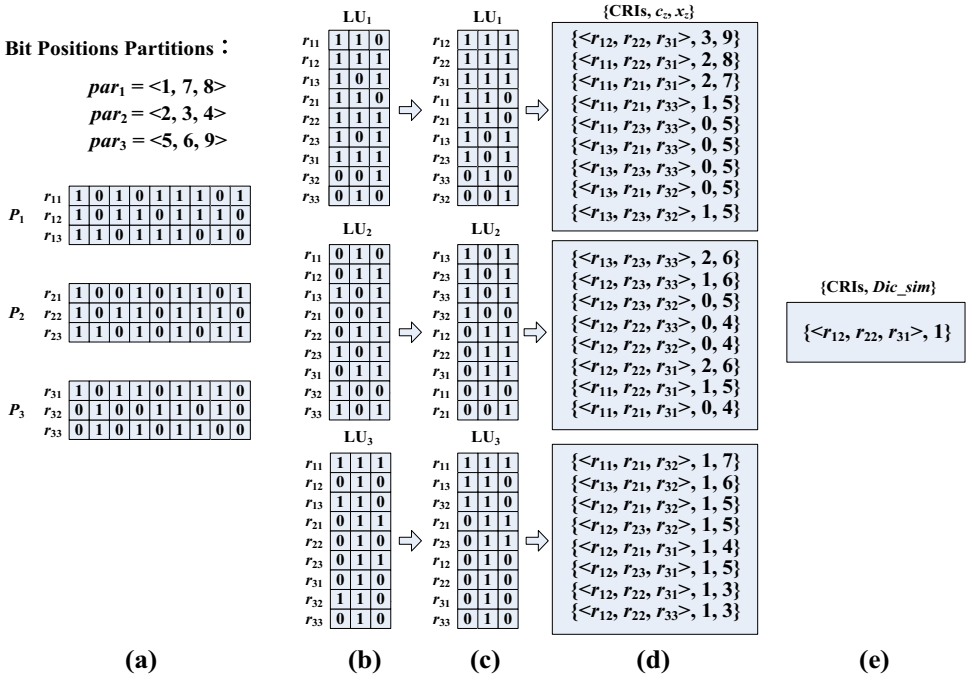

Fig. 2 Each party individually encodes its records into BFs and partitions each BF into three segments based on the bit positions partitions $par_1 = \{1, 7, 8\}, par_2 = \{2, 3, 4\}, par_3 = \{5, 6, 9\}$ as **a** shows. Then each part sends the segments to the corresponding LUs as **b** shows. Each LU independently sorts the segments they received by twice sorting algorithm as **c** shows. And then each LU slides the window to generate candidate records IDs (CRIs). For each CRIs, we calculate c_z and x_z as described in Sect. 4.2 to generate a triple $\{\text{CRIs}, c_z, x_z\}$ as **d** shows. At last, we integrate each CRIs from all of LUs and calculate *Dic_sim* as described in Eq. (1), by comparing the results of similarity with personalized threshold p_t, we can decide the CRIs $\{r_{12}, r_{22}, r_{31}\}$ is a match as **e** shows. In this example, $p = 3, N = 3, s = 3, w = 1$, and $p_t = 0.9$

Table 1 Table of frequent symbols

Symbol	Description
D_i	Datasets for party i, $1 \le i \le p$
N	Number of records in the dataset
p	Number of participants
k	Number of hash functions
l	The length of BF
s	Number of segments/LUs
B_i	The BFs representing dataset D_i
z	The zth segments/LUs, $1 \le z \le s$
g	The gth bit position, $1 \le g \le l$
w	The size of window
s_t	Similarity threshold
p_t	Personalized threshold

4.1 Distributed Multi-LUs protocol

BFs are vulnerable to frequency-based cryptography attacks, as mentioned in introduction. Some work has been done to resist the attacks, but recently Anushka Vidanage et al. proposed a new attack method [17, 18]. More specifically, the attack method applies frequent pattern mining to identify sets of *frequently co-occurring bit positions* (*fcobp*) that correspond to encoded frequent *q*-grams. Furthermore, they could re-identify plain-text values based on the identified *q*-grams.

In Fig. 3, an example of the new attack is shown. Firstly, it identifies that bit positions p_5 and p_{13} have co-occurring 1-bits in the same three BFs (b_1, b_3 and b_4) and therefore must encode "ma" which is the only q-gram that occurs in three plain-text values. Next, it sends that positions p_1 and p_{10} must encode 'jo' as they have co-occurring 1-bits in the same two BFs (b_2 and b_5) and 'jo' is the only q-gram that occurs in two plain-text values. Based on the identified q-grams and their bit positions, we learn that BFs b_2 and b_5 can only encode "john" and "joan", while b_1, b_3 and b_4 can encode "maude", "mary" or "max".

Therefore, to avoid the leakage of plain-text values, we propose a distributed multi-LUs protocol with aim to partition the *fcobp*s into different LUs to resist attacks. The distributed Multi-LUs protocol consists of two main phases:

4.1.1 Phase 1: generate partitions of bit positions

(a) Identify a set of fcobp s in each BF dataset. Before this phase, the sets of records need to be encoded into BFs.

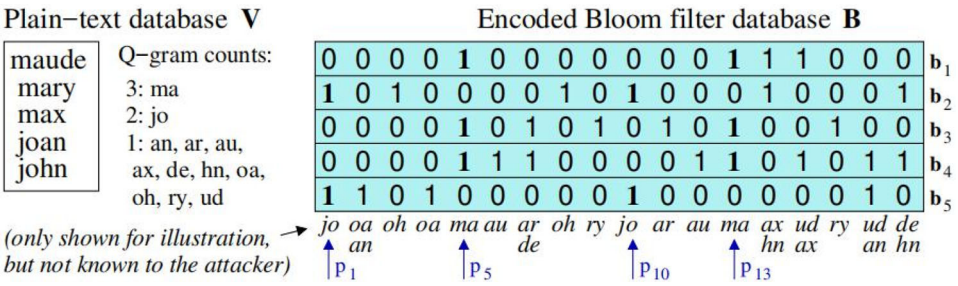

Fig. 3 The example of the new attack

After that, each party independently applies frequent pattern mining on its own BF dataset to identify a set of *fcobp*s. u is a support ratio of the require number of co-occurring 1-bits against the number of the records in the dataset. v is the minimum number of bit positions satisfying the ratio u. The bit positions that satisfy u and v are regarded as a *fcobp*. As shown in Fig. 4a, $u = 2/3$, $v = 2$, each party independently identifies a set of *fcobp*s.

(b) Construct a graph based on the sets of fcobps. Construct an undirected $G = (V, E), V = \{tv_1, v_2, \ldots, v_l\}$, v_g $(1 \leq g \leq l)$ represents the gth bit position; $E_{g,h}$ $(E_{g,h} \in E)$ represents that v_g and v_h are a *fcobp*; The number on $E_{g,h}$ represents the number that both v_g and v_h appear in the sets of *fcobp*s. As shown in Fig. 4b, an undirected graph is constructed based on the sets of *fcobp*s from three parties.

(c) Partition the bit positions. After constructing an undirected graph, to resist attacks, we only need to make sure that the adjacent vertices are partitioned into different LUs. Therefore, our problem can be changed into the well known vertex coloring problem (VCP) [19] that is defined as follows:

Definition 2 (*Vertex coloring problem*). Given an undirected graph G = (*V*, *E*) with vertex set V and edge set E, the VCP requires to assign a color to each vertex in such a way that colors on adjacent vertices are different and the number of colors used is minimized.

There have been many approaches to solve VCP [19]. Employing any approach, the adjacent vertices with different colors would be partitioned into different partitions, which realizes the partition of *fcobp*s. As shown in Fig. 4b, the partitions of bit positions are $par_1 = \{1, 7, 8\}$, $par_2 = \{2, 3, 4, 5, 9\}$, $par_3 = \{6\}$.

However, due to the load balance of our distributed protocol, we should guarantee that the number of vertices in each partition is equal. Specifically, to obtain the equal partitions, the bit positions (vertices) partition method is performed as follows. Firstly, we classify the vertices into two categories, the one with edges and the one without edges. The isolated vertices without edges can be put into any partitions to adjust the size of partition. Next, we only perform the approach in VCP on the vertices with edges to generate the initial partitions. We assume the number of final partitions is s, the cardinalities of vertex set is l. To guarantee the size of each partition is equal, we compare the size of each initial partition with l/s, if they are equal, the bit positions in this partition would be regarded as a final partition. If the size of a initial partition is smaller than l/s, we would insert the isolated vertices into this partition until the size of it is equal to l/s. If the size of a initial partition is larger than l/s, we would retain l/s vertices as a final partition. And the other vertices are regarded as a new partition.

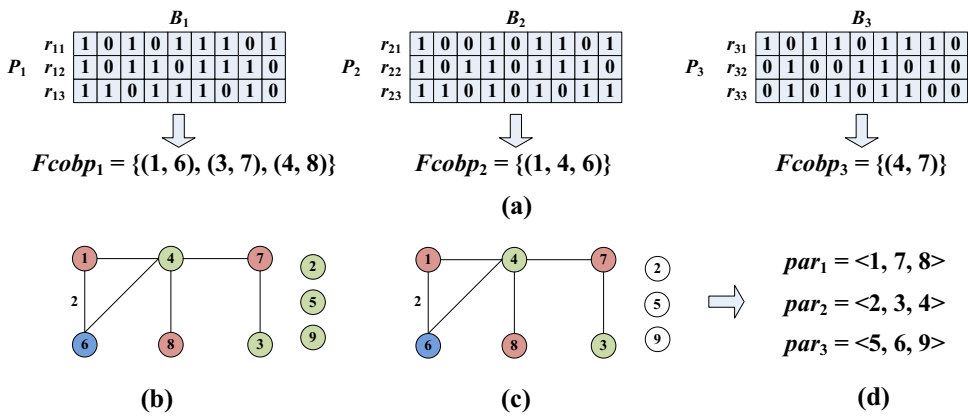

Fig. 4 The process of phase 1 in distributed Multi-LUs protocol

For example, in Fig. 4c, d, the vertices are first classified into isolated vertices {2, 5, 9} and the vertices with edges {1, 3, 4, 6, 7, 8}. Then, we perform the approach in VCP on the vertices {1, 3, 4, 6, 7, 8}. The initial partitions are $par_1 = \{1, 7, 8\}$, $par_2 = \{3, 4\}$, $par_3 = \{6\}$. In the example, $l = 9$, $s = 3$, $| par_1 | = 3$, which is equal to l/s, so par_1 is regarded as a final partition; $| par_2 | = 2$, which is smaller than l/s, so we insert vertex 2 into par_2; $| par_3 | = 1$, which is smaller than l/s, so we insert vertex 5 and vertex 9 into par_3. The final partitions are $par_1 = \{1, 7, 8\}$, $par_2 = \{2, 3, 4\}$, $par_3 = \{5, 6, 9\}$.

4.1.2 Phase 2: generate segments for each LU

In this phase, each party partitions its BFs into s segments according to the final partitions. After that, the segments are sent to the corresponding LU_z $(1 \leq z \leq s)$ by each party independently. Each LU receiving the segments from all the parties would perform blocking and matching distributed in the next texts. The security of our distributed Multi-LUs protocol is discussed as follows:

Proposition 1 *For each LU, knowing the segments from p BF datasets, it can not identify the fcobps based on the frequent pattern mining.*

Proof We assume a *fcobp* has been identified in a LU based on the frequent pattern mining.

For the existence of a *fcobp*, it should exist the bit positions that satisfy the u and v in a LU. Because the segments in a LU are from p BF datasets, then at least in one BF dataset, the bit positions in this *fcobp* should satisfy the u and v. Therefore, the bit positions in this *fcobp* have been regarded as a *fcobp* in at least one BF dataset and would have been partitioned into different LUs. The conclusion is opposite to the proposition that the bit positions are in the same LU.

As a consequence, the assumption is false, we prove the Proposition 1. □

Based on the proof above, we conclude that our approach can resist the new frequency-based attacks.

4.2 *BF-SNN* blocking method

In this section, we present the *BF-SNN* blocking method, which aims at clustering the encoded segments and reducing the number of comparisons in each LU. Originally SNN methods consist of the following steps. Firstly, a summary of each record is created. Next, all records are sorted upon the values of these summaries. Finally, matching is performed by sliding a size window over the resulting list of records. Different from the previous SNN blocking methods that are only applicable to two data sources [20, 21], our *BF-SNN* blocking method is for multiple data sources. Therefore, a new sorting algorithm and a new sliding window algorithm are designed in our *BF-SNN* blocking method.

4.2.1 Twice sorting algorithm

Firstly, we merge and sort the segments to make the similar segments from multiple parties to be close. To achieve this goal, we design a twice sorting algorithm to sort the segments. In the first sorting, we sort the segments according to the number of 1-bits. As to the segments with the same number of 1-bits, we perform the twice sorting to sort them in descending order (Algorithm 1, lines 1–6). As Fig. 2c shows, the segments in each LU have been sorted by twice sorting algorithm.

4.2.2 Sliding window algorithm

After twice sorting algorithm, we use a sliding window of size w on the sorted list to identify the CRIs that fall in the same window (Algorithm 1, lines 7–17). The value for w represents the number of segments that must be included in the window from each party. In other words, if $w = 1$, we must guarantee in the window one segment is included from each participant. The windows in LU_1 are shown in Fig. 5. Any p records from different parties in the same window are regarded as a CRIs. For each CRIs in LU_z, we calculate the number of common 1-bits bit positions in all BFs, c_z, and the number of bit positions set to 1 in all BFs, x_z. Finally, as Fig. 2d presents, the triples {CRIs, c_z, x_z} are generated (Algorithm 1, lines 18–21).

4.3 Matching results generation and personalized threshold

In the previous steps, the CRIs have been generated in each LU. Due to the distributed process, integrating each CRIs from all LUs is necessary to classify the CRIs into matches or non-matches. In our approach, the process of generating matching results contains two steps. In the first step, only the CRIs that exist in all LUs are retained. In the second step, we calculate the Dice coefficient similarity of each remaining CRIs according to the c_z and x_z. Consequently, our decision rules (DR) can be described as follows:

$$DR = \begin{cases} Dice\_sim(b_1, b_2, \ldots, b_p) \geq s_t, & match, \\ otherwise, & non\text{-}match. \end{cases} \quad (2)$$

Only when the similarity is no smaller than the similarity threshold s_t, we regard the CRIs as a match.

Algorithm 1 *BF-SNN* Blocking Method

Require: LU_z: Set of the zth segments from all B_{ij}, $1 \le i \le p$, $1 \le j \le N$, $1 \le z \le s$; w: Size of the window;
Ensure: Triples {CRIs, c_z, x_z};
1: **for** each LU_z **do**
2: **for** each $B_{ij,z}$ **do**
3: countnumber($B_{ij,z}$);
4: sort the $B_{ij,z}$ according to the number of 1;
5: **end for**
6: **end for**
7: **for** each $B_{ij,z}$ with the same number of 1 **do**
8: sort the $B_{ij,z}$ according to the descending order;
9: **end for**
10: $m = 0$;
11: **while** $m < len(\mathrm{LU}_z)$ **do**
12: **for** each D_i **do**
13: $n = 0$;
14: **while** (any $len(\mathrm{D}_i) \le w$ and $m + n < len(\mathrm{LU}_z)$ and $\mathrm{LU}_z[m+n] \neq 0$) = true **do**
15: **if** $\mathrm{LU}_z[m+n] \in P_i$ **then**
16: D_i += $\mathrm{LU}_z[m+n]$;
17: *cluster_dataset*[c] = *cluster_dataset*[c] + D_i;
18: n += 1;
19: **end if**
20: **end while**
21: c += 1;
22: m += n;
23: **end for**
24: **end while**
25: **for** each *cluster_dataset* **do**
26: **for** any p records (r_1, r_2,..., r_p) in it **do**
27: generate {CRIs, c_z, x_z};
28: **end for**
29: **end for**
30: return {CRIs, c_z, x_z}

The setting of similarity threshold is so important that it determines the error-tolerance of the approach and consequently the linkage quality. When the threshold is set to 1, the approach would only support exact matching, which classifies record sets as matches if their masked QIDs are exactly the same. However, the data in real-world often contains typographical errors or variations. To improve error-tolerance, previous approaches usually discretionarily set a similarity threshold smaller than 1, which is unreasonable and would induce low linkage quality. In our approach, we set the personalized threshold according to the levels of errors to provide a more accurate error-tolerance for 'dirty' data.

We assume a record exists e errors, then the generated BF corresponding this record would have no more than ekq positions be influenced. As the similarity calculation shown in Eq. (1), the errors would produce a greater impact with the more participants p. Therefore, the influence of errors on the similarity is calculated as follows:

$$p_t = s_b - \frac{ekqp}{\sum_{z=1}^{s} x_z}. \tag{3}$$

s_b is a basic similarity threshold. The personalized threshold p_t is decided by the p and e. When p is maintained, the bigger the e the smaller the threshold of p_t, and consequently the higher error-tolerance; vice versa.

5 Analysis of the approach

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

Privacy As with most of the existing PPRL approaches [22–24], we assume that all parties follow the semi-honest adversary model [25], where the parties follow the protocol honestly, but may try to infer private information based on messages they receive during the process without collusion. Next we summarize the information that our approach discloses to the participants and LUs.

$\boldsymbol{P}_i$ $(1 \le i \le p)$: Each party does not receive any messages regarding other participants. Therefore, without collusion each part cannot infer any information.

$\mathbf{LU}_z$ $(1 \le z \le s)$: Each LU only learns l/s bits of each BF. And the segments in each LU can resist new frequency-based attacks, which has been proved in Sect. 4.1. Therefore, without collusion each LU also cannot infer any information.

Complexity We assume p parties in the approach, each having a dataset of N records. In step 1 of our approach, the agreement of parameters has a constant communication complexity. And the creation of BFs using k hash functions for N records is $O(kN)$. In step 2 of our approach, each party sends its BF segments (each of length l/s) to the corresponding LUs. If we assume direct communication, s^2 messages are required in this step, each of these of size $N \times l/s$ ($O(Nls)$ total communication). In the step of *BF-*

Fig. 5 The windows in LU_1

	$\mathbf{LU_1}$		
r_{12}	1	1	1
r_{22}	1	1	1
r_{31}	1	1	1
r_{11}	1	1	0
r_{21}	1	1	0
r_{13}	1	0	1
r_{23}	1	0	1
r_{33}	0	1	0
r_{32}	0	0	1

Windows / Records	w_1	w_2	w_3	w_4	w_5
In P_1	r_{12}	r_{11}	r_{11}	r_{11}, r_{13}	r_{13}
In P_2	r_{22}	r_{22}	r_{21}	r_{21}, r_{23}	r_{21}, r_{23}
In P_3	r_{31}	r_{31}	r_{31}	r_{33}	r_{32}

SNN blocking method, we assume the size of sliding window is w and then the maximum number of windows is N/wp, the number of candidate record pairs for each LU generated by our approach is $w^{p-1}N/p$. The combinatorial complexity is low because of the small value of w. The similarity calculation phase consists of the integrating each CRIs from all LUs, which requires a communication over all LUs of $O(s \cdot w^{p-1}N/p)$.

Linkage quality Our approach supports approximate matching of QIDs values, in that data errors and variations are taken into account depending on the personalized threshold p_t. The personalized threshold varies with different levels of 'dirty' data to improve linkage quality. The quality of BF masking depends on the BF parameters. For a given BF length l, the number of elements q, and the optimal number of hash functions k, the minimum false positive rate f is calculated as $f = (1/2^{ln(2)})^{l/q}$.

6 Experiments

In this section, we evaluate the scalability, linkage quality and privacy of different approaches on a variety of real-world datasets. We implemented all approaches in Python 3.6.5, and ran all experiments on a server with a 64-bit, 8.0G of RAM Intel Core (3.30 GHz) CPU.

6.1 Datasets

We used the large real-world voter registration dataset from North Carolina (NC) as available from ftp://alt.ncsbe.gov/data/. We downloaded the dataset ncvoter_Statewide that contains over 8 million records. To evaluate our approach with different dataset sizes, different number of parties and different data quality, we used a recently proposed data corruptor [26] to create a variety of datasets with different characteristics. We extracted four attributes commonly used for record linkage: first name, last name, city, and zipcode. To generate datasets of different sizes, we extracted sub-sets of 5000, 10000, 50000, 100000, 500000, and 1000000 records from the NC dataset for each party where the number of matching records was set to 50% (i.e. half of all selected records occur in the datasets of all parties). The number of parties was set to $p = [3, 5, 7, 10]$. To investigate how our approach deals with 'dirty' data, 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).

6.2 Baselines and settings

6.2.1 Baselines

The experiments were twofold. **In the first part**, we evaluated the scalability of our *BF-SNN* blocking method. For comparative evaluation purposes we used two state-of-the-art multi-party private blocking techniques, the hierarchical canopy clustering-based (HCC) blocking method [27] and the distributed clustering and hashing (DCH) blocking method [28]. **In the second part**, we compared the complexity, linkage quality, and privacy of our approach with the previous two approximate PPRL approach [13, 14], which we call BF-based approach [13] and CBF-based approach[14].

6.2.2 Settings

Following earlier BF work in PPRL [13, 14], we set the BF parameters as the length of BF $l = 1000$, the number of hash functions $k = 20$, the length of grams $q = 2$. In the distributed Multi-LUs protocol, we set the number of LUs as $s = [5, 10, 20, 50], u = 2/3, tv = 2$. In the *BF-SNN* blocking method, the size of window was set to $w = [1, 2, 3, 4, 5]$. We set the parameters of the HCC and DCH multi-party private blocking approaches according to the settings provided by the authors.

6.3 Evaluation metrics

We evaluate the scalability of our approach by runtime. The linkage quality of our approach is measured by precision and recall. Precision is calculated as the ratio of the number of true matched record pairs found against the total number of candidate record pairs compared across datasets. And recall is calculated as the ratio of the number of true matched record pairs against the total number of true matched record pairs across all datasets.

The blocking quality is measured by reduction ratio (RR) and pair completeness (PC). RR is the fraction of record pairs that are removed by a blocking technique and PC is the fraction of true matching record sets that are included in the candidate record sets generated by a blocking technique.

In line with other work in PPRL [13, 14], we evaluate privacy using disclosure risk (DR) that measures based on the probability of suspicion, i.e. the likelihood a masked dataset record can be matched with one or several masked records a^M in a publicly available global dataset D^M. We show mean DR values is calculated as follows:

$$DR = \frac{1}{n} \sum_{a^M \in D^M} P_s(a^M). \tag{4}$$

To evaluate the ability of resisting the frequency-based attacks, we measure the quality of the identified frequent q-grams, as the precision and recall of how many bit positions are correctly identified for a q-gram.

6.4 Performance evaluation

6.4.1 Scalability

In our first set of experiments, we evaluate the scalability of three multi-party private blocking methods *BF-SNN*, DCH, and HCC with different dataset sizes and the number of parties. According to the experimental results illustrated in Figs. 6 and 7, our *BF-SNN* blocking method requires less runtime than previous HCC and DCH blocking methods and is scalable to large datasets. Figures 8 and 9 respectively show the blocking quality of *BF-SNN* blocking method and average time required for blocking with different window sizes w. As expected, PC and runtime for blocking increase with w while RR decreases. This is because there are more candidate records generated with increasing w. When $w = 2$, there is a drastic improvement in PC with a smaller increase in runtime and smaller decrease in RR, thus $w = 2$ is chosen as the default parameter. Our *BF-SNN* blocking method performs well by achieving high values for both RR and PC, which indicates the effectiveness of proposed twice sorting algorithm and sliding window algorithm in *BF-SNN* blocking method.

6.4.2 Complexity

Figure 10 shows the complexity of three approximate MP-PPRL approaches in terms of the runtime required over the whole process with different s for five parties. In our approach, the runtime decreases at first and then increases. The reason is that with increasing s, the segments received by each LU with the length of l/s get shorter, accordingly the time of matching would decrease. But we need to integrate the distributed results from more LUs. When $s = 20$, these two achieve a balance. For the previous two approaches, the CBF-based approach requires more runtime, because it needs more communications than the BF-based approach. The results above indicate that our approach achieves better efficiency by adjusting s.

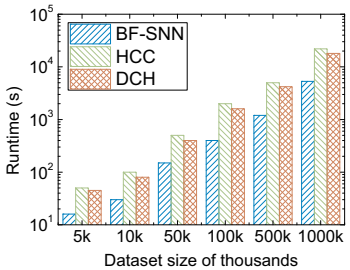

Fig. 6 Runtime with different N

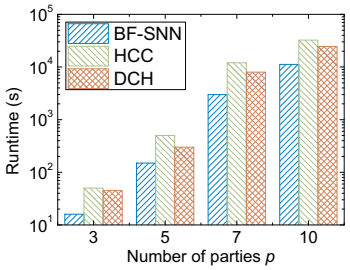

Fig. 7 Runtime with different p

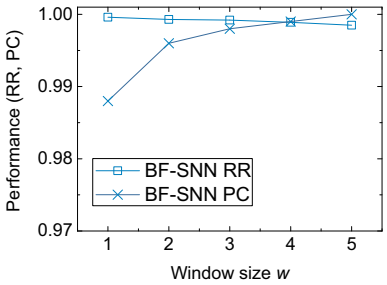

Fig. 8 The quality of *BF-SNN* blocking method with different w

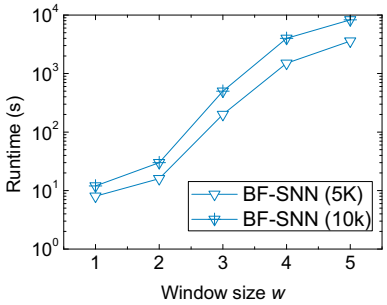

Fig. 9 The average time required for *BF-SNN* blocking method with different w

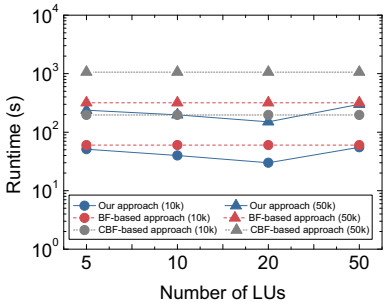

Fig. 10 Runtime with varying s

6.4.3 Linkage quality

The linkage quality of three approximate MP-PPRL approaches with different dataset sizes and numbers of parties is presented in Figs. 11, 12, 13 and 14 on modified and non-modified datasets. As can be seen, in Fig. 11, precision and recall are both high on non-modified datasets in three approaches. However, on mod-1 datasets the recall of BF-based approach and CBF-based approach drop quite drastically with increasing p as shown in Fig. 12. The reason is that the invariable similarity threshold induces an increase in the number of missed true matches with increasing p. Owing to the setting of personalized threshold, the recall of our approach is still high. Figures 13 and 14 show that the linkage quality of our approach keeps high even on different corruption levels of datasets or the size of dataset becoming larger.

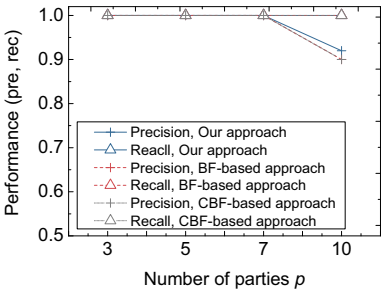

Fig. 11 The linkage quality on non-mod dataset with different p

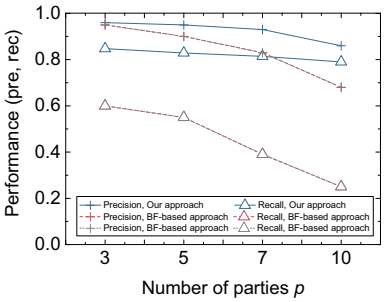

Fig. 12 The linkage quality on mod-1 dataset with different p

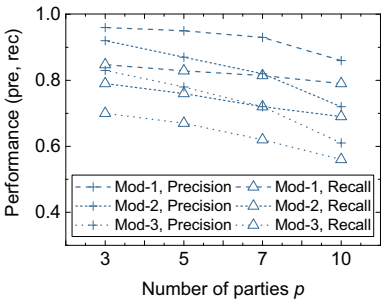

Fig. 13 The linkage quality with different levels of errors

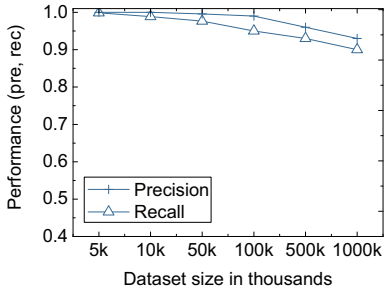

Fig. 14 The linkage quality with different N

6.4.4 Privacy

The privacy of three approaches with different p for 20 LUs, as measured by DR of an exact matching attack using the full NC dataset as the global dataset, is shown in Fig. 15. The DR of our approach is lower than the previous two approaches, for the reason that the BF segments in our approach with much shorter length are matched to more global records. In addition, we apply new frequency-based attacks to the three approaches to measure the ability to resist new attacks. Figure 16 shows the quality of the identified frequent q-grams in three approaches. In the BF-based approach, a small part of q-grams is re-identified with the low quality of identified frequent q-grams, this is because the attacks are based on the BF segments. The privacy is increased in CBF-based approach which is identified less frequent q-grams than the BF-based approach. In our approach, the quality of the identified frequent q-grams is close to zero, which verifies the effectiveness of our distributed Multi-LUs protocol.

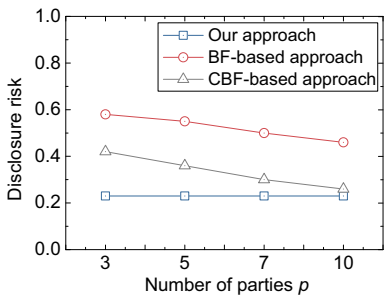

Fig. 15 Disclosure risk with different p

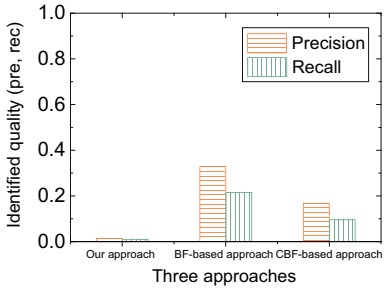

Fig. 16 The quality of the identified frequent q-grams

Above all, we conclude that our approach outperforms previous techniques in scalability, linkage quality and privacy.

7 Conclusion

In this paper, the problem of approximate matching entities from more than two sources in privacy has been studied. It is a challenging task due to the massive amount of data, multiple data sources, and 'dirty' data. Therefore, an enhanced approximate MP-PPRL approach has been proposed to improve privacy, scalability, and linkage quality. To enhance privacy, a novel distributed Multi-LUs protocol is proposed to resist the frequency-based attacks. To reduce complexity, a *BF-SNN* blocking method is developed. To improve linkage quality, a personalized threshold varying with different levels of 'dirty' data is introduced. Experiments conducted on real datasets show the approach in this paper is better in scalability by comparing with previous MP-PPRL blocking and matching methods while achieving superior results in terms of linkage quality and privacy. In the future, in order to apply the technique of PPRL to big data, improving the efficiency, linkage quality and privacy of PPRL is still an urgent problem to be solved.

Acknowledgements This work is supported by the National Basic Research 973 Program of China under Grant No. 2012CB316201 and the National Natural Science Foundation of China under Grant Nos. (61472070, 61672142, U1435216, 61602103).

Funding The authors have not disclosed any funding.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest The authors have not disclosed any competing interests.

References

1. Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer, Berlin (2012)
2. Vatsalan, D., Karapiperis, D., Verykios, V S.: Privacy-preserving record linkage. In: Encyclopedia of Big Data Technologies. Springer, Cham (2019)
3. Xu, X., Xue, Y., Qi, L., et al.: An edge computing-enabled computation offloading method with privacy preservation for internet of connected vehicles. Future Gener. Comput. Syst. **96**(July), 89–100 (2019)
4. Qi, L., Zhang, X., Li, S., et al.: Spatial–temporal data-driven service recommendation with privacy-preservation. Inf. Sci. **515**, 91–102 (2019)
5. Vatsalan, D., Christen, P., Verykios, V.S.: A taxonomy of privacy-preserving record linkage techniques. Inf. Syst. **38**(6), 946–969 (2013)
6. Vatsalan, D., Sehili, Z., Christen, P., Rahm, E.: Privacy-preserving record linkage for big data: current approaches and research challenges. In: Handbook of Big Data Technologies, pp. 851–895. Springer, Cham (2017)
7. Nóbrega, T., Pires, C., Nascimento, D.C.: Blockchain-based privacy-preserving record linkage enhancing data privacy in an untrusted environment. Inf. Syst. **102**, 101826 (2021)
8. Rohde, F., Franke, M., Sehili, Z., et al.: Optimization of the Mainzelliste software for fast privacy-preserving record linkage. J. Transl. Med. **19**(1), 33 (2021)
9. Kantarcioglu, M., Wei, J., Malin, B.: A privacy-preserving framework for integrating person-specific databases. In: UNESCO Chair in Data Privacy International Conference on Privacy in Statistical Databases, pp. 298–314 (2008)
10. Christine, M.O., Yung, M., Gu, L.F., Rohan, B.: Privacy-preserving data linkage protocols. In: Proceedings of ACM Workshop on Privacy in the Electronic Society, pp. 94–102 (2004)
11. Lai, P.K.Y., Yiu, S.M., Chow, K.P., Chong, C.F., Hui, L.C.K.: An efficient bloom filter based solution for multi-party private matching. In: Proceedings of the 2006 International Conference on Security and Management, 2006, pp. 286–292 (2006)
12. Karapiperis, D., Vatsalan, D., Verykios, V.S., Christen, P.: Large-scale multi-party counting set intersection using a space efficient global synopsis. In: International Conference on Database Systems for Advanced Applications, pp. 329–345 (2015)
13. Vatsalan, D., Christen, P.: Scalable privacy-preserving record linkage for multiple databases. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, pp. 1795–1798 (2014)
14. Vatsalan, D., Christen, P., Rahm, E.: Scalable privacy-preserving linking of multiple databases using counting bloom filters. In: IEEE 16th International Conference on Data Mining Workshops, pp. 882–889 (2016)
15. Schnell, R., Bachteler, T., Reiher, J.: Privacy-preserving record linkage using bloom filters. BMC Med. Inform. Decis. Mak. **9**(1), 41 (2009)
16. Karr, A.F., Lin, X.D., Sanil, A.P., Reiter, J.P.: Analysis of integrated data without data integration. Chance **17**(3), 26–29 (2004)
17. Christen, P., Vidanage, A., Ranbaduge, T.: Pattern-mining based cryptanalysis of bloom filters for privacy-preserving record linkage. In: Pacific–Asia Conference on Knowledge Discovery and Data Mining, pp. 530–542 (2018)
18. Vidanage, A., Ranbaduge, T., Christen P., Schnell R.: Efficient pattern mining based cryptanalysis for privacy-preserving record linkage. In: IEEE 35th International Conference on Data Engineering, 2019, pp. 1698–1701 (2019)

19. Malaguti, E., Toth, P.: A survey on vertex coloring problems. Int. Trans. Oper. Res. **17**(1), 1–34 (2010)
20. Vatsalan, D., Christen, P.: Sorted nearest neighborhood clustering for efficient private blocking. In: Advances in Knowledge Discovery and Data Mining, pp. 341–352 (2013)
21. Vatsalan, D., Christen, P., Verykios, V.S.: Efficient two-party private blocking based on sorted nearest neighborhood clustering. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, pp. 1949–1958 (2013)
22. Kuzu, M., Kantarcioglu, M., Inan, A., Bertino, E., Durham, E., Malin, B.: Efficient privacy-aware record integration. In: Proceedings of the 16th ACM International Conference on Extending Database Technology, pp. 167–178 (2013)
23. Bonomi, L., Xiong, L., Chen, R., Fung, B.C.: Frequent grams based embedding for privacy preserving record linkage. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 1597–1601 (2012)
24. Inan, A., Kantarcioglu, M., Bertino, E., Scannapieco, M.: A hybrid approach to private record linkage. In: Proceedings of the 24th IEEE International Conference on Data Engineering, pp. 496–505 (2008)
25. Franke, M., Gladbach, M., Sehili, Z., Rohde, F., Rahm, E.: ScaDS research on scalable privacy-preserving record linkage. Datenbank-Spektrum **19**(1), 31–40 (2019)
26. Christen, P., Vatsalan, D.: Flexible and extensible generation and corruption of personal data. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, pp. 1165–1168 (2013)
27. Ranbaduge, T., Vatsalan, D., Christen, P.: Clustering-based scalable indexing for multi-party privacy preserving record linkage. In: Pacific–Asia Conference on Knowledge Discovery and Data Mining, pp. 549–561 (2015)
28. Ranbaduge, T., Vatsalan, D., Christen, P., Verykios, V.S.: Hashing-based distributed multi-party blocking for privacy-preserving record linkage. In: Pacific–Asia Conference on Knowledge Discovery and Data Mining, pp. 415–427 (2016)

Shumin Han is a PhD Candidate in the College of Computer Science and Engineering, Northeastern University, China. She received her BS Degree from the same university in 2014. Her research interest is privacy preserving record linkage.

Derong Shen is a Professor and PhD Supervisor in the College of Computer Science and Engineering, Northeastern University, China, from where she received her PhD in 2004. She received her BS and MS from Jilin University, China in 1987 and 1990, respectively. Her interests include distributed data management and data integration.

Tiezheng Nie is an Associate Professor in the College of Computer Science and Engineering, Northeastern University, China, from where he received his BS, MS and PhD in 2002, 2005 and 2009, respectively. His interests include data quality and data integration.

Yue Kou is an Associate Professor in the College of Computer Science and Engineering, Northeastern University, China, from where she also received her BS, MS and PhD in 2002, 2005, and 2009, respectively. Her interests include entity search and data mining.

Ge Yu is a Professor and PhD Supervisor in the College of Computer Science and Engineering, Northeastern University, China, from where he received his BS and MS in 1982 and 1985, respectively. He received his PhD from Kyushu University of Japan, Japan in 1996. He is a Senior Member of the CCF, and a Member of the ACM, IEEE. His interests include databases and big data management.