



# Post-processing Methods for High Quality Privacy-Preserving Record Linkage

Martin Franke<sup>(✉)</sup>, Ziad Sehili, Marcel Gladbach, and Erhard Rahm

Database Group, University of Leipzig, Leipzig, Germany  
{franke,sehili,gladbach,rahm}@informatik.uni-leipzig.de

**Abstract.** Privacy-preserving record linkage (PPRL) supports the integration of person-related data from different sources while protecting the privacy of individuals by encoding sensitive information needed for linkage. The use of encoded data makes it challenging to achieve high linkage quality in particular for dirty data containing errors or inconsistencies. Moreover, person-related data is often dense, e.g., due to frequent names or addresses, leading to high similarities for non-matches. Both effects are hard to deal with in common PPRL approaches that rely on a simple threshold-based classification to decide whether a record pair is considered to match. In particular, dirty or dense data likely lead to many multi-links where persons are wrongly linked to more than one other person. Therefore, we propose the use of post-processing methods for resolving multi-links and outline three possible approaches. In our evaluation using large synthetic and real datasets we compare these approaches with each other and show that applying post-processing is highly beneficial and can significantly increase linkage quality in terms of both precision and F-measure.

**Keywords:** Record linkage · Post-processing · Privacy  
Linkage quality

## 1 Introduction

Privacy-preserving record linkage (PPRL) is the task of identifying records across different data sources referring to the same real-world entity, without revealing sensitive or personal information [47]. In contrast to traditional record linkage (RL), PPRL has to protect sensitive data to ensure the privacy and confidentiality of the entities, usually representing persons [43]. PPRL techniques are required in many areas, for instance in medical and health care applications. A typical use case is the integration of patient-related data from different sources, i.e., hospitals, registries and insurance companies, to allow comprehensive analysis and research about certain diseases or treatments [13, 23, 25, 29]. Other use cases for PPRL techniques include epidemiological or demographical studies as well as marketing analysis [47].

To preserve the privacy of represented entities, PPRL techniques have to ensure that no personal identifiers or other sensitive information is revealed during or after the linkage process. For an adversary it should be impossible to identify a person or to infer any sensitive information, like a person’s state of health. Therefore, the data needed for analysis, e.g., medical data, is separated from the data required for linkage. Record linkage is conducted by comparing commonly available attributes, called quasi-identifiers (QIDs), like first and last name or date of birth. Since such QIDs also contain private information, these attributes are encoded (masked) for PPRL to preserve privacy. Consequently, such data encodings have to be highly secure while still allowing linkage, i.e., they still have to enable efficient similarity calculations between records. Real-world data often contains errors or inconsistencies [15, 36]. Hence, encoding techniques for PPRL have to support approximate matching to achieve high linkage accuracy.

A high linkage quality is essential for practical applicability of PPRL, especially in the medical domain. Ideally, a PPRL approach should find all matches, despite possible data quality problems in the source databases. On the other hand, false matches should be strictly avoided, as otherwise (medical) conclusions based on incorrect assumptions could be made.

Classification models are used to decide whether a record pair represents a match or a non-match. For traditional RL sophisticated classification models based on supervised machine learning approaches, e.g., support vector machines or decision trees, can be used to achieve highly accurate linkage results [4]. Moreover, linkage results could be manually reviewed to increase final quality or to adjust parameter configurations. In contrast, currently most PPRL approaches only apply threshold-based classification based on a single threshold, as training data is usually not available in a privacy-preserving context [43]. In general, it is also not feasible to manually inspect actual QID values of records because this would give up part of the privacy. Finally, recent encoding techniques often aggregate all attribute values into a single binary encoding making it hard to deploy attribute-wise or rule-based classification [43, 47]. All these effects likely reduce linkage quality of PPRL, indicating the demand for refined classification techniques [8].

In this paper, we study post-processing methods for improving linkage quality of PPRL in terms of precision. By using simple threshold-based classification approaches, only low linkage accuracy is likely achieved in PPRL scenarios dealing with *dirty* or *dense* data. Dirty data such as missing or erroneous attribute values can lead to a low similarity between matching records that are thus easily missed with a higher similarity threshold. Another problem case are dense data where many non-matching records can have a high similarity. For example, members of a family often share the same last name and address leading to a high similarity for different persons. Datasets focusing on a specific city or region also tend to have many persons with similar addresses. For such dense data there can be many non-matching record pairs with a similarity above a fixed threshold.

A key drawback of classification approaches based on a single threshold is that they often produce multi-links, i.e., one record is linked (matched) to many records of another source, and moreover, each record pair exceeds the similarity threshold. But, assuming deduplicated source databases, each record can at most match to one record of another source. Hence, the linkage result should exclusively contain one-to-one links as otherwise precision is deteriorated. Therefore, we analyze methods that can be executed after any (threshold-based) classification to clean multi-links, i.e., to transform the linkage result such that only one-to-one links occur in the final result. In particular, we make the following contributions:

- We study three post-processing strategies for the cleaning of multi-links, or selection of match candidates respectively, to increase the overall linkage quality of PPRL, especially when dealing with dense or dirty data.
- We evaluate the different post-processing approaches using large synthetic and real datasets showing different data characteristics and difficulty levels.
- In our evaluation, we consider both linkage quality in terms of recall, precision and F-measure, as well as efficiency in terms of runtime.

In Sect. 2, we outline the basic PPRL process and discuss related work in the field of PPRL. Then, we formalize the multi-link cleaning problem that we want to address with post-processing (Sect. 3) and describe approaches for solving it (Sect. 4). In Sect. 5, we evaluate selected approaches in terms of quality and efficiency. Finally, we conclude.

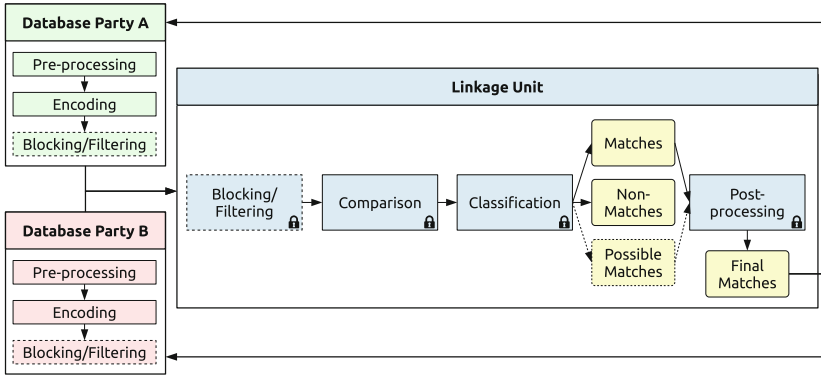
## 2 Background and Related Work

In this section we describe the overall PPRL process and discuss related work.

### 2.1 PPRL Process

A PPRL pipeline contains multiple steps which are shown in Fig. 1. Following previous work, we assume a three-party protocol, where a (trusted) third party, called linkage unit (LU), is required [43]. The LU conducts the actual linkage of encoded records from two or more database owners (DBOs). While we focus on only two DBOs with their respective databases  $\mathbf{D}_A$  and  $\mathbf{D}_B$  the PPRL process can be extended to multiple DBOs. In the following, each step of the PPRL process is described and relevant techniques are discussed. It is assumed that general information and parameters are exchange in advance between the DBOs.

**Data Pre-processing:** At first, the source databases to be linked need to be pre-processed by the DBOs. Pre-processing includes deduplication, data cleaning and standardization. In each individual database, duplicate records may occur due to inconsistent or repetitive recording processes. Therefore, the DBOs have to internally link and deduplicate their databases to ensure that a record from one data source can only be linked to at maximum one record from another data



**Fig. 1.** Outline of the basic PPRL process under a three-party protocol. Steps in dotted boxes are optional. Steps with a lock symbol process encoded data.

source. Furthermore, data cleaning and standardization is required since real-world data often contain erroneous, missing, incomplete, inconsistent or outdated data [15]. Data cleaning techniques aim at curating or weakening such errors, e.g., by filling in missing data or removing unwanted values [4]. Moreover, different data sources often use different formats and structures to represent data. Hence, standardization techniques are used to overcome heterogeneity by transforming data into well-defined and consistent forms [36]. Ideally, all DBOs conduct the same pre-processing steps to reduce heterogeneity thereby facilitating high linkage quality. However, even extensive pre-processing may not resolve all quality issues, as inconsistencies, like contradicting or outdated values, are hard to detect.

**Encoding:** We focus on RL with the additional challenge to preserve the privacy of referenced entities. Consequently, each record needs to be encoded to protect sensitive data. A widely-used approach is to encode each record into a Bloom filter (BF) [43, 47]. A BF [1, 39, 40] is a bit array of fixed size  $m$  where initially all bits are set to zero.  $k$  independent cryptographic hash functions are used to map a set of record features into the BF. Each hash function takes as input every feature from the feature set and produces a value in  $[0, m - 1]$ . Then, the bits at the resulting  $k$  positions are set to one for every feature. The set of features can be extracted in several ways from the record attributes. In general, for each record attribute a function is defined, which takes as input the attribute value and outputs a set of feature values. Typically, all QID values of a record are represented as string and then split into a set of  $q$ -grams (substrings of length  $q$ ), where  $q$  is equal for each attribute. Several BF variants have been proposed for PPRL to either improve quality [20, 44, 45] or privacy properties [8, 35, 38, 40, 41]. Multiple studies have analyzed attacks on BF variants and respective hardening techniques [5, 24, 26, 27, 35].

**Blocking/Filtering:** The trivial approach to link two databases is to compare every possible record pair of the two data sources. To overcome this quadratic complexity blocking or filtering techniques are used to reduce the number of record comparisons [4]. This is achieved by pruning record pairs not fulfilling defined blocking or filter criteria and are hence unlikely considered to be a match. The output of this step are candidate record pairs that need to be further compared. Blocking and filtering can be executed on encoded or unencoded data. Most privacy-preserving approaches perform blocking or filtering at the LU side on encoded data (bit vectors) [47]. State-of-the-art blocking techniques can significantly reduce the search space by applying blocking based on locality-sensitive hashing (LSH) [8, 21, 22] or performing filtering based on multibit-trees [3, 38] or pivot-based filtering for metric distance functions [42].

**Comparison:** Each candidate pair is compared in detail by using (binary) similarity measures, mainly the Jaccard or Dice similarity [43]. The output of this step are candidate pairs with their respective similarity score. The similarity score is a numerical value in  $[0, 1]$  determining how similar two records are.

**Classification:** Most PPRL approaches use a single similarity threshold which is used to classify candidate record pairs into matches, i.e., records representing the same real-world entity, and non-matches [43]. A second threshold can be used to add a third class consisting of possible matches where no clear decision is possible. Another common approach is the probabilistic method developed by Fellegi and Sunter [4, 10].

Finally, the match result, e.g., the IDs of matching record pairs, is returned to the DBOs. However, by using simple threshold-based classification approaches, multi-links occur in the final match result. Since commonly deduplicated databases are assumed, the desired outcome should be a linkage result consisting of only one-to-one links between records. We address this problem by introducing a post-processing step after classification to clean multi-links in the linkage result. The main problem in the post-processing step is to decide which candidates should be selected leading to only one-to-one links and high linkage quality.

## 2.2 Related Work

PPRL and RL have been addressed by numerous research studies and approaches as summarized in several surveys and books [4, 9, 43, 47]. The key challenge of PPRL is to achieve high linkage quality and scalability to potentially large datasets while preserving the privacy of represented entities by using secure encodings and protocols. In order to achieve a high linkage quality previous work mostly focuses on developing or optimizing encoding techniques to support approximate matching, attribute weighting or different data types [19, 43, 45, 46]. Besides, efficient blocking and filtering techniques have been proposed that do not compromise linkage quality outcome [47].

The problem of post-processing corresponds to *weighted bipartite graph matching problems* [48]. In fact, applying a one-to-one matching restriction, i.e., to clean multi-links, is highly related to problems in graph theory like the *assignment problem* (AP) or the *stable marriage problem* (SMP). Various algorithms have been developed to solve such kind of problems [17, 48]. The most prominent approaches are variants of the *Hungarian algorithm* (*Kuhn-Munkres algorithm*) [34] for solving AP as well as variants of the *Gale-Shapley algorithm* [12, 14, 16, 31] for solving SMP.

For PPRL, post-processing methods have only been studied to a limited extent so far: Though several approaches were considered for RL, they were not comparatively evaluated in a PPRL context [2, 4, 8, 9, 28]. As a consequence, it is unknown to which degree post-processing is useful and which method is suited best for PPRL. Note that privacy restrictions only allow simple approaches for match classification so that the need for post-processing is increased for PPRL.

A similar post-processing problem, namely selecting the most probable correspondences from a mapping, has been studied in the field of schema [7, 30, 33] and ontology matching [32]. In [7, 33] best match selection strategies, called *MaxN* or *Perfectionist Egalitarian Polygamy*, are used to enforce a one-to-one cardinality constraint by selecting only candidates offering the best similarity scores. Additionally, algorithms for solving the *maximum weighted bipartite graph matching problem* and the SMP have been considered as selection strategies [30, 32].

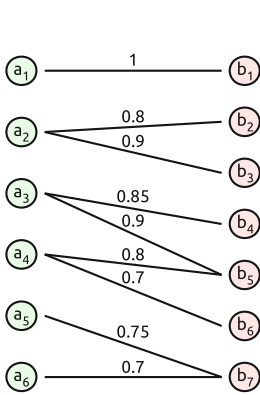
### 3 Problem Definition

After the classification step (see Sect. 2) all candidate record pairs  $\mathbf{C}$  are classified into matches  $\mathbf{C}_{\text{Match}}$  and non-matches  $\mathbf{C}_{\text{Non-Match}}$ . We assume, that a simple threshold-based approach is used for classification. Thus, the class of matches  $\mathbf{C}_{\text{Match}}$  contain all candidate record pairs with a similarity score  $\text{sim}(\cdot, \cdot)$  above a single predefined similarity threshold  $t$ , i.e.,  $\mathbf{C}_{\text{Match}} = \{(a, b) \mid a \in D_A, b \in D_B, \text{sim}(a, b) \geq t\}$ . We also assume that the databases to be linked are deduplicated before linkage.

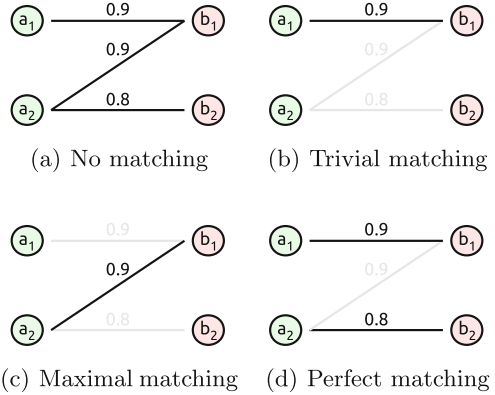
The set of matches  $\mathbf{C}_{\text{Match}}$  constitutes a *weighted bipartite linkage* or similarity graph  $\mathbf{G} = (\mathbf{V}_A, \mathbf{V}_B, \mathbf{L})$ . Let  $\mathbf{V}_A$  and  $\mathbf{V}_B$  be two partitions consisting of vertices representing records (entities) from database  $D_A$  or database  $D_B$  respectively, which occur in the linkage result, i.e., are part of a record pair classified as match. Thus,  $\mathbf{V}_A = \{a \in D_A \mid \exists b \in D_B : (a, b) \in \mathbf{C}_{\text{Match}}\}$  and  $\mathbf{V}_B = \{b \in D_B \mid \exists a \in D_A : (a, b) \in \mathbf{C}_{\text{Match}}\}$ .  $\mathbf{L}$  denotes the set of edges representing links between two records classified as match. Each edge (link) has a property for the similarity score of the record pair. Between records of the same database no direct link exists. An example linkage graph is depicted in Fig. 2.

After classification it is still possible that the linkage graph contains multi-links, i.e., one-to-many, many-to-one or many-to-many links. Since deduplicated databases are assumed, only one-to-one links should be present in the final linkage result. Hence, the aim of post-processing is to find a *matching* (match mapping) over  $\mathbf{G}$ . A matching  $\mathbf{M} \subseteq \mathbf{L}$  is a subset of links such that each record

in  $\mathbf{V} = \mathbf{V}_A \cup \mathbf{V}_B$  appears in at most one link, i.e., contributes to at most one matching record pair. As a consequence, post-processing applies a one-to-one link (cardinality) restriction on the set of classified matches  $\mathbf{C}_{\text{Match}}$ .



**Fig. 2.** Example linkage graph containing several multi-links.



**Fig. 3.** Types of matchings.

In general, several matchings over  $\mathbf{G}$  can be found. Thus, the challenge of post-processing is to select the matching yielding the best linkage quality in terms of either recall, precision or F-measure. Ideally, no true-match should be pruned (no loss of recall) while resolving all multi-links to improve precision. Links providing high similarity scores should be favored over those with a low similarity, e.g., near  $t$ , as very high similarities typically indicate definite matches. Also other link features, like link degree, can be used for link prioritization [37].

A matching can be selected in such a way that it fulfills certain properties. Basic types of matchings are **trivial**, **maximal**, **maximum** and **perfect** matchings [48]. A matching  $\mathbf{M}$  is called *maximal*, if any link not in  $\mathbf{M}$  is added to  $\mathbf{M}$ , then  $\mathbf{M}$  would be no longer a matching. If a matching is not maximal then it is a *trivial matching*. Furthermore, if a matching contains the largest possible number of edges (links) then it is a *maximum matching*. Each maximum matching is also maximal but not vice versa. Finally, a *perfect matching* is defined as a matching where every vertex of the graph is incident to exactly one edge of the matching. Every perfect matching is maximum and hence maximal. However, not for every linkage graph a perfect matching exists. The different types of matchings are illustrated in Fig. 3.

Since PPRL is confronted with potentially large datasets containing millions of records [47], post-processing approaches need to be scalable and efficient.

## 4 Post-processing Strategies for PPRL

We now present post-processing strategies for PPRL to enable a one-to-one link restriction on the linkage result. We chose three frequently used approaches known from schema matching for obtaining matchings in bipartite graphs. The approaches are described in detail below.

### 4.1 Symmetric Best Match

At first, we consider a symmetric best match strategy (SBM) as proposed in [7, 33]. The basic idea is that for every record only the best matching record of the other source is accepted. A record  $a \in V_A$  may have links to several records  $b \in V_B$ . From these links only the one with the highest similarity score, called *best link*, is selected. This approach is equivalent to a MaxN strategy which extracts the maximum N correspondences for each record setting  $N = 1$  (Max1).

To obtain a matching  $M$  over a linkage graph  $G$  for every record of both partitions  $V_A$  and  $V_B$  the best link is extracted. Thus, two sets  $L_A^{Max1}$  and  $L_B^{Max1}$  are build containing the best links for each record of the respective partition, e.g.,  $L_A^{Max1} = \{(a, b) \in L \mid \forall b' \in V_B : (b \neq b' \wedge (a, b') \in L) \rightarrow (sim(a, b') \leq sim(a, b))\}$ . Then, the final matching is obtained by building the intersection of these two sets, i.e.,  $M_{SMB} = M_{Max1-both} = L_A^{Max1} \cap L_B^{Max1}$ . Since the best links from both partitions are considered this strategy is also called *Max1-both*.

In Fig. 4(a) Max1-both is applied on the linkage graph from Fig. 2. It is important to note that the obtained matching is not maximal. Since only record pairs with a common best link are accepted, other record pairs are excluded from the matching even if they do not violate the one-to-one link restriction and have a relative high similarity.

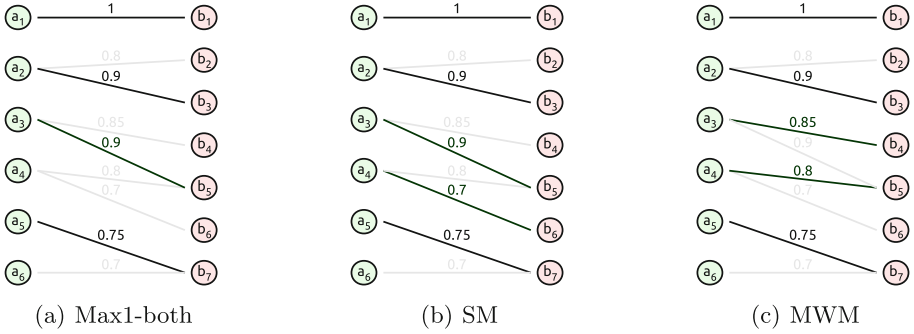
### 4.2 Stable Marriage and Stable Matchings

The stable marriage problem (SMP) [12] is the problem of finding a *stable matching* (SM) between two sets of elements given an (strictly) ordered preference list for each element. A matching is defined as *stable*, if there are no two records of the different partitions who both have a higher similarity to each other than to their current matching record. Used as post-processing method for PPRL several extensions to the classic SMP need to be considered [17, 31]:

**Unequal Sets:** Usually, an SMP instance consists of two sets of elements having the same cardinality. The partitions of the linkage graph are in general of different size, i.e.,  $|V_A| \neq |V_B|$ , as not every record may have a duplicate in the other source.

**Incomplete Preference Lists with Ties:** In traditional SMP each element has a preference list that *strictly* orders *all* members of the other set. Since blocking or filtering techniques are used for PPRL not every record  $a \in V_A$  has a link to a record  $b \in V_B$  and vice versa. Moreover, a record may have two





**Fig. 4.** Illustration of the resulting linkage graph from Fig. 2 after applying different post-processing methods. For Max1-both (a) the link  $a_4-b_6$  is removed since the best link for  $a_4$  is to  $b_5$ . In contrast, for SM (b) the link  $a_4-b_6$  is included as it does not violate the one-to-one-link restriction nor the stable property. For the MWM (c), the links  $a_3-b_4$  and  $a_4-b_5$  are included in the matching as the sum of their similarities is higher than for  $a_3-b_5$  and  $a_4-b_6$ . However, the MWM is not stable due to the links  $a_3-b_4$  and  $a_4-b_5$ , as  $a_3$  and  $b_5$  prefer each other over their current matching records.

links with the same similarity score to two different records of the other source, called *tie* or *indifference* [16], e.g.,  $sim(a, b_1) = 0.9$  and  $sim(a, b_2) = 0.9$  where  $a \in V_A$  and  $b_1, b_2 \in V_B$ . The simplest way to handle indifference is to break ties arbitrary [16]. Also secondary link features can be used for resolving ties [37].

**Symmetry:** For SMP it is not required that two elements prefer each other the same (asymmetric preference). In our case, the SMP instance is symmetric since the similarity of a record pair is symmetric.

To obtain a SM the Gale-Shapley algorithm [12] or one of its variants taking the described extensions into account [16, 17, 31] can be used. A simple approach is to order all links (or candidate pairs) based on their similarity score to process them iteratively in descending order. The current link is added to the final matching if it does not violate the one-to-one-link restriction. The algorithm stops if all links have been processed [30]. In Fig. 4(b) a SM for the linkage graph from Fig. 2 is depicted. In contrast to matchings obtained by the SBM strategy, SMs are maximal. In general, multiple SMs may exist for a linkage graph.

### 4.3 Maximum Weight Matchings

As third method we consider to find a *maximum weight matching* (MWM). A MWM is a matching that has maximum weight, i.e., that maximizes the sum of the overall similarities between records in the final linkage result. This problem corresponds to the assignment problem (AP) which consists of finding a MWM in a weighted bipartite graph. To solve the AP on bipartite graphs in polynomial time the Hungarian algorithm (Kuhn-Munkres algorithm) can be used [34].

For the linkage graph from Fig. 2 the corresponding MWM is depicted in Fig. 4(c). Each MWM is maximal, but does not have to be stable.

## 5 Evaluation

In this section we evaluate the introduced post-processing methods in terms of linkage quality and efficiency. Before presenting the evaluation results we describe our experimental setup as well as the datasets and metrics we use.

### 5.1 Experimental Setup

All experiments are conducted on a desktop machine equipped with an Intel Core i7-6700 CPU with  $8 \times 3.40$  GHz, 32 GB main memory and running Ubuntu 16.04.4. and Java 1.8.0\_171.

### 5.2 PPRL Setup

Following previous work, we implemented the PPRL process as three-party protocol utilizing BF's as privacy technique as proposed by Schnell [40]. To overcome the quadratic complexity we make use of LSH-based blocking utilizing the family of hash functions which is sensitive to the Hamming distance (HLSH) [8]. The respective hash functions are used to build overlapping blocks in which similar records are grouped. For HLSH-based blocking mainly the two parameters  $\Psi$ , determining the number of hash functions used for building a blocking key, and  $\Lambda$ , defining the number of blocking keys, are important for high efficiency and linkage quality outcome [11]. Based on [11] we empirically set  $\Psi$  and  $\Lambda$  individual for each dataset as outlined in Table 1 leading to high efficiency and effectiveness. Finally, we apply the Jaccard similarity to determine the similarity of candidate record pairs [18].

### 5.3 Datasets

For evaluation we use synthetic and real datasets containing one million records with person-related data. An overview about all relevant dataset characteristics and parameters is given in Table 1. The synthetic datasets  $G_1$  and  $G_2$  are generated using the data generator and corruption tool GeCo [6]. We customized the tool by using lookup files containing German names and addresses with realistic frequency values drawn from German census data<sup>1</sup>. Moreover, we extended GeCo by a *family* and *move rate* used for  $G_2$ . The family rate determines how many records of a dataset belong to a family. All records of the same family agree on their last name and address attributes. The size of each family is chosen randomly between two and five. To simulate moves we added a move rate that defines in how many records the address attributes are altered. The move

<sup>1</sup> <https://www.destatis.de/DE/Methoden/Zensus./Zensus.html>.

rate does not introduce data errors like typos, instead it simulates inconsistencies between data sources. A generated dataset  $D$  consists of two subsets,  $D_A$  and  $D_B$ , to be linked with each other. While the original tool requires all records of  $D_B$  to be duplicates to records of  $D_A$  we changed the tool to support arbitrary degrees of overlaps between  $D_A$  and  $D_B$ . As a consequence, records in both  $D_A$  and  $D_B$  may have no duplicate which is more realistic. We also use a refined model to corrupt records by allowing a different number of errors per record instead of a fixed maximum number of errors for all records. We may thus generate duplicates such that 50% of the duplicates contain no error, 20% one error and 10% two errors while the remaining 20% have an address change (move rate). For the real dataset  $N$ , we use subsets of two snapshots of the North Carolina voter registration database (NCVR) at different points in time.<sup>2</sup>

**Table 1.** Dataset characteristics and used parameters.

Characteristic	$G_1$	$G_2$	$N$
<i>Type</i>	Synthetic (GeCo)		Real (NCVR)
$ D_A $	800 000	700 000	500 000
$ D_B $	200 000	300 000	500 000
$ D_A \cap D_B $	200 000 (100 %)	150 000 (50 %)	250 000 (50 %)
<i>Attributes</i>	First name, last name, city, zip, date of birth		First Name, middle name, last name, city, year of birth
<i>q-grams</i>	$q = 2$ (bigrams), no character padding		
<i>g</i>	28		25
$ Errors /record$	2	0 - 2 : 0 (40 %) 1 (30 %) 2 (10 %)	
$ Errors /attr$	0 - 1	0 - 2	
<i>Moves</i>		20 %	
<i>Families</i>		25 %	
<i>m</i>	1024		
<i>k</i>	26		29
<i>BF type</i>	CLK with random hashing [40,41]		
<i>HLSH key length <math>\Psi</math></i>	16		
<i>HLSH keys <math>\Lambda</math></i>	20		30

## 5.4 Evaluation Metrics

To assess the linkage quality we measure *recall*, *precision* and *F-measure*. Recall measures the proportion of true-matches that have been correctly classified as matches after the linkage process. Precision is defined as the fraction of classified matches that are true-matches. Finally, F-measure is the harmonic mean of recall and precision. To evaluate efficiency we measure the *execution times* of the post-processing methods in seconds.

<sup>2</sup> <http://www.ncsbe.gov/>.

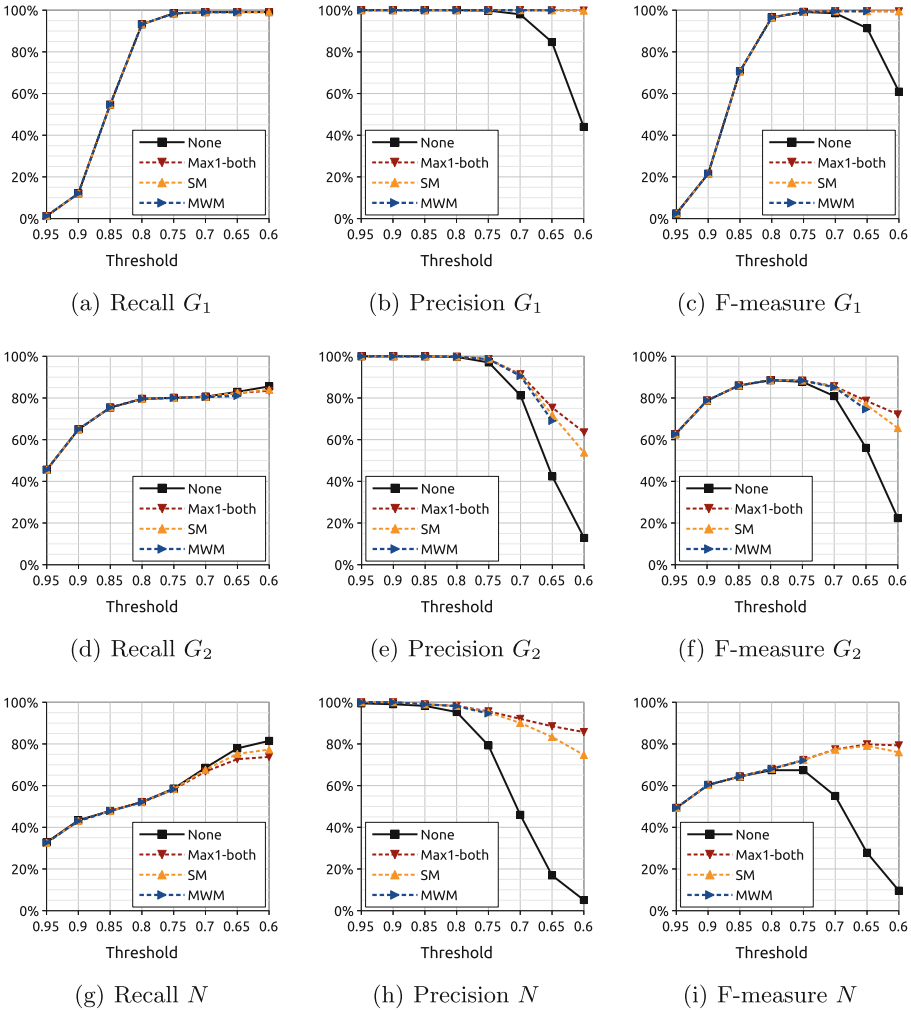


Fig. 5. Quality results for the datasets  $G_1, G_2$  and  $N$ .

### 5.5 Evaluation Results

In order to analyze the impact of post-processing on the linkage quality we compare the three strategies described in Sect. 4 to the standard PPRL without post-processing. The aim of post-processing is to optimize precision while recall is ideally preserved. The results in Fig. 5 show the obtained linkage quality for datasets  $G_1, G_2$  and  $N$ .

Dataset  $G_1$  is based on settings of the original GeCo tool with 100% overlap and a fixed error rate. We observe that a high linkage quality is achieved even if post-processing is disabled with near-perfect recall for  $t \leq 0.8$  and near-perfect precision for  $t \geq 0.7$ . The three post-processing methods achieve very

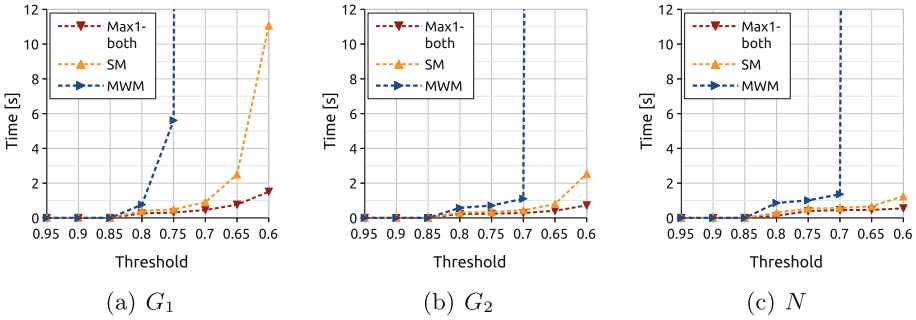
similar results for  $G_1$ : While recall remains stable, precision and consequently F-measure can be significantly improved to almost 100% even for low thresholds  $t \leq 0.7$ . This is due to the high overlap of the two subsets making false-matches after post-processing only possible if a record has a higher similarity to a record having no duplicate than to its actual true-match. Despite this best-case situation simulated with  $G_1$ , only low precision is achieved for low threshold values without post-processing.

For datasets  $G_2$  and  $N$  overall a lower linkage quality is obtained since the data is more dense making it harder to separate matches and non-matches. Similar to  $G_1$ , precision significantly decreases for  $G_2$  using lower threshold values. All post-processing strategies can again improve precision for lower threshold values. The best results are achieved for Max1-both outperforming both SM and MWM. SM yields slightly better results than MWM. For the synthetic datasets  $G_1$  and  $G_2$ , post-processing does not increase the top F-measure but the best linkage quality is reached with a wider range of threshold settings thereby simplifying the choice of a suitable threshold.

The post-processing methods are most effective for the real dataset  $N$ . Here a higher recall can only be achieved for lower threshold values  $t \leq 0.7$  but precision drops dramatically in this range without post-processing due to a high number of multi-links. As a result, the best possible F-measure is limited to only 67%. By contrast, the use of post-processing can maintain a high precision even for lower thresholds at only small decrease in recall compared to disabled post-processing. As a result, the top F-measure is substantially increased to around 80% underlining the high effectiveness and significance of the proposed post-processing. Again, the use of Max1-both is most effective followed by SM.

Additionally, we comparatively evaluated the post-processing strategies in terms of runtime. The results depicted in Fig. 6 show that Max1-both achieves the lowest execution times even for low thresholds. The extended Gale-Shapley algorithm we used for SM shows a significant performance decrease for lower similarity thresholds, most notably for dataset  $N$  and  $t \leq 0.7$ . For higher thresholds  $t > 0.7$  the runtimes are very similar to those of Max1-both. The computation of the MWM by using the Hungarian algorithm incurs a high computational complexity and massive memory consumption. As a consequence, we were not able to obtain a MWM for low threshold values (compare Fig. 5). Hence, we consider the MWM approach as not scalable enough for large datasets with millions of records.

In conclusion, both Max1-both and SM are able to significantly improve the linkage quality of PPRL, especially for low thresholds, while showing good performance. In our setup, the execution of the entire PPRL process takes only a few minutes. Hence, introducing post-processing taking a few seconds for execution does not affect the overall performance. In general, Max1-both can achieve the best linkage quality in terms of precision and F-measure. For applications favoring recall over precision, a SM should be applied.



**Fig. 6.** Runtime results for the datasets  $G_1$ ,  $G_2$  and  $N$ .

## 6 Conclusion

We evaluated different post-processing methods for PPR to restrict the linkage result to only one-to-one links. Our evaluation for large synthetic and real datasets containing one million records showed that without post-processing only low linkage quality is achieved, especially when dealing with dense or dirty data. In contrast, using a symmetric best match strategy for post-processing is a lightweight approach to raise the overall linkage quality. As a side effect, by using post-processing the similarity threshold used for classification can be selected lower without compromising linkage quality. Since in practical applications a appropriate threshold is hard to define, this fact becomes highly beneficial.

In future, we plan to investigate further post-processing strategies using further link features and other heuristics. We also plan to analyze post-processing methods for multi-party PPR where more than two databases need to be linked.

## References

1. Bloom, B.: Space/time trade-offs in hash coding with allowable errors. *CACM* **13**(7), 422–426 (1970)
2. Böhm, C., de Melo, G., Naumann, F., Weikum, G.: LINDA: distributed web-of-data-scale entity matching. In: *ACM CIKM*, pp. 2104–2108 (2012)
3. Brown, A.P., Borgs, C., Randall, S.M., Schnell, R.: Evaluating privacy-preserving record linkage using cryptographic long-term keys and multibit trees on large medical datasets. *BMC Med. Inf. Decis. Making* **17**(1), 83 (2017)
4. Christen, P.: *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-31164-2>
5. Christen, P., Schnell, R., Vatsalan, D., Ranbaduge, T.: Efficient cryptanalysis of bloom filters for privacy-preserving record linkage. In: Kim, J., Shim, K., Cao, L., Lee, J.-G., Lin, X., Moon, Y.-S. (eds.) *PAKDD 2017. LNCS (LNAI)*, vol. 10234, pp. 628–640. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-57454-7\\_49](https://doi.org/10.1007/978-3-319-57454-7_49)
6. Christen, P., Vatsalan, D.: Flexible and extensible generation and corruption of personal data. In: *ACM CIKM*, pp. 1165–1168 (2013)

7. Do, H.H., Rahm, E.: COMA - a system for flexible combination of schema matching approaches. In: VLDB, pp. 610–621 (2002)
8. Durham, E.A.: A framework for accurate, efficient private record linkage. Ph.D. thesis, Vanderbilt University (2012)
9. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: a survey. *IEEE TKDE* **19**(1), 1–16 (2007)
10. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. *JASA* **64**(328), 1183–1210 (1969)
11. Franke, M., Sehili, Z., Rahm, E.: Parallel privacy preserving record linkage using LSH-based blocking. In: IoTBDS, pp. 195–203 (2018)
12. Gale, D., Shapley, L.S.: College admissions and the stability of marriage. *Am. Math. Mon.* **69**(1), 9–15 (1962)
13. Gibberd, A., Supramaniam, R., Dillon, A., Armstrong, B.K., OConnell, D.L.: Lung cancer treatment and mortality for Aboriginal people in New South Wales, Australia: results from a population-based record linkage study and medical record audit. *BMC Cancer* **16**(1), 289 (2016)
14. Gusfield, D., Irving, R.W.: *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, Cambridge (1989)
15. Hernández, M.A., Stolfo, S.J.: Real-world data is dirty: data cleansing and the merge/purge problem. *Data Min. Knowl. Discovery* **2**(1), 9–37 (1998)
16. Irving, R.W.: Stable marriage and indifference. *Discrete Appl. Math.* **48**(3), 261–272 (1994)
17. Iwama, K., Miyazaki, S.: A survey of the stable marriage problem and its variants. In: *IEEE ICKS*, pp. 131–136 (2008)
18. Jaccard, P.: The distribution of the flora in the alpine zone. *New Phytol.* **11**(2), 37–50 (1912)
19. Karapiperis, D., Gkoulalas-Divanis, A., Verykios, V.S.: Distance-aware encoding of numerical values for privacy-preserving record linkage. In: *IEEE ICDE*, pp. 135–138 (2017)
20. Karapiperis, D., Gkoulalas-Divanis, A., Verykios, V.S.: FEDERAL: a framework for distance-aware privacy-preserving record linkage. *IEEE TKDE* **30**(2), 292–304 (2018)
21. Karapiperis, D., Verykios, V.S.: A distributed framework for scaling up LSH-based computations in privacy preserving record linkage. In: *Proceedings of the BCI* (2013)
22. Karapiperis, D., Verykios, V.S.: A fast and efficient hamming LSH-based scheme for accurate linkage. *KAIS* **49**(3), 861–884 (2016)
23. Kho, A.N., Cashy, J.P., Jackson, K.L., Pah, A.R., Goel, S., Boehnke, J., Humphries, J.E., Kominers, S.D., Hota, B.N., Sims, S.A., et al.: Design and implementation of a privacy preserving electronic health record linkage tool in Chicago. *JAMIA* **22**(5), 1072–1080 (2015)
24. Kroll, M., Steinmetzer, S.: Automated cryptanalysis of bloom filter encryptions of health records. In: *ICHI* (2014)
25. Kuehni, C.E., et al.: Cohort profile: the Swiss childhood cancer survivor study. *Int. J. Epidemiol.* **41**(6), 1553–1564 (2012)
26. Kuzu, M., Kantarcioglu, M., Durham, E., Malin, B.: A constraint satisfaction cryptanalysis of bloom filters in private record linkage. In: Fischer-Hübner, S., Hopper, N. (eds.) *PETS 2011*. LNCS, vol. 6794, pp. 226–245. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-22263-4\\_13](https://doi.org/10.1007/978-3-642-22263-4_13)

27. Kuzu, M., Kantarcioglu, M., Durham, E.A., Toth, C., Malin, B.: A practical approach to achieve private medical record linkage in light of public resources. *JAMIA* **20**(2), 285–292 (2013)
28. Lenz, R.: Measuring the disclosure protection of micro aggregated business micro-data: an analysis taking as an example the German structure of costs survey. *J. Official Stat.* **22**(4), 681 (2006)
29. Luo, Q., et al.: Cancer-related hospitalisations and unknownstage prostate cancer: a population-based record linkage study. *BMJ Open* **7**(1), e014259 (2017)
30. Marie, A., Gal, A.: On the Stable Marriage of Maximum Weight Royal Couples. In: *AAAI Workshop on Information Integration on the Web* (2007)
31. McVitie, D.G., Wilson, L.B.: Stable marriage assignment for unequal sets. *BIT Numer. Math.* **10**(3), 295–309 (1970)
32. Meilicke, C., Stuckenschmidt, H.: Analyzing mapping extraction approaches. In: *OM*, pp. 25–36 (2007)
33. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: a versatile graph matching algorithm and its application to schema matching. In: *IEEE ICDE*, pp. 117–128 (2002)
34. Munkres, J.: Algorithms for the assignment and transportation problems. *SIAM J.* **5**(1), 32–38 (1957)
35. Niedermeyer, F., Steinmetzer, S., Kroll, M., Schnell, R.: Cryptanalysis of basic bloom filters used for privacy preserving record linkage. *JPC* **6**(2), 59–79 (2014)
36. Rahm, E., Do, H.H.: Data cleaning: problems and current approaches. *IEEE Data Eng. Bull.* **23**(4), 3–13 (2000)
37. Saeedi, A., Peukert, E., Rahm, E.: Using link features for entity clustering in knowledge graphs. In: Gangemi, A., et al. (eds.) *ESWC 2018. LNCS*, vol. 10843, pp. 576–592. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-93417-4\\_37](https://doi.org/10.1007/978-3-319-93417-4_37)
38. Schnell, R.: Privacy-preserving record linkage. In: *Methodological Developments in Data Linkage*, pp. 201–225 (2015)
39. Schnell, R., Bachteler, T., Reiher, J.: Privacy-preserving record linkage using Bloom filters. *BMC Med. Inf. Decis. Making* **9**(1), 41 (2009)
40. Schnell, R., Bachteler, T., Reiher, J.: A novel error-tolerant anonymous linking code. *GRLC*, No. WP-GRLC-2011-02 (2011)
41. Schnell, R., Borgs, C.: Randomized response and balanced bloom filters for privacy preserving record linkage. In: *IEEE ICDMW* (2016)
42. Sehili, Z., Rahm, E.: Speeding up privacy preserving record linkage for metric space similarity measures. *Datenbank-Spektrum* **16**(3), 227–236 (2016)
43. Vatsalan, D., Christen, P., Verykios, V.S.: A taxonomy of privacy-preserving record linkage techniques. *Inf. Syst.* **38**(6), 946–969 (2013)
44. Vatsalan, D., Christen, P.: scalable privacy-preserving record linkage for multiple databases. In: *ACM CIKM*, pp. 1795–1798 (2014)
45. Vatsalan, D., Christen, P.: Privacy-preserving matching of similar patients. *J. Biomed. Inf.* **59**, 285–298 (2016)
46. Vatsalan, D., Christen, P., O’Keefe, C.M., Verykios, V.S.: An evaluation framework for privacy-preserving record linkage. *JPC* **6**(1), 3 (2014)
47. Vatsalan, D., Sehili, Z., Christen, P., Rahm, E.: Privacy-preserving record linkage for big data: current approaches and research challenges. In: Zomaya, A.Y., Sakr, S. (eds.) *Handbook of Big Data Technologies*, pp. 851–895. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-49340-4\\_25](https://doi.org/10.1007/978-3-319-49340-4_25)
48. West, D.B., et al.: *Introduction to Graph Theory*, vol. 2. Prentice Hall, Upper Saddle River (2001)