Scalable and Approximate Privacy-Preserving Record Linkage

Dinusha Vatsalan

A thesis submitted for the degree of
Doctor of Philosophy in Computer Science
The Australian National University

October 2014

Except where otherwise indicated, this thesis is my own original work.

Dinusha Vatsalan
24 October 2014

to all the loved ones

Acknowledgments

The accomplishment of the objectives of this research study would not have been viable without the support of a great number of people.

I would like to first thank my primary supervisor, Peter Christen, whom I have been extremely fortunate to work under. The support, guidance, motivation, and directions he has given me throughout the period are incredible. I am extremely thankful for the immense amount of time Peter has spent on reading all my drafts and providing detailed comments and valuable suggestions that significantly helped to improve my work.

I would like to acknowledge my second supervisor, Vassilios S. Verykios, who has been very supportive and helpful in providing constructive feedback on my research with his expertise and proof-reading my writing despite his busy schedule. I'm also grateful to my advisor, Lexing Xie, for her time, advice, discussions, and encouragement throughout.

In addition, I would also like to extend my thanks to Christine M. O'Keefe and Khoi-Nguyen Tran, whom I've co-authored with, for their valuable contributions.

I thank the Australian National University (ANU) for offering me an opportunity to conduct my research studies. The university is well supportive of students, and their progress, as well as their wellbeing.

I am honored to be an Endeavour Postgraduate Research Award recipient from the Australian Government Department of Education, Employment and Workplace Relations (DEEWR). I'm sincerely thankful for the funding provided by the Endeavour scholarship for my studies, without which it would not have been possible.

I have been fortunate to work within a team of great people. Thanks to Banda Ramadan, Minkyoung Kim, Zhichun Fu, Khoi-Nguyen Tran, Jeffrey Fisher, Huizhi Liang, and Thilina Ranbaduge for your feedback, support, and encouragement.

And finally, last but not least, I would like to thank my husband, parents, siblings, and friends for all their love, support, understanding, and encouragement along the way to the completion of this work.

Abstract

Record linkage, the task of linking multiple databases with the aim to identify records that refer to the same entity, is occurring increasingly in many application areas. Generally, unique entity identifiers are not available in all the databases to be linked. Therefore, record linkage requires the use of personal identifying attributes, such as names and addresses, to identify matching records that need to be reconciled to the same entity. Often, it is not permissible to exchange personal identifying data across different organizations due to privacy and confidentiality concerns or regulations. This has led to the novel research area of privacy-preserving record linkage (PPRL).

PPRL addresses the problem of how to link different databases to identify records that correspond to the same real-world entities, without revealing the identities of these entities or any private or confidential information to any party involved in the process, or to any external party, such as a researcher. The three key challenges that a PPRL solution in a real-world context needs to address are (1) scalability to large databases by efficiently conducting linkage; (2) achieving high quality of linkage through the use of approximate (string) matching and effective classification of the compared record pairs into matches (i.e. pairs of records that refer to the same entity) and non-matches (i.e. pairs of records that refer to different entities); and (3) provision of sufficient privacy guarantees such that the interested parties only learn the actual values of certain attributes of the records that were classified as matches, and the process is secure with regard to any internal or external adversary.

In this thesis, we present extensive research in PPRL, where we have addressed several gaps and problems identified in existing PPRL approaches. First, we begin the thesis with a review of the literature and we propose a taxonomy of PPRL to characterize existing techniques. This allows us to identify gaps and research directions. In the remainder of the thesis, we address several of the identified shortcomings. One main shortcoming we address is a framework for empirical and comparative evaluation of different PPRL solutions, which has not been studied in the literature so far. Second, we propose several novel algorithms for scalable and approximate PPRL by addressing the three main challenges of PPRL. We propose efficient private blocking techniques, for both three-party and two-party scenarios, based on sorted neighborhood clustering to address the scalability challenge. Following, we propose two efficient two-party techniques for private matching and classification to address the linkage quality challenge in terms of approximate matching and effective classification. Privacy is addressed in these approaches using efficient data perturbation techniques including k-anonymous mapping, reference values, and Bloom filters. Finally, the thesis reports on an extensive comparative evaluation of our proposed solutions with several other state-of-the-art techniques on real-world datasets, which shows that our solutions outperform others in terms of all three key challenges.

List of Publications

1. **Major Contributions:**

   (a) **An Evaluation Framework for Privacy-Preserving Record Linkage:** Dinusha Vatsalan, Peter Christen, Christine M. O'Keefe, and Vassilios S. Verykios. In Journal of Privacy and Confidentiality (JPC), Volume 6, Issue 1, 2014, Pages 35-75 - *corresponds to Chapters 5 and 10 of this thesis*.

   (b) **A Taxonomy of Privacy-Preserving Record Linkage Techniques:** Dinusha Vatsalan, Peter Christen, and Vassilios S. Verykios. In Elsevier Journal of Information Systems (JIS), Volume 38, Issue 6, September 2013, Pages 946-969 - *corresponds to Chapters 3 and 4 of this thesis*.

   (c) **Efficient Two-Party Private Blocking based on Sorted Nearest Neighborhood Clustering:** Dinusha Vatsalan, Peter Christen, and Vassilios S. Verykios. In proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13), San Francisco, United States, October 2013. In ACM Digital Library, ISBN: 978-1-4503-2263-8, Pages 1949-1958 (Full paper, Acceptance rate 16.86%) - *corresponds to Chapter 7 of this thesis*.

   (d) **Sorted Nearest Neighborhood Clustering for Efficient Private Blocking:** Dinusha Vatsalan, and Peter Christen. In proceedings of the Seventeenth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'13), Gold Coast, Australia, April 2013. In Springer Lecture Notes in Computer Science, Volume 7819, Pages 341-352 (Full paper, Acceptance rate 11.34%) - *corresponds to Chapter 6 of this thesis*.

   (e) **An Iterative Two-Party Protocol for Scalable Privacy-Preserving Record Linkage:** Dinusha Vatsalan, and Peter Christen. In proceedings of the Tenth Australasian Data Mining Conference (AusDM'12), Sydney, December 2012. In Conferences in Research and Practice in Information Technology (CRPIT), Volume 134, Pages 127-138 (Full paper, Acceptance rate 45%) - *corresponds to Chapter 8 of this thesis*.

   (f) **An Efficient Two-Party Protocol for Approximate Matching in Private Record Linkage:** Dinusha Vatsalan, Peter Christen, and Vassilios Verykios. In proceedings of the Ninth Australasian Data Mining Conference (AusDM' 11), Ballarat, December 2011. In Conferences in Research and Practice in Information Technology (CRPIT), Volume 121, Pages 125-136 (Full paper, Acceptance rate 44%) - *corresponds to Chapter 9 of this thesis*.

2. **Minor Contributions:**

   (a) **Challenges for privacy preservation in data integration:** Peter Christen, Dinusha Vatsalan, and Vassilios S. Verykios. To appear in ACM Journal of Data and Information Quality (JDIQ), 2014 - *used in Chapter 11 of this thesis*.

   (b) **Flexible and extensible generation and corruption of personal data:** Peter Christen, and Dinusha Vatsalan. In proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM' 13), San Francisco, United States, October 2013. In ACM Digital Library, ISBN: 978-1-4503-2263-8, Pages 1165-1168 (Poster paper, Acceptance rate 37.72%) - *used in Chapter 5 of this thesis*.

   (c) **GeCo: an online personal data generator and corruptor:** Khoi-Nguyen Tran, Dinusha Vatsalan, and Peter Christen. In proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13), San Francisco, United States, October 2013. In ACM Digital Library, ISBN: 978-1-4503-2263-8, Pages 2473-2476 (Demo paper) - *used in Chapter 5 of this thesis*.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter provides an introduction to the research problem addressed by this thesis in Section 1.1, real-world applications of the problem in Section 1.2, the aim of the research study in Section 1.3, the contributions of this work in Section 1.4, and the methodology of how the problem is addressed in Section 1.5. An outline to the organization of the thesis is also given at the end of the chapter, as are the notation and terminology used throughout this thesis.

1.1 Problem Statement

In recent times the world has seen an explosion in the volume of data that is being collected by organizations as well as individuals. Much of these data are about people, or they are generated by people. Examples of the former include financial data such as shopping transactions, telecommunication records, or electronic health records. Examples of the latter include emails, tweets, blog posts, and so on. It has been recognized that analyzing large data collections through the use of data mining and analytics techniques can provide a competitive edge to a commercial enterprise, can allow improved crime and fraud detection, can lead to better patient outcomes in the health sector, and can be of vital importance to national security [29, 80].

Analyzing and mining large datasets often requires information from multiple data sources to be integrated in order to enable more sophisticated analysis. Integrating data also improves the quality of data by allowing the identification (and possible automatic correction) of conflicting data values, the enrichment of data, or the imputation of missing values [86]. The analysis of integrated data can, for example, facilitate the detection of adverse drug reactions in particular patient groups, or enable the accurate identification of terrorism suspects [24, 63].

The process of matching and aggregating records that relate to the same entity from one or more datasets is known as 'record linkage', 'data matching' or 'entity resolution' [63, 86]. In computer science, a long line of research has been conducted in record linkage, based on the theoretical foundation provided by Fellegi and Sunter in 1969 [65]. Today, record linkage not only faces computational and operational challenges due to the increasing size of datasets, but also privacy preservation challenges due to growing privacy concerns. Generally, record linkage is a challenging

task because unique entity identifiers are not available in all the databases that are linked. Therefore, the common attributes available which are sufficiently well correlated with entities, known as quasi-identifiers (QIDs) [47], need to be used for the linkage. For databases that contain personal information about people, these QID attributes generally include names, addresses, dates of birth, and other details. Using such personal information often leads to privacy and confidentiality concerns.

Record linkage aims to classify the pairs of records from different databases into matches (i.e. pairs of records that refer to the same entity) and non-matches (i.e. pairs of records that refer to different entities) based on the matching / comparison results of the QIDs [33]. The presence of real-world data errors makes this classification task more challenging. In practice, the matching of two records is generally determined by applying similarity comparison functions between QID attributes to calculate how similar the records of a pair are. We outline the three key challenges that are associated with the record linkage problem in the following.

1. **Scalability:** The first challenge of record linkage is the scalability to large databases which is generally dependent on the complexity of the process. Assume two databases that are to be linked, $\mathbf{D^A}$ and $\mathbf{D^B}$, contain $n^A = |\mathbf{D^A}|$ and $n^B = |\mathbf{D^B}|$ records, respectively. In order to classify the record pairs (a, b) from these two databases ($a \in \mathbf{D}^A$ and $b \in \mathbf{D}^B$) into matches and non-matches, in a naïve approach the number of comparisons required is the product of the size of the two databases ($n^A \times n^B$) which is the bottleneck of the whole linkage process [13, 33]. This quadratic complexity makes naïve linkage not scalable to large databases. Blocking techniques can be used to overcome this problem [29] as will be discussed further in Chapter 2. The complexity of record linkage also depends on the techniques employed. Complex techniques for linkage, such as secure multi-party computation techniques [41, 78, 135] or advanced classification techniques including machine learning or graph-based approaches [15, 85, 153], generally have higher computational complexity and therefore they might not be scalable to large databases.

2. **Linkage quality:** It is commonly accepted that real-world data are 'dirty' [84], which means they contain errors, variations, values can be missing, or can be out of date. Therefore, even when records that correspond to the same real-world entity are being compared using the values of their personal identifying details (QIDs), the variations and errors in these values will lead to ambiguous matches [23]. The exact comparison of QID values is therefore not sufficient to achieve accurate linkage results. Approximate matching as well as accurate classification techniques are needed to achieve accurate linkage quality in record linkage applications [23, 44].

3. **Privacy:** When personal information about people (contained in QIDs) is used for the linking of databases across organizations, then the privacy of this information needs to be carefully protected. Individual databases can contain information that is already highly sensitive, such as medical or financial de-

> tails of individuals, or confidential business data. When linked, detailed information about individuals that is even more revealing might become available, such as for people who have certain chronic diseases and who also have financial problems; or confidential business information like the amount a company owes to all its suppliers. It is therefore paramount that the privacy of data used for record linkage across organizations, as well as the sensitive details of the matching results of such a linkage, are preserved throughout the linkage process [40].

The privacy requirements in the record linkage process led to the development of a new research avenue called the 'privacy-preserving record linkage' (PPRL), 'blind data linkage', or 'private record linkage' problem [37, 58, 78, 194]. In this thesis, we use the name 'PPRL' to state this problem. PPRL is the problem of how to identify matching records in different databases that refer to the same entities without compromising privacy and confidentiality of the entities represented by these records.

In today's organizations it is often not legally and ethically allowed in many countries to share data across organizations due to the growing concerns of privacy and confidentiality. The recently established program by the Office for National Statistics (ONS) in the UK, 'Beyond 2011', for example carries out research to study the options for production of population and socio-demographics statistics for England and Wales, by linking anonymous data to ensure that high levels of privacy of data about people are maintained [68]. The Data-Matching Program Act in Australia [1], the European Union (EU) Personal Data Protection Act in Europe [2], and the Health Insurance Portability and Accountability Act (HIPAA) in the USA [3] are few examples that describe the legal restrictions of disclosing private or sensitive data.

In a PPRL project, the database owners (or data custodians) agree to reveal only selected information about matched records among them, or to an external party, such as a researcher. However, to identify the matched records, generally the QIDs need to be revealed between the parties involved in the PPRL process. Personal information contained in the QIDs is often not allowed to be shared or exchanged between different organizations due to privacy concerns or legal requirements. Therefore, the linkage has to be conducted on an encoded and / or perturbed version of the QIDs to preserve the privacy of entities. Encoding and / or perturbation is also known as 'masking', i.e. the original data are transformed in such a way that there exists a specific functional relationship between the original data and the masked data [67].

Generally, two approaches are adopted to conduct the linkage on the masked data: (1) two-party linkage where only the database owners participate in the protocol and (2) three-party linkage where a third party is involved to perform the linkage. The advantages and drawbacks of these two types of protocols will be discussed in Section 1.3 (and in detail in Chapter 5). At the end of the linkage process, the database owners agree to reveal some of the selected attributes of the record pairs

[1] http://www.privacy.gov.au/law/other/datamatch [Accessed: 02/12/2013]
[2] http://ec.europa.eu/justice/data-protection/index_en.htm [Accessed: 02/12/2013]
[3] http://www.hhs.gov/ocr/privacy/ [Accessed: 02/12/2013]

that were classified as matches. However, the PPRL process is required to guarantee that any information regarding the non-matching records (that could be used to identify or infer the non-matching entities) is not revealed during or after the process. In Section 1.2 we describe several example scenarios where PPRL is required in real-world applications. We formally define the problem of PPRL as follows.

Definition 1.1. Privacy-preserving record linkage (PPRL):
Assume $\mathbf{O}_1, \ldots, \mathbf{O}_m$ are the m owners of the databases $\mathbf{D}^1, \ldots, \mathbf{D}^m$, respectively. They wish to determine which of their records $R_i^1 \in \mathbf{D}^1$, $R_j^2 \in \mathbf{D}^2$, ..., $R_k^m \in \mathbf{D}^m$ match based on their (masked) QIDs according to a decision model $C(R_i^1, R_j^2, \ldots, R_k^m)$ that classifies record pairs into one of the two classes **M** of matches, and **U** of non-matches. $\mathbf{O}_1, \ldots, \mathbf{O}_m$ do not wish to reveal their actual records $R_i^1, \ldots, R_k^m$ with any other party. They however are prepared to disclose to each other, or to an external party, the actual values of some selected attributes of the record pairs that are in class **M** to allow analysis.

A viable PPRL solution that can be used in real-world applications should address all three challenges (or properties) of scalability, linkage quality, and privacy. There have been many different approaches proposed for PPRL as recently surveyed in [184, 193]. As described in these surveys, some attempts to address the problem of PPRL fall short in providing a sound solution, either because they are not scalable to large databases, because they do not provide sufficient privacy guarantees, or because they are unable to provide high linkage quality. A review of existing PPRL techniques is presented in Chapter 3.

1.2 Applications of PPRL

Linking records from different databases with the aim to improve data quality or enrich data for further analysis and mining is occurring in an increasing number of application areas including healthcare, government services, crime and fraud detection, and business applications [29]. For example, health researchers are interested in aggregating health databases from different organizations for quality health data mining such as epidemiological studies or to investigate adverse drug reactions [21, 139]. Linked health databases can also be used to develop health policies in a more efficient and effective way compared to the use of small-scale and time-consuming survey studies which traditionally have been used for this purpose [39, 110].

Another application of record linkage is the linking of census data to provide an easy platform for compiling data for different studies, which can then be further analyzed statistically [205]. Record linkage is increasingly being required by social scientists in the field of population informatics to study insights into our society from the 'social genome' data (i.e., person-level data about social being) [120]. Record linkage techniques are also being used by national security agencies and crime investigators to effectively identify individuals who have committed fraud or crimes [98, 156, 195]. Many businesses take advantage of record linkage techniques for deduplicating their

list of customers, which helps them to reduce the cost of running an advertising campaign or conducting other types of marketing activities. Businesses which collaborate often need to link records across their databases for successful collaborations.

When record linkage is applied within a single organization (i.e., only data owned by the same organization are linked), then generally privacy and confidentiality are not of great concern (assuming there are no internal threats within the organization). However, when data from several organizations are linked, then privacy and confidentiality need to be carefully considered, as the following scenarios illustrate.

1. **Public health research**: Assume a group of public health researchers aims to investigate the types of injuries caused by car accidents, with the objective to uncover correlations between types of accidents and the resulting injuries [24]. Such research can have significant impact on policy changes that potentially save many lives [39]. This research requires data from hospitals, the police, as well as public and private health insurers. Neither of these parties is willing or allowed by law to provide their databases to the researchers [176]. The researchers only require access to some attributes of the records that are matched across all the different databases, such as the medical details and basic biographic information, like age and gender, of people who were involved in accidents. An effective governance model has recently been proposed for health data linkage that specifies privacy policies, guiding principles, best practices, and roles and responsibilities of participants of such linkage project [176].

2. **Health surveillance**: Preventing infectious diseases early before they spread widely around a country or worldwide is important for a healthy nation. Such prevention can be done by continuously monitoring early occurrences of infectious diseases. Such early outbreak detection systems require data from several sources to be collected and linked on an ongoing basis, such as human health data, consumed drugs data, and animal health data [40]. Privacy concerns arise when such data are linked and stored at a central location [139]. Techniques are needed to ensure that private patient data, as well as the confidential data collected from healthcare organizations, are kept confidential and secure.

3. **Business collaboration**: Collaboration benefits businesses for example in improving efficiency and reducing the costs of their supply chains. However, businesses generally are not willing to share confidential data, such as strategies and competitive knowledge. Linking the supplier and customer databases between two businesses needs to be conducted without revealing any knowledge besides the suppliers and customers that are present in both databases [40].

4. **Plagiarism detection**: Plagiarism detection, another related PPRL application, is useful in many real-world applications [127, 146] to detect copyright violations, research work duplications, and copied text segments within documents. In many of these cases, this information is confidential and cannot be shared to detect plagiarism. However, without having access to the confidential source,

violations and duplications cannot be identified. PPRL techniques can solve this problem by performing the matching of long texts (considered as records) without disclosing the confidentiality to any parties.

5. **Serious and organized crime**: Imagine a national crime investigation unit which is tasked with fighting against crimes that are of national significance, such as organized crime syndicates. Such a unit will likely manage various national databases which draw from many different sources, including law enforcement agencies, Internet service providers, and financial institutions. Such data are highly sensitive. The collection of such data in one place for retrieval and analysis makes them vulnerable to both outsider attacks and internal adversaries, such as employees who access certain records without authorization. Generally employees are asked by the organization to sign disclosure agreements for accessing confidential data in order to reduce internal threats. Employing techniques that facilitate linking without the need of all data being given to the crime investigation unit would mean that only linked records (such as those of suspicious individuals) are available to the unit. This would significantly reduce any risks of privacy and confidentiality breaches.

1.3 Aim of Research

An optimal PPRL solution should balance all three properties of scalability, linkage quality, and privacy, as described in Section 1.1, which have a trade-off among each other. While various approaches have been proposed to deal with privacy within the record linkage process [193], a practical solution that is well applicable to real-world conditions needs to address the major challenge of scalability of linking very large databases while preserving privacy and achieving high linkage quality.

Privacy needs to be preserved in the linkage process by calculating the similarity of the encoded and / or perturbed (also known as 'masked', as will be described in Chapter 5) attribute values of two records without revealing the actual attribute values of the record pair. Linkage quality can be defined by both the degree of fault-tolerance to real-world data errors (measured by the ability to perform approximate linkage in the presence of typographical errors and other variations in real-world data) and the accuracy of classification. High linkage quality can be achieved by using approximate similarity comparison functions to compensate for data errors, and by using an effective classification model to accurately classify the compared record pairs into matches and non-matches.

Developing approximate and scalable PPRL algorithms without compromising privacy and quality of linkage is an emerging research problem. The primary aim of this study is to conduct extensive research on scalable and approximate PPRL, and the following is a list of research questions that the thesis aims to address.

1. **Extensive survey of PPRL:** Various techniques have been developed for PPRL over the past two decades. An extensive survey of current PPRL techniques is

required to provide insights into the shortcomings of current techniques and directions for future research. PPRL techniques involve many different dimensions and hence characterizing existing PPRL techniques according to such different dimensions for analysis and comparison is challenging, yet very useful.

2. **Efficient privacy techniques:** The approaches proposed for PPRL in the literature can be classified into two based on the privacy techniques employed: (1) data perturbation privacy techniques [108, 109, 199] and (2) Secure Multiparty Computation (SMC) privacy techniques [41, 133, 135]. The first category of techniques perturb private data by attributes reduction, generalization, or transformation, to prevent re-identification of individual records. The privacy and quality of linkage provided by these perturbation-based solutions have a trade-off, in that increasing one often means lowering the other, and vice versa. On the other hand, SMC-based solutions provide highly secure solutions with high linkage quality, but they generally are very expensive in terms of runtime and memory space required, and thus they are impractical in many real-world scenarios. Hence, efficient data perturbation-based privacy techniques with high linkage quality and sufficient privacy protection need to be developed and employed for practical PPRL applications.

3. **Two-party efficient algorithms:** Existing PPRL techniques can also be categorized based on their need (or not) of a third party for performing record linkage [24, 28, 194]. In three-party protocols, a (trusted) third party is involved in conducting the linkage, while in two-party protocols only the two database owners participate in the PPRL process. Three-party protocols would often not suffice in real-world applications since they have the risk of collusion between one of the database owners and the third party with the aim to learn the other database owner's sensitive data. However, two-party protocols generally require more complex techniques to ensure that the two database owners cannot infer any sensitive information from each other during the linkage process. Most of the two-party solutions proposed for PPRL in the literature are SMC-based techniques and are not scalable and practical in real-world settings. Thus, developing efficient two-party protocols that employ cost-effective privacy techniques and preserve the privacy of sensitive data at the same time with less accuracy loss is an important research question.

4. **Private blocking techniques:** There have been various advances with regard to the quality of linking and privacy of PPRL in recent times [102, 184]. Scalability for PPRL, however, is still a major concern, and scaling the linkage process to real-world databases that contain many millions of records without compromising privacy and quality is a challenging task. Similar to blocking techniques that have been used in traditional record linkage [30], private blocking techniques can be used in PPRL to reduce the large number of comparisons required between records by removing potential non-matching record pairs before comparing them in detail using private matching and classification

techniques. Private blocking techniques not only provide a trade-off between complexity and quality, but they also have a trade-off with privacy.

5. **Evaluation framework for PPRL:** The evaluation of PPRL techniques in terms of the three properties of scalability, linkage quality, and privacy is important to allow the assessment and comparison of different solutions. Measuring privacy is difficult compared to quality and scalability for which widely accepted and used measures exist [30, 161]. A general framework with numerical and normalized measures for all three properties of PPRL will provide a baseline for comparison and analysis of PPRL solutions. Developing such a framework is therefore important for PPRL research.

1.4 Contributions of this Work

This thesis provides a detailed study of PPRL techniques. Specifically, it proposes new algorithms for scalable and approximate PPRL addressing several gaps in existing PPRL research. Contributions of the study are visualized in Figure 4.2 on Page 43 (which we will discuss in detail in Section 4.3). We categorize the contributions into three, which are (a) conceptual, (b) methodology, and (c) evaluation. The thesis mainly covers:

(a) Conceptual:

1. **A taxonomy of PPRL techniques** (Chapters 3 and 4): As discussed in Section 1.3, conducting an extensive survey of PPRL techniques with regard to different dimensions of PPRL is important to analyze the shortcomings in the current approaches. We are the first to carry out such a large-scale survey in PPRL. We present a taxonomy of PPRL techniques that characterizes existing PPRL techniques along 15 dimensions of PPRL in Chapter 4. These 15 dimensions are categorized into five main topics which are privacy aspects, linkage techniques, theoretical analysis, evaluation, and practical aspects. We then characterize around 40 PPRL techniques that have been proposed in the literature in the last two decades (as surveyed in Chapter 3) along the proposed taxonomy, and we analyze the gaps that exist in existing techniques that will provide directions into future research.

(b) Evaluation:

2. **An evaluation framework for PPRL** (Chapter 5): One main shortcoming (identified in our survey) we address in this thesis is an evaluation framework for PPRL solutions. We propose a general framework with normalized measures to practically evaluate and compare different PPRL solutions with regard to the three properties of scalability, linkage quality, and privacy. While the scalability and linkage quality properties can be assessed based on available standard measures (such as precision, recall, reduction ratio, pairs completeness, etc.)

that will be discussed in Sections 5.3.2 and 5.3.3, respectively, the privacy protection provided by a PPRL technique is comparatively more difficult to assess. The proposed evaluation framework introduces a novel set of numerical privacy measures to quantify the amount of privacy provided by a privacy-preserving solution based on a linkage attack, as will be described in detail in Chapter 5.

(c) Methodology:

3. **An efficient private blocking technique** (Chapter 6): As discussed in Section 1.1, private blocking techniques are needed to make PPRL applications scalable to large databases by reducing the number of candidate record pairs. There have been several private blocking solutions proposed in the literature in recent times [3, 18, 56, 104, 124, 172] that adapt existing blocking techniques into a privacy-preserving context. Among different blocking techniques the sorted neighborhood approach is the most efficient in terms of number of candidate record pairs generated, as we will discuss in Section 6. However, only limited work has investigated sorted neighborhood-based private blocking [106]. We propose an efficient three-party private blocking technique based on the sorted neighborhood approach [52, 84]. Our approach uses a combination of two privacy techniques which are *k*-anonymous mapping [72] and reference values [154] to adapt the sorted neighborhood approach in a privacy-preserving context. The previously proposed sorted neighborhood-based private blocking approach by Karakasidis et al. [106] uses *k*-nearest neighbor (or *k*-medoids) clustering to group similar (candidate) records into the same block individually by the database owners (which is less efficient than the sorted neighborhood approach in terms of computation complexity), followed by using the sorted neighborhood approach by the third party to group candidate blocks from both database owners. In contrast, our approach only uses the sorted neighborhood approach for the private blocking of databases. This results in more efficient blocking than several other existing private blocking solutions, as we will empirically validate in Section 10.

4. **An efficient two-party private blocking technique** (Chapter 7): Another important shortcoming we identified is that most of the private blocking solutions and private matching and classification solutions in PPRL are three-party approaches that require a trusted third party to perform the blocking and / or linkage. Since three-party solutions are often not suffice in many real-world applications due to the risk of parties colluding, we study how the three-party private blocking solution proposed in Chapter 6 can be converted into a two-party solution by eliminating the need of a third party to perform blocking between databases. Similar as in the three-party solution, we use a combination of privacy techniques which are *k*-anonymous mapping [72] and reference values [154]. As we will empirically evaluate in Chapter 10, our proposed two-party private blocking approach outperforms several other existing private blocking approaches in terms of all three properties of PPRL.

5. **An efficient two-party private matching and classification technique based on reference values** (Chapter 8): We develop efficient two-party private matching and classification algorithms that have low computational burdens and allow high quality approximate matching while still preserving the privacy of the databases that are matched. Low computational burdens can be achieved by using perturbation-based privacy techniques such as public reference values. Reference values have previously been used by Pang et al. [154] as an efficient perturbation-based privacy technique for private matching and classification in PPRL. This approach requires a trusted third party to perform the linkage which might not be available in a real-world application. We propose a novel two-party solution for private matching and classification in PPRL using reference values. Pang et al.'s approach [154] is based on the triangular inequality property of distance metrics on the actual similarity values calculated between private attribute values and public reference values. Our approach, on the other hand, uses the reverse triangular inequality property of distance metrics on the binned similarity values.

6. **An efficient two-party private matching and classification technique based on Bloom filters** (Chapter 9): The second efficient two-party private matching and classification technique we propose is based on Bloom filters. Bloom filters is another efficient perturbation-based privacy technique that has been successfully used for PPRL [59, 174, 175]. However, Bloom filters in a two-party context has so far not been studied. Our approach performs an iterative classification of the Bloom filters by exchanging certain bit positions at each iteration without compromising privacy and complexity. Any Bloom filter encoding methods including those proposed by Schnell et al. [174, 175] and Durham et al. [56, 59] can be used in our approach. In Chapter 10 we will empirically compare different Bloom filter encoding methods in our two-party solution.

(b) Evaluation:

7. **Empirical study** (Chapter 10): Finally, we conduct a comprehensive empirical evaluation of our proposed solutions for PPRL on large real-world and synthetic datasets in terms of the three properties of PPRL. We provide comparative evaluation results of our proposed solutions with several other state-of-the-art techniques [56, 59, 104, 124, 175] using the evaluation framework proposed in Chapter 5.

Parts of this thesis have been published in refereed journals [193, 190] and conferences [188, 189, 191, 192]. In particular, compared to our published survey [193] we cover more recent publications to update the survey and we study the trends in PPRL research over the years, as will be illustrated in Figure 3.1 on Page 32. Compared to our work published in [191, 188, 189, 192], we present an extensive empirical study of our proposed algorithms on several realistic datasets (corrupted with real-world data characteristics using our GeCo tool [183], as will be explained in Section 5.4). We use our evaluation framework based on linkage attack methods we propose on

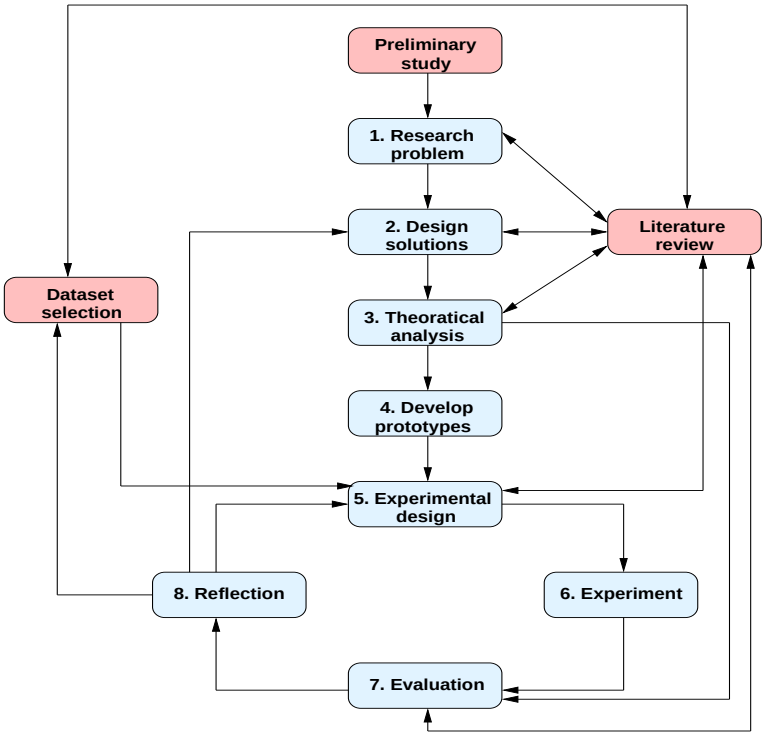


Figure 1.1: The proposed research methodology (adapted from [115]).

those solutions for empirical privacy evaluation. We also include a detailed analysis of the solutions with respect to complexity, linkage quality, and privacy. In addition to the Bloom filter-based two-party private matching and classification solution published in [188], we include several Bloom filter encoding methods and different noise addition techniques.

1.5 Research Methodology

The preliminary approach of this study is to review the literature and understand the basic concepts and current approaches in PPRL research. From this initial understanding, the followed research methodology (adapted from [115]) for the research study is illustrated in Figure 1.1.

1. Define or identify the research problem.
2. Design new algorithms for PPRL that address the research problem.
3. Theoretically analyze the PPRL techniques in terms of the three properties, scalability (based on complexity), linkage quality, and privacy.
4. Prototype the algorithms to be used as proof of concepts (POC) for experimental study.

5. Design experimental study with datasets and evaluation measures to be used.
6. Conduct the experimental study.
7. Validate the theoretical analysis of the solution using experimental evaluation.
8. Reflect the experimental results with regard to experimental design, dataset selection, or algorithm design.

1.6 Thesis Outline

We begin by discussing the preliminaries of record linkage and PPRL in Chapter 2, and we survey existing PPRL techniques in Chapter 3. In Chapter 4 we propose a taxonomy of PPRL techniques, and we characterize existing PPRL techniques along the proposed taxonomy to identify research directions. In Chapter 5 we present an evaluation framework for PPRL techniques that can be used for evaluation and comparison in the following chapters. We then propose a three-party private blocking technique based on the sorted neighborhood approach in Chapter 6 to address the scalability challenge, and convert this three-party solution into a two-party private blocking technique by eliminating the need of a third party in the next chapter. In Chapters 8 and 9 we present efficient two-party private matching and classification algorithms based on reference values and Bloom filters, respectively. In Chapter 10 we compare our proposed algorithms with several existing state-of-the-art techniques using the evaluation framework proposed in Chapter 5. Finally, we conclude the study by summarizing our findings and discussing future research directions in Chapter 11.

1.7 Notation and Terminology

The following table provides the general notation (symbols) and terminology used throughout this thesis. Further notation specific to individual chapters will be introduced at the beginning of the relevant chapters.

Table 1.1: General notation and terminology used in this thesis.

$\mathbf{D}^A$, $\mathbf{D}^B$	Databases held by database owners Alice and Bob, respectively
$\mathbf{D}$, $\mathbf{D}^M$	Original databases, encoded and / or perturbed (masked) databases
$\mathbf{G}$	Global database that contains values in the same domain of database $\mathbf{D}$ which is used for a frequency linkage attack for privacy evaluation
$\mathbf{G}^M$	Masked global database with the same masking function as used in $\mathbf{D}^M$
$\mathbf{R}$	Publicly available reference dataset
RA_i, RB_j	A record in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
A, a	Attributes common to $\mathbf{D}^A$ and $\mathbf{D}^B$ that are used for linking, an attribute $a \in A$
m	Number of attributes used to link records ($m = \|A\|$)
R	A record in the original database $\mathbf{D}$
R^M	A masked record in the masked database $\mathbf{D}^M$
a^M	A masked attribute value in $\mathbf{D}^M$
v, v_i, v_j	An individual attribute value
r, r_i, r_j	A reference value
b^A, b^B	A Bloom filter of each record in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
[]	An empty list
$enc(\cdot, h)$	Function and key used to hash-encode values
$block(\cdot, \cdot)$	Function used to block/index a database
$dist(\cdot, \cdot)$	Distance measure used to calculate distances between two values ($0 \leq dist(\cdot, \cdot) \leq 1$)
$sim(\cdot, \cdot)$	Function used to calculate similarity between two values ($0 \leq sim(\cdot, \cdot) \leq 1$)
s_t	Minimum similarity threshold value to determine a pair of values as similar
n^A, n^B	Number of records in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
n_R, n	Total number of reference values used, and number of records in databases
n_g	Number of global values in $\mathbf{G}^M$ that match a certain masked value in $\mathbf{D}^M$
n_B	Number of blocks in a database generated by a $block(\cdot, \cdot)$ function
q	Number of characters that make a q-gram
$P_s(\cdot)$	Probability of suspicion function of a value
S	Scalability score calculated for scalability evaluation ($0.0 \leq S \leq 1.0$)
LQ	Linkage quality score calculated for linkage quality evaluation ($0.0 \leq LQ \leq 1.0$)
DR	Disclosure risk score calculated for privacy evaluation ($0.0 \leq DR \leq 1.0$)

SNC-3PSize	Three-party private blocking based on sorted neighborhood clustering (SNC) with size-based merging proposed in Chapter 6
SNC-3PSim	Three-party private blocking based on sorted neighborhood clustering (SNC) with similarity-based merging proposed in Chapter 6
SNC-2P	Two-party private blocking based on sorted neighborhood clustering (SNC) proposed in Chapter 7
HCLUST	Private blocking based on hierarchical clustering (HCLUST) proposed by Kuzu et al. [124]
k-NN	Private blocking based on k-nearest neighbor clustering (k-NN) proposed by Karakasidis et al. [104]
HLSH	Private blocking based on hamming-based locality sensitive hashing (HLSH) proposed by Durham [56]
2P-Bin	Two-party reference values-based binning solution for private matching and classification proposed in Chapter 8
2P-BF	Two-party Bloom filter-based private matching and classification solution proposed in Chapter 9
CLK	Cryptographic Longterm Key (CLK) encoding for Bloom filters proposed by Schnell et al. [175]
RBF	Record-based Bloom filter (RBF) encoding proposed by Durham et al. [59]
CLKRBF	Hybrid encoding of CLK and RBF for Bloom filters proposed in Chapter 9

Chapter 2

Background

Building on the introduction to the privacy-preserving record linkage (PPRL) problem in Chapter 1, in this chapter we summarize the background material that contributes to the understanding of basic concepts and techniques of record linkage in general in Section 2.1, and then describe the challenges involved in introducing privacy requirements into the record linkage process in Section 2.2.

2.1 Record Linkage

Record linkage is a general classification problem where record pairs from two different databases (with one record from each database) are classified as 'matches' if the records in pairs refer to the same entity, or as 'non-matches' if they don't [26, 65]. This is a simple SQL-join problem if the databases to be linked contain common unique entity identifiers [32]. However, often unique entity identifiers are not available in all the databases to be linked, and thus common quasi-identifying attributes (QIDs) such as names and addresses need to be used as linkage attributes to identify matching records. The record linkage solutions that can be practically used in real-world applications need to address scalability and linkage quality.

1. **Scalability:** The number of comparisons required for the classification task equals to the product of the size of the two databases. This is a performance bottleneck in the record linkage process since it requires detailed comparison of all the record pairs in the databases using expensive comparison functions [13, 33]. Due to the increasing size of data collections in organizations, comparing all record pairs is not feasible and this would render record linkage solutions impractical in real-world applications. Hence, record linkage applications need to address the requirement of linking very large databases more efficiently.

2. **Linkage quality:** The frequency of typographical errors and other variations in real-world data makes the linkage problem more challenging. The exact matching of identifying attribute values is not sufficient to meet the need of fault-tolerance to real-world data errors. Also record pairs need to be classified accurately by reducing the number of false positives and false negatives

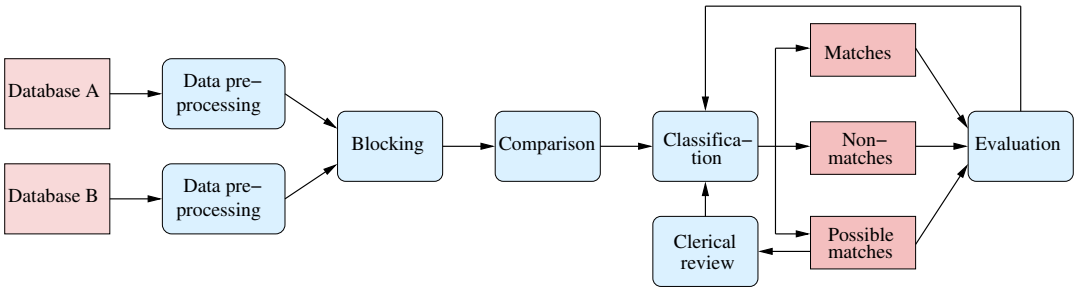


Figure 2.1: Outline of the general record linkage process as discussed in Section 2.1 (taken from [193]).

and thus increasing the accuracy of the classification model used. Hence, techniques that provide both approximate matching for fault-tolerance and effective classification are required for practical record linkage applications.

The record linkage process consists of several steps [30, 63], as Figure 2.1 illustrates. The first step of data pre-processing (data cleaning and standardisation) is crucial for quality record linkage outcomes, because most real-world data contain noisy, incomplete and inconsistent data [12, 162]. This step includes filling in missing data, removing unwanted values, transforming data into well-defined and consistent forms, and resolving inconsistencies in data representations and encodings [38].

The second step in record linkage is blocking [30], which is aimed at reducing the number of comparisons that need to be conducted between records by removing as many record pairs as possible that are unlikely to correspond to matches [13]. Only pairs that are potentially matching, the so called 'candidate record pairs' among which we expect to find matches, are brought together to be compared in detail in the next step, the comparison step. The record pairs that are excluded by a blocking technique are classified as non-matches without being compared explicitly. The process of blocking is discussed further in Section 2.1.1.

Candidate record pairs are compared in detail in the comparison step using a variety of similarity functions [27]. If a linkage is based on using name and address details, for example, then approximate string comparison functions need to be employed which take typographical errors and variations into account [23, 43]. Linkage based on date, age and numerical values needs to employ comparison functions specific to such data [26]. Section 2.1.2 describes several popular comparison techniques in more detail. Several attributes are normally used for comparing records, resulting in a vector that contains the numerical similarity values of all compared attributes.

In the classification step, the similarity vectors of the compared candidate record pairs are given to a decision model which will classify record pairs into matches (where it is assumed the two records in the pair correspond to the same entity), non-matches (where it is assumed the two records in the pair correspond to different entities), and possible matches (where the classification model cannot make a clear decision) [33, 65, 76]. Various classification techniques have been developed for record linkage, and Section 2.1.3 discusses these in more detail.

If record pairs are classified as possible matches, a clerical review process is required where these pairs are manually compared and classified into matches or non-matches [204]. This is usually a time-consuming and error-prone process which depends upon experience of the experts who conduct the review. The manually classified record pairs can also be used as training data for training supervised classification techniques [30]. Alternatively, collective entity resolution techniques [15, 100] can be employed that analyze not only attribute values of records but also relationships between records to determine the match status of pairs or groups of records.

Measuring the complexity, completeness, and quality, of a record linkage study is the final step in the record linkage process before the results of a linkage study can be used in an application, or the linkage approach can be implemented into an operational system. A variety of evaluation measures have been proposed [30, 33]. More details of these measures are provided in Chapter 5.

In what follows we discuss the steps of the record linkage process in more detail, and present techniques that have been used in each of the steps. As we will discuss in Chapter 4, however, many of the state-of-the-art techniques developed for record linkage have not been investigated so far within a privacy-preserving context.

2.1.1 Blocking

If the two database tables $\mathbf{D}^A$ and $\mathbf{D}^B$ which are to be linked contain n^A and n^B records, respectively, then potentially each record from $\mathbf{D}^A$ has to be compared with all records from $\mathbf{D}^B$, resulting in $n^A \times n^B$ comparisons. In large databases, comparing all pairs of records is not feasible. It is also not necessary, because the majority of these comparisons corresponds to non-matching records [30].

To reduce this large number of potential record pair comparisons, some kind of filtering of the unlikely matches can be performed. Techniques that accomplish this are generally known as blocking, searching, or indexing techniques [13, 30]. A single record attribute, or a combination of attributes, commonly called the 'blocking key', is used to decide into which blocks (or clusters) to insert a record. Records that have the same value for the blocking key will be grouped into the same block, and candidate record pairs are generated only from records within the same block. These candidate record pairs are then compared in detail in the comparison step.

Blocking has a trade-off between the computational complexity and the quality of the generated candidate record pairs [13]. Having many small blocks (b) or clusters generated based on a more specific blocking key definition will result in a smaller number of candidate record pairs and thus reduces the computation cost (though communication cost will be increased with many blocks due to the start-up costs). At the same time it is more likely that true matches are being missed. On the other hand, a less specific blocking key definition will lead to larger blocks and more candidate record pairs, but likely also to more true matches that are found [30].

Various blocking techniques for record linkage have been developed in recent years, and several surveys of these techniques have been presented [13, 30, 151]. In the traditional standard blocking approach used since the 1960s [65], all records that

have the same blocking key value will be inserted into the same block, and only the records within the same block will be compared with each other in detail in the comparison step. This reduces the number of comparisons to $(n^A \times n^B)/b$.

An efficient blocking technique is the sorted neighbourhood approach [83, 84], where the database tables are sorted according to a 'sorting key' over which a sliding window of fixed size w is moved. Candidate record pairs are then generated from the records that are within the current window (results in $(n^A + n^B)w$ comparisons). We study this approach in a privacy-preserving context in Chapters 6 and 7.

In mapping-based blocking [96], the blocking key values are mapped to objects in a multi-dimensional Euclidean space whereby the similarities (or distances) between the blocking key values are preserved. A clustering or nearest-neighbour approach is then applied on these multi-dimensional objects to extract candidate record pairs.

To overcome the issues with data that are of low quality, q-gram-based blocking techniques can be used that insert each record into several blocks by generating variations of the record's blocking key value through the use of q-grams (sub-strings of length q characters) [13, 30]. Related to q-gram-based and sorted neighbourhood blocking is suffix array-based blocking [2, 49], where suffixes are generated from the blocking key values, and blocks are extracted from the sorted array of suffix strings.

Canopy clustering [44, 142] generates overlapping clusters (canopies) using two thresholds and an efficient similarity function (usually based on q-grams and Jaccard or TF-IDF/Cosine similarity), such that each record is inserted into several clusters. Each cluster then forms one block from which candidate record pairs are generated.

Locality sensitive hashing (LSH) [113] has recently been used for blocking that allows similar values to be hashed into the same block with high likelihood. Multibit trees [9, 117] is another recent blocking technique that first transforms records into bit vectors and then filters pairs of vectors by using a binary array tree structure.

2.1.2 Comparison

Comparisons between two records can be conducted either at the record level or at the attribute (field) level. Record level comparisons concatenate the attribute values in a record into one long string, and then compare these longs strings between records. With comparisons at the attribute level, comparisons are conducted between individual attribute values, with specialized comparison functions used depending upon the type of data in these attributes.

The comparison of values can either be done exact or approximate. With the former approach, a comparison function simply measures whether the values in two attributes are the same or different. Approximate comparison functions, on the other hand, measure how similar the values in two attributes are with each other. In many real-world record linkage scenarios it is not possible to simply compare two strings exactly because they can contain typographical errors and variations [23, 83].

Approximate matching of values requires a function that represents similarity as a numerical value. Generally, exact agreement is represented as a similarity of 1, total disagreement as a similarity of 0, and partial agreements as similarity values

in-between 0 and 1. Many approximate comparison functions have been developed for different types of data [29, 77, 97, 148, 157]. In the following, popular techniques for approximate string comparison are described in more detail.

The Levenshtein edit distance [148] is a commonly used comparison method for approximate string and sequence matching. It calculates the smallest number of edit operations (character inserts, deletes and substitutes) that are required to convert one string into another. Various modifications and extensions of the basic edit distance approach have been developed. Some allow for different costs of different types of edits, while others allow for gaps, or they are optimized for certain types of data. Two surveys of edit distance-based approximate string comparison functions can be found in [97, 148]. We use distance-based comparison function for PPRL in Chapter 8.

Another type of comparison function is based on the idea of comparing the sub-strings, known as q-grams, that two strings have in common [111, 118, 186]. The strings to be compared are first split into shorter sub-strings of length q characters using a sliding window approach, and then the number of q-grams that occur in both strings is counted. Three different normalized similarity scores can be calculated using the overlap, Dice, or Jaccard coefficient [23, 29]. This is used in Chapter 9.

One string comparison technique that is commonly used in record linkage applications where names and addresses need to be compared is the Jaro-Winkler approach [93, 202]. This technique was developed at the US Bureau of the Census based on the expertise gained in conducting large record linkage projects. The Jaro technique combines an edit distance and a q-gram-based approach [93] by counting the number of common and transposed characters in two strings. Winkler later added several improvements to this basic comparison function [202, 203], such as increased similarity if the beginning of two strings is the same, or weight adjustments based on the lengths of two strings and how many similar characters they contain.

The SoftTF-IDF string comparison technique developed by Cohen et al. [43] aims at comparing strings that contain several words. It can therefore be used for record level comparisons. Similar to the concepts of Term Frequency (TF) and Inverse Document Frequency (IDF) [169], as used in information retrieval, it gives weights to words according to their overall occurrence in a database. The similarity between two strings is calculated as the highest similarity between pairs of words in the strings.

2.1.3 Classification

Assuming k attributes have been compared, the outcome of the comparison step is a vector of similarity values (this is typically called 'comparison vector'), $[s_1, \ldots, s_k]$, for each candidate record pair. These vectors are used to classify record pairs as matches, non-matches, or possible matches, depending upon the decision model used [76]. Record linkage classification techniques can be broadly grouped into four categories: threshold-based, probabilistic, rule-based, and machine learning-based.

Threshold-based classification provides a simple way to classify record pairs based on the calculated overall similarity values of the pairs [25]. The similarity values contained in the comparison vector are summed into a single overall similar-

ity, $S = \sum_{i=1}^{k} s_i$, for each candidate record pair. This similarity value is then used to determine into which class the record pair belongs to based on the threshold values.

A widely used approach to record linkage classification is the probabilistic method developed by Fellegi and Sunter in the 1960s [65]. In this model, the likelihood that two records correspond to a match or non-match is modelled based on a-priori error estimates in the data, as well as frequency distributions of individual attribute values, and the approximate similarities s_i calculated in the comparison step [29]. Extensions to the basic Fellegi and Sunter approach include the use of the Expectation Maximization (EM) algorithm to estimate the conditional probabilities required by the method in an unsupervised fashion [143, 200, 201, 203].

Rule-based classification techniques (also known as deterministic techniques [75]) use sets of rules to classify record pairs [42, 84, 147]. Generating rules is often a time-consuming and complex process, since it requires manual efforts to build and maintain rule systems. An alternative is to learn rules from training data [29].

To accurately classify record pairs, many recently developed classification techniques for record linkage employ supervised machine learning approaches [16, 62, 63]. These supervised approaches require training data with known class labels for matches and non-matches to train the decision model. Once trained, the model can be used to classify the remaining unlabelled pairs of records. Support vector machines and decision trees are two popular supervised learning techniques that have been employed for record linkage [16, 25, 62]. One limitation with supervised learning techniques is, however, that they require training data, which are not always available in record linkage applications, especially in privacy-preserving settings [29].

Alternatively unsupervised learning techniques can be employed, such as clustering, which do not require training data to classify record pairs [147]. Clustering groups record pairs that are similar, such that each cluster consists of records that refer to one real-world entity [44, 142]. Recently developed collective [15, 100], group [153], and graph-based [85, 147] classification techniques, while achieving high linkage quality, are not scalable to very large databases due to their quadratic or higher computational complexity. Active learning [6] is a semi-supervised learning technique that is being used for manual classification required in clerical review.

2.1.4 Evaluation

Evaluating the performance of record linkage algorithms in terms of how efficient and effective they are is the final step in the linkage process. The efficiency of the linkage provides a measure of how scalable a linkage technique is on large real-world applications with potentially millions of records, while the effectiveness of a linkage exercise is measured by the accuracy of the classification model used. A variety of evaluation measures has been proposed that can be used to assess the scalability [30, 33] and quality [33] of the linkage process (as detailed in Chapter 5).

Scalability can be evaluated using measures that are dependent on the computing platform and networking infrastructure used, or measures that are based on the number of candidate record pairs generated. The quality of a linkage can be measured by

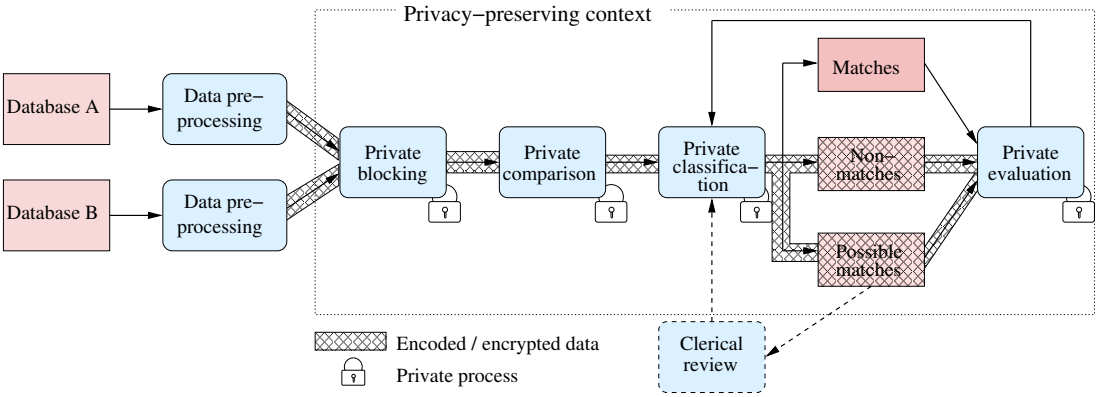


Figure 2.2: Outline of the general privacy-preserving record linkage process as described in Section 2.2 (taken from [193]).

using the metrics commonly employed in both information retrieval, and in machine learning and data mining [140, 161]. Accuracy, precision, recall, and F-measure are such commonly used quality measures. However, accuracy is not a suitable quality measure because record linkage is generally a very imbalanced classification problem with many non-matching record pairs compared to matching pairs [33] that can significantly distort accuracy values. Precision, recall and the F-measure are more suitable for measuring linkage quality [29].

2.2 An Overview of PPRL

As the scenarios in Section 1.2 have shown, the exchange of private or confidential data between organizations is often not feasible due to privacy concerns, legal restrictions, or because of commercial interests. Databases from different organizations therefore need to be linked in such ways that no sensitive information is being revealed to any of the parties involved in a cross-organizational linkage project, and no adversary is able to learn anything about these sensitive data. The increasing need of being able to link databases across organizations while, at the same time, preserving the privacy of the entities stored in these databases, has introduced a new research problem called *privacy-preserving record linkage* (PPRL) [37, 78, 194].

The privacy-preserving requirement in the record linkage process adds a third challenge, privacy, to the two main challenges of scalability and linkage quality that were discussed in Section 2.1. The question now arises how to conduct the steps in the record linkage process (as was shown in Figure 2.1) in a privacy-preserving setting. Privacy needs to be considered in all steps of the record linkage process, making the task of linking databases across organizations more difficult. Figure 2.2 outlines the record linkage process within a privacy-preserving context. Several privacy techniques have been used for PPRL ranging from SMC-based techniques to perturbation techniques such as k-anonymity, reference values, Bloom filters, differential privacy,

and random noise, which we will discuss in detail in Chapter 4. Adversaries in PPRL are assumed to follow different models including the most commonly used honest but curious (HBC) and malicious models which will be described in Chapter 4.

Because data pre-processing can be conducted independently at each data source, it is not part of the techniques that are required for PPRL. However, it is important that all data sources conduct the same data pre-processing steps on the data they will use for linking. Some exchange of information between the data sources about what data pre-processing approaches they use, as well as which attributes they have in common that are to be used for the linkage, is therefore required.

As was discussed in Section 2.1.1, the blocking step is crucial to make record linkage across large databases scalable. This also applies to PPRL, but blocking for PPRL needs to be conducted in such a way that no information that would allow to infer individual records in the databases is revealed to any party or to an external adversary. The scalability challenge of PPRL has been addressed by several recent approaches using private blocking techniques, as we will discuss in Section 3.4.

The attribute values used for comparing records often contain variations and errors, and therefore simply encoding these values with a standard cryptographic technique and comparing the encoded values will not lead to high linkage quality for PPRL [37, 152]. Since a small variation in an attribute value leads to a completely different encoded value [37], only exactly matching attribute values can be identified with such a simple approach. Therefore, an approach for securely and efficiently calculating the approximate matching of attribute values is required. Several of the approximate comparison functions described in Section 2.1.2 have been adapted into a PPRL context as will be discussed in Sections 3.3 and 3.4.

As we discussed in Section 2.1.3, the output of the comparison step are the calculated similarity values for each compared record pair that will be used to classify the pairs into matches, non-matches, or possible matches. In a PPRL context, this classification needs to be conducted in such a way that no party learns anything about the records in the other parties' databases that do not match, such as similarity values for certain attributes of individual record pairs, which record pairs have low similarities, or even the distribution of similarity values across all compared record pairs.

The evaluation of linkage quality in a privacy-preserving context is challenging, because in PPRL access to the actual record values is unlikely to be possible as this would reveal private or confidential information about these records. How to evaluate linkage quality as well as privacy protection is still an open challenge, as we will discuss further in Section 4.3.

2.3 Summary

We have presented the background material to understand the research problem by outlining the process and techniques used in the record linkage process, and the challenges posed when privacy is considered in the process. In the following chapter we will review the literature in PPRL.

Chapter 3

Related Work

A study of related work in privacy-preserving record linkage (PPRL) is reviewed in this chapter. We conducted a survey of around 40 existing PPRL techniques which are then characterized according to the taxonomy (as described in Chapter 4) in Table 4.1 on Page 42 in order to identify future research directions in PPRL.

3.1 Introduction

Research directions for PPRL were provided in [24, 40] stating the needs, problems and current approaches in this area, while various techniques have been developed addressing the research problem [58, 102, 184, 194]. In the following we provide a review of existing PPRL techniques. We highlight important terms in the techniques that relate to our taxonomy we propose in Chapter 4 in *italic* font.

We categorize PPRL techniques into three generations according to the factors that have been considered. These three generations are (1) techniques that consider exact matching of attribute values only; (2) techniques that can conduct approximate matching to improve the quality of linkage; and (3) techniques that also address scalability while conducting approximate matching. We present PPRL techniques under each category in a chronological order to study how the techniques have been developed developed over time. Each technique is given an identifier composed of the first three letters of the first author and the last two digits of the year of publication, which is then used in Table 4.1 to identify individual techniques.

3.2 Exact Matching PPRL Techniques

The first generation of PPRL techniques focus only on the exact matching of records.

Qua98: This is the first approach to PPRL proposed in the *1990s* by Quantin et al. [19, 60, 158, 159] within the framework of *epidemiological* follow-up studies. This approach is applicable for linking *more than two databases* by using a *third party* for conducting the linkage. One-way *secure hash algorithms* (SHA) are used with two pads added in order to avoid dictionary attacks. The comparison is limited to likely pairs of matches by using a *blocking* method based on *phonetic* encodings. Record linkage is then performed using a statistical model (*probabilistic classification*) with weights

estimated by the EM algorithm. An empirical evaluation conducted using *real health datasets* showed high linkage quality of the approach.

Van00: A secure *three-party* approach proposed by Van Eycken et al. [187] in *2000*, is based on creating a single hash pseudonym for maintaining privacy. In this approach, both database owners merge the values of their linkage attributes into a single string (*record-based*) which is then double-hashed using a *secure hash function* and a public key encryption algorithm in order to prevent dictionary attacks. These hash strings are then used by a third party to classify the records using a *deterministic* classification technique. Experiments conducted on *real health datasets* showed that the accuracy of the classification increases if the concatenated string includes the full date of birth value.

OKe04: In *2004*, a *multi-party SMC*-based approach was proposed by O'Keefe et al. [152] for PPRL, as well as privacy-preserving extraction of a cohort of individuals' data from a database, without revealing the identity of these individuals to the database owners. The authors assumed an untrusted *third party*, in that the only way for the third party to obtain identifying information is through *collusion* with a database owner. The approach improves on the security and information leakage characteristics of several previous protocols, including Agrawal et al.'s [1] two-party secure intersection and equi-join protocols that use commutative encryption schemes. However, variations and (typographical) errors in the linkage attributes are not considered (*exact matching*), and the protocol is computationally more expensive than PPRL solutions that use perturbation-based privacy techniques (as will be discussed in Chapter 5).

Fre05: Privacy-preserving information retrieval (PPIR) is a research area related to PPRL, whereby PPIR employs a single query record while PPRL employs all records as match queries. Freedman et al. [69] in *2005* presented an efficient *two-party* privacy-preserving keyword search algorithm for PPIR. The proposed approach assumes both *HBC* and *malicious* adversarial models. Their approach uses *SMC* techniques (homomorphic encryption) and *oblivious pseudo random functions*. The server holds a database of n pairs (x_i, r_i), each consisting of a keyword x_i and its record identifier (payload) r_i. The client's input is a search keyword w. If there is a pair where the keyword x_i is equal to the search keyword w (i.e., *exact matching*), then the corresponding record identifier r_i will be returned to the client.

Lai06: Lai et al. [125] in *2006* proposed a *multi-party* protocol that uses *Bloom filters* for private matching *without a third party* for performing the linkage. In their approach, all the records are first converted into a Bloom filter bit array, and each party partitions its Bloom filter into the number of parties involved in the linkage and sends a segment to the corresponding party. The segments received by a party are combined using a conjunction (logical AND) operation. The resulting combined Bloom filter segments are then exchanged between the parties. Each party checks its own full Bloom filter with the result, and if the membership test is successful then it is considered to be a match. Though the cost of this approach is low since the computation is completely distributed between the parties and the creation and processing of Bloom filters are very fast, the approach can only perform *exact matching*.

Kan08: A *multi-party* approach based on a *generalization* technique (*k*-anonymity) for person-specific *biomedical* data was introduced by Kantarcioglu et al. [101] in *2008*. This approach performs efficient secure joins of encrypted databases by a *third party* without decrypting or inferring the contents of the joined records. It is guaranteed that each record can be linked to no less than *k* entities in the databases. The database owners k-anonymize their databases and send the encrypted databases to the third party. When the third party performs a join, it constructs buckets corresponding to each combination of *k*-anonymous values. For each bucket, the third party performs a *secure equi-join*. This approach is only applicable to categorical data.

Web12: Similar to Van Eycken et al.'s approach, a simple heuristic method for privately linking *medical data* in a *three-party* protocol was presented by Weber et al. [197] in *2012*. The authors experimentally validated the hypothesis that using a concatenated identifier made of the first two characters of the given name and surname attributes along with the date of birth attribute as the linkage attribute provides better results in terms of *sensitivity* and *specificity*, compared to performing the linkage based on the identifier consisting of patients' full names and date of birth. This approach is useful when health policies preclude the full exchange of identifiers that is commonly required by other more sophisticated algorithms.

3.3 Approximate Matching PPRL Techniques

Techniques in the second generation of PPRL techniques look into the approximate matching of attribute values to remedy the problem of errors and variations in real-world data.

Du01: Du et al. [53] in *2001* suggested a secure approach for private remote database access with an untrusted *third party* that is assumed to not collude with any of the two database owners. They propose four different *SMC*-based *e-commerce* models for secure remote database access, all of which require privacy of customer data. The four models are the Private Information Matching (PIM), the PIM from Public Database (PIMPD), the Secure Storage Outsourcing (SSO), and the Secure Storage and Computing Outsourcing (SSCO). *Approximate* record matching is performed using distance functions and Monte Carlo techniques. *Random values* are used to disguise the query and the intermediate results. The minimum value of the final distance values of the records in the database, as compared with the query, is computed to identify the closest match.

Ata03: A *two-party* protocol was proposed by Atallah et al. [7] in *2003* where the *edit distance* algorithm, as presented in Section 2.1.2, is modified for providing privacy to genome sequence *approximate* comparisons in the area of *bioinformatics*. The three types of edit operations are insertions, deletions and substitutions of characters a and b, and each operation has an associated cost, namely $I(a)$, $D(a)$ and $S(a,b)$. The smallest overall cost of transforming one sequence into another is calculated as the edit distance. The dynamic programming matrix M is split across the two parties such that $M = M_A + M_B$. At each step, the minimum of three costs needs to be de-

termined without revealing at which position the minimum occurred. This approach is aimed towards sequence comparisons and has a considerable communication cost. One communication step is required for each element in the matrix M, which is quadratic in the length of the sequences that are compared. It is therefore unsuited for tasks with large databases.

Rav04: In *2004*, Ravikumar et al. [165] used *SMC* techniques for secure computation of several distance functions. In their work, they presented methods for *approximate comparison* of values using string *distance metrics*, specifically TF-IDF [169], SoftTF-IDF [43] and the Euclidean distance. They use a secure stochastic dot product protocol for secure computation of these distance metrics in a *two-party* setting. The use of SMC techniques for achieving privacy makes the protocol computationally intensive. To overcome this drawback, they use *sampling* techniques to control the amount of communication between the two parties. Experiments on the public *Cora bibliographic dataset* [29] showed high linkage quality with average *precision* of 0.85 after $1,000$ samples for vectors of length $10,000$ (0.1%).

Chu04: A token-based *three-party* approach suggested by Churches and Christen [37] in *2004* uses *hash-encoded q-grams* to achieve *approximate* private linkage. Subsets of q-gram sets are used to calculate the *Dice coefficient* between attribute values. All matching hash values are compared by a third party using extra information, such as the number of q-grams contained in a subset and the total number of q-grams comprising an attribute value. A *threshold-based* classification is used for deciding which record pairs are matches. This is a costly approach because of the power set generation of q-gram subsets it requires. Another drawback of this approach is that it is susceptible to *frequency attacks* [184].

Sch09: An approach based on a combination of *Bloom filters* and *q-grams* (to facilitate *approximate matching*) was proposed by Schnell et al. [174] in *2009*. The q-grams of linkage attribute values of each record are mapped to one Bloom filter bit array (*record-based* comparison) using multiple cryptographic *hash functions*. Then the Bloom filters are compared in a bit-wise manner by a *third party*, and similarity between Bloom filters is calculated according to the *Dice-coefficient*, because this similarity function is insensitive to many matching zeros in long Bloom filters. Bloom filters are efficient to generate and compare, and this approach supports approximate matching of values as well, rendering it applicable to real-world conditions. However, due to the use of q-grams this approach is only applicable to matching of *string* attribute values. This approach can be compromised by a *cryptanalysis* attack given the knowledge of certain parameters, as shown by Kuzu et al. [122].

Dur10: Durham et al. [57] in *2010* adapted Schnell et al.'s *Bloom filters*-based approach [174] in their work to evaluate three different PPRL approaches. They investigated (1) *deterministic* classification techniques for *exact comparison*, (2) *probabilistic* classification techniques for *exact comparison*, and (3) *probabilistic* classification techniques for *approximate comparison*. Eleven attributes from a *clinical dataset* from the Vanderbilt University Medical Center were used for this study. The empirical evaluation of these three approaches indicated that approximate comparison using probabilistic classification technique [65] outperformed the other two approaches.

Li11: An approach for privacy-preserving group linkage (PPGL) has been introduced by Li et al. [128] in *2011* to measure the similarity of groups of records rather than individuals. A *threshold-based* PPGL method is proposed to overcome the problem of group membership inference attacks which could be employed to learn the member records of the other party's groups even though the groups are not linked. *K*-combinations of records are first extracted from the groups and then *SMC* techniques are used to privately calculate the set intersection of the *k*-combinations. The *Jaccard* coefficient is used at group level to calculate the similarity between two groups. In order to support *approximate* matching of groups of records, the *Cosine similarity* is employed in a bipartite graph to calculate the similarity of pairs of records between two groups. Both parties only learn the verdict of whether the two groups are matched or not, instead of learning the group similarity value. However, this approach has an exponential complexity in the size of the databases.

Jon13: A group level *anonymous* matching approach for *two-party* PPRL was proposed by Jones et al. [99] in *2013*. In their work only group level data (*generalization*) are revealed between the database owners, not the individual level data. The database owners concatenate the linkage attributes and generate a *hash key* for each record, which is then used to calculate the group ID based on the predetermined number of groups. They repeat this process several times with different salt values added at the end of the concatenated string to generate a different hash key at each iteration and calculate the group ID for each record. Finally, the records are compared based on the group IDs calculated at each iteration. If a record from one dataset is matched with a record from the other dataset in more than a certain number of group IDs, then the records are classified as a match. This method was empirically tested on *real-world data* (to match voter data and Facebook data), and the results showed a high *accuracy* of 95% while generating the same level of uncertainty about individual records as theoretically predicted.

Kum13: An effective *three-party* human-machine hybrid system for PPRL was recently introduced by Kum et al. [121]. Frequent *human interaction* is required in the linkage process to improve the quality of linkage results. In human interactive record linkage, people are involved in fine tuning the false matches as well as examining the uncertain record pairs to make classification decisions [119]. In this work, the authors adapted three privacy techniques which are *decoupling sensitive data, adding random values, and recoding the values* to ensure that no sensitive values are disclosed during any such human interaction. This is the first work in the direction of privacy-preserving interactive record linkage and more research is required for investigating on how much information is needed for tuning the linkage results by an expert without any potential harm that can result from disclosing sensitive information.

3.4 Scalable and Approximate Matching PPRL Techniques

In this section, we survey the third generation of PPRL techniques that address scalability to large databases while allowing the approximate matching of attribute values.

Son00: The approach of Song et al. [178] in *2000* in a *two-party* context takes into consideration the problem of *approximate matching* by calculating *encoded permutations* of values using *pseudo random functions* for private searching of documents by certain query values. The approximate comparison on advanced queries containing multiple words is processed based on individual words (i.e., *field-based*). If an encoded query value matches at least one of the encoded permutations (*rule-based*), then the pair of values can be considered as a match since the permutation occurred due to a typographical error. The use of an *encoded index* data structure-based *blocking* provides an efficient search when the data size is large. However, it is practically impossible to predict all possible permutations by pre-computing all types of errors and variations that might occur in real-world applications. The approach is also susceptible to *frequency attacks* if a certain number of words are being queried.

All05: Al-Lawati et al. [3] proposed a secure *three-party* private blocking protocol in *2005* for achieving high performance private record linkage by using *secure hash-encoding* for computing the *TF-IDF distance* in a secure fashion. In their work, three methods have been explored which are simple blocking, record-aware blocking, and frugal third party blocking. Simple blocking arranges hash signatures in blocks where the similarity of a pair may be computed more than once if they are in more than one common block. Record-aware blocking solves this issue by using an identifier with every hash signature to indicate the record it belongs to. However, these methods provide a trade-off between privacy, and computation and communication costs. The third method, the frugal third party blocking, uses a secure set intersection (SSI) *SMC* protocol to reduce the cost of transferring the whole databases to the third party by first identifying the hash signatures that occur in both databases.

Sca07: Scannapieco et al. [170] in *2007* presented an approach that provides privacy for both data and schema matching without revealing any information. This approach transforms records into objects in an *embedding metric space* using a set of *reference values*, while preserving the distances between record values. These distances are then sent to a *third party* to perform the linkage. To achieve secure schema matching, it is assumed that the third party holds a global schema to which the schemas of the database owners are mapped. A greedy re-sampling heuristic based on the SparseMap [87] algorithm allows the *mapping* of values into a vector space at low computational costs. However, the experimental results presented in [170] indicate the trade-off between a more efficient mapping and the resulting quality.

Ina08: A hybrid approach that combines *generalization* and *cryptographic techniques* to solve the PPRL problem was proposed by Inan et al. [91] in *2008*. This method uses a *blocking* approach based on value generalization hierarchies and the record pairs that cannot be blocked are compared in a computationally expensive *SMC* computation step using cryptographic techniques. This approach manages to perform *approximate matching* both due to the use of the *generalization* scheme in the blocking step, as well as due to the *SMC* step. However, the blocking method is only useful with attributes that can form hierarchies.

Pan09: Pang et al. [154] in *2009* suggested a protocol based on a set of *reference strings* that are available to both the database owners. The database owners com-

pute the *distance* between the reference strings and their attribute values (assumed to be strings), and send the results to a *third party* that sums these distance values and finds the minimum distance. Based on the triangular property of distance-based measures [131], if this minimum distance value lies below a certain *threshold*, then the two original strings are classified as a match. To reduce the size of the matching space, nearest neighbour *clustering* is applied. The performance of the protocol depends crucially on the set of reference strings. Increasing the size of the reference table improves the linkage quality to some extent, but this leads to longer runtime.

Yak09: Based on the work by Scannapieco et al. [170], a similar approach was proposed by Yakout et al. [210] in *2009* which uses Scannapieco's vector representation of attribute values and eliminates the need of a third party for performing PPRL. Complex numbers are calculated to create a *complex plane*, and in the first step the likely matched pairs are computed by moving an adjustable width slab within this complex plane. *Euclidean distance* is used to measure the *approximate similarity* between records. Based on these distances, similar record pairs are classified as those that are within the slab width. These similar pairs are compared in detail in the second step using a *SMC*-based *secure scalar product* protocol based on randomized vectors. This is an improvement over Scannapieco's work in the privacy and scalability aspects.

Ina10: Inan et al. [92] in *2010* presented a hybrid approach for PPRL that combines *differential privacy* and *cryptographic methods* in a *two-party* setting. It uses multi-dimensional *blocking* based on specialized tree data structure (kd-tree, BSP-tree, R*-tree, etc.) to improve scalability. Previous work presented by Inan et al. [91] focused on generalization based on *k*-anonymity to provide a scalable solution, which does not provide sufficient privacy. The work based on differential privacy provides strong privacy guarantees and a trade-off between accuracy, privacy, and scalability [92].

Haw11: Hawashin et al. [81] in *2011* proposed a private *three-party* approach for semantic similarity joins using *long string attributes* (corresponding to *record-based* comparison), such as paper abstracts, product descriptions, and user feedbacks. The two database owners generate their term by long string value matrices, such that each row represents a term (word) and each column represents a long string value, and calculate TF-IDF weights to perform *unsupervised feature selection*. The list of selected features along with some *random features* are sent to a third party that returns the intersection of these two feature lists. The database owners then send the selected feature values of the records with *randomly generated records* to the third party that performs the semantic join operation using methods such as diffusion maps [45], latent semantic indexing [51], and locality preserving projection [82], and classifies the pairs as matches that have a *Cosine similarity* greater than or equal to a minimum *threshold* value. The results of the experimental evaluation showed that the diffusion maps method provided the best performance results in terms of *F-measure* [81].

Moh11: Mohammed et al. [145] in *2011* proposed a *two-party* approach for efficient PPRL using *k*-anonymity-based *generalization*. This work is based on the secure DkA framework proposed by Jiang and Clifton [94] for integrating two private data tables into a *k*-anonymous table. However, the DkA framework is not scalable to large databases. Mohammed et al. presented two scalable methods to securely inte-

grate private data from *multiple data sources* based on the *HBC* and *malicious* adversarial models. The database owners find the global winner candidate with the best score that provides less information to the other party according to some criteria, and then perform a top-down specialization on that candidate for generalizing the databases. The well-known C4.5 classifier is used to recursively *block* (generalized buckets) and classify the records. To prevent *malicious* parties from sending false scores, game-theoretic concepts are used. Empirical studies conducted on *real-world census dataset* [150] showed that this method outperforms DkA in terms of efficiency.

Kar11a: A *three-party* approach to PPRL consisting of a secure *blocking* component based on *phonetic encoding* (Soundex [23]) algorithm and a secure matching component where *approximate matching* is performed using a *distance-based* method is presented by Karakasidis et al. [103] in *2011*. This approach uses a secure version of the Levenshtein *edit distance* [148] function on *Bloom filters* data structure. *Field-based* comparisons between records are conducted and they are classified using a *threshold-based* model. The experimental study conducted on a *synthetic dataset* generated using the *Febrl* [27] tool showed that the approach outperforms the original edit distance algorithm in terms of complexity (due to the secure blocking component) while preserving privacy, and it also offers almost the same matching accuracy.

Kar11b: Karakasidis et al. [105] in *2011* proposed three different faked *random values* injection techniques for phonetic-based PPRL [103]. These techniques are the Uniform Cipher Text/Uniform Plain Text, Uniform Cipher Texts by Swapping Plain Texts, and *k-anonymous* Cipher Texts. In the first method, fake values are added such that both the actual values and the Soundex [23] phonetic values exhibit uniform distributions. This increases the complexity due to massively oversized datasets. The second method overcomes this drawback by modifying the frequency of attribute values such that all Soundex values occur equally frequent. This does not create an excessive number of faked records as with the first method. However, the attribute values that were removed will not participate in the linkage process. The third method aims at creating datasets where each Soundex code reflects at least k attribute values. This work is experimentally evaluated using a *real-world Australian telephone database*. It is stated that in terms of *information gain*, using a Soundex-based fake injection strategy offers adequate privacy for private blocking [105].

Dur12: Durham [56] in *2012* studied the Bloom filter-based approach proposed by Schnell et al. [174, 175] in more detail. In this work the author proposed how *record-level* Bloom filter encoding can be done effectively in order to overcome the problem of cryptanalysis attack associated with field-level Bloom filter encoding [122], and also used *locality sensitive hash* (LSH) functions for private blocking to reduce the computational complexity. A single Bloom filter is used to encode the entire record by using weighted random bits selection from each field-level Bloom filter. A *probabilistic* method based on agreement and disagreement weights is used for classification. Empirical studies conducted on *real datasets* showed that this approach outperforms existing Bloom filter-based approaches.

Bon12: Bonomi et al. [18] in *2012* proposed a new *embedding* strategy for PPRL based on Scannapieco et al.'s [170] approach using *q-grams* and *differential privacy*. In

contrast to using random strings to generate the common base among the parties, this approach uses frequent q-grams mined from their own databases under the differential privacy framework as a base for secure embedding. Frequent q-grams are mined using a top-down approach on a prefix tree. The database owners then map their attribute strings into this common base and send the embedded space to a *third party* that can calculate the *euclidean distance* between vectors in the embedded space to classify the record pairs using a *threshold*. Experiments conducted on *real datasets* showed that the approach achieves better scalability and provides formal privacy proof of differential privacy while resulting in comparable linkage quality.

Kar12: In *2012* a three-party private blocking approach based on k-anonymous (*generalization*) and *reference values* was proposed by Karakasidis et al. [104]. Initially *clusters* are created for a set of reference values that are shared by the database owners using *k-nearest neighbor clustering* such that each cluster consists of at least k elements in the reference set. Each database owner then assigns the blocking key values in their data to the respective clusters according to their *Dice-coefficient* similarity. These clusters are sent to a third party that merges the corresponding clusters to generate candidate record pairs. A main drawback of this approach is that it requires calculation of similarities between each record and all the used reference values.

Kuz13: Recently, Kuzu et al. [124] introduced a private blocking approach based on *hierarchical clustering* and *differential privacy*. Initially global clusters are generated for a set of reference values using *hierarchical clustering*. Then each database owner assigns their records into these global clusters based on their similarity. Differential privacy is used by adding random noise drawn from a Laplace distribution to ensure privacy when these clusters are released. A *three-party SMC* is then used in the second step to compare and classify the candidate record pairs. This approach is computationally expensive in terms of similarity calculations.

Sch13: A recent work by Schnell [172] demonstrated the scalability of their earlier *Bloom filter*-based approach [174] by using a private *blocking* method based on *multibit trees* [117]. Multibit trees work in three steps: First, the Bloom filter vectors in one dataset (optimally the larger dataset) are grouped into bins depending on the number of bits set to 1 in the vectors. A multibit tree is built within each bin in the second step and finally the Bloom filters in the second dataset are searched in the trees based on the *Jaccard* similarity to generate candidate vector pairs. Experiments were conducted on *synthetic* datasets and the results showed the scalability of this approach. Schnell et al.'s Bloom filter-based approach [174] was also empirically evaluated on large real-world hospital admissions datasets (comprising over 26 millions records) combined with a standard blocking method [65] in a latest work by Randall et al. [164] to illustrate the feasibility of this approach on large scale datasets.

Kar13: PPRL in a distributed framework using *LSH* was studied by Karapiperis et al. [107] in *2013*. Data are first encoded into *Bloom filters* and sent to a trusted *third party* that utilizes a map/reduce system in order to distribute the workload efficiently. The third party maps Bloom filters into a low dimensional representation, LSH, based on which Bloom filters are distributed into different reduce tasks. Bloom filters that exhibit the same fragmented LSH minhash keys are routed to the same

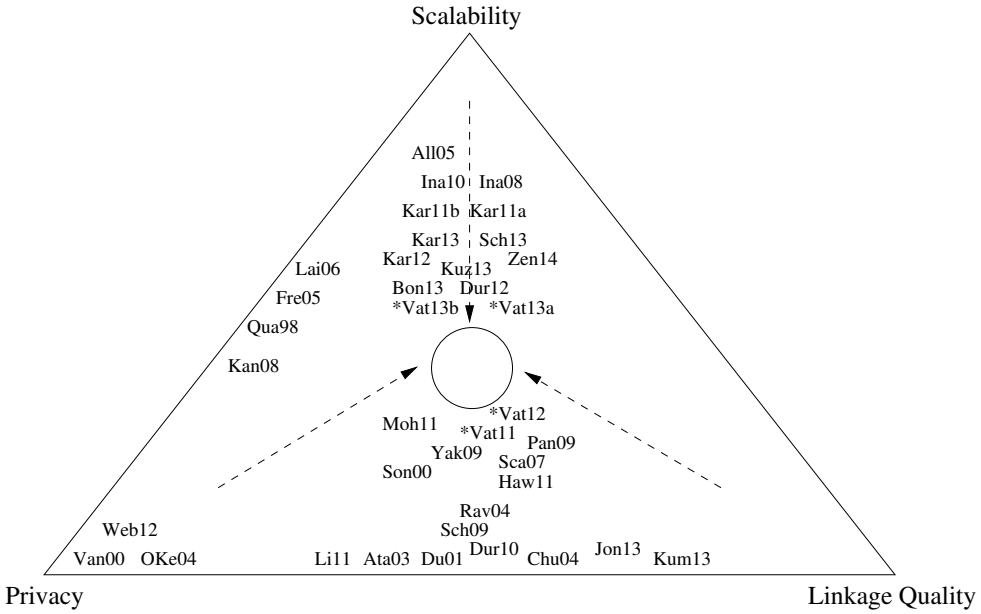


Figure 3.1: Research trends in PPRL. The arrows show the trends towards an optimal solution represented by a circle in the center of the triangle. The techniques that are proposed in this thesis are marked with an asterisk (*).

reduce task. Then pairs are formulated and compared using the *Jaccard distance* metric in order to be classified as matched pairs or not according to a *threshold*. As stated by the authors, optimizing workload distribution by measuring computational cost without overhead in the distributed framework requires further research.

Wen14: The latest work in PPRL by Wen et al. [198] presents efficient *two-party* protocols based on *Oblivious Bloom filter intersection* (OBI) and *private set intersection* protocols for *exact* and *approximate* private linkage. In their work they extended the OBI algorithm by including record identifiers in the garbled Bloom filter to enable the identification of matching records. The second protocol is built on top of this to support approximate matching by incorporating *LSH*. They conducted experiments on *synthetic datasets* and the results showed the efficiency and accuracy of the protocols.

3.5 Summary

In this chapter we have presented a survey of historical and current state-of-the-art techniques for PPRL. Figure 3.1 reflects our view of existing PPRL techniques and the trends in the field of PPRL based on our research in terms of the three main properties of PPRL which are privacy, linkage quality, and scalability. As the figure illustrates, more research in recent times starts focusing towards the center of the triangle by addressing all three properties. We will next characterize the techniques presented in this chapter according to the taxonomy we propose for PPRL and analyze research directions in detail in the next chapter.

Chapter 4

A Taxonomy of Privacy-Preserving Record Linkage Techniques

In this chapter we present a taxonomy of privacy-preserving record linkage (PPRL) that characterizes PPRL techniques along fifteen dimensions. We characterize existing techniques (surveyed in the previous chapter) along this taxonomy and summarize them in Table 4.1. We then highlight shortcomings of current techniques based on this characterization and discuss avenues for research in Section 4.3.

4.1 Introduction

Several surveys on privacy-preserving string matching have been presented in the literature [58, 102, 184, 194]. Trepetin [184] theoretically analyzed four different techniques for anonymized string matching and concluded that many existing techniques fall short in providing a sound solution either because they are not scalable to large databases, or because they are unable to provide both linkage quality and privacy guarantees.

Similar conclusions were also drawn in [102] and [194], which both survey several existing techniques for private matching ranging from classical record matching techniques enhanced by SMC techniques to provide privacy, to advanced solutions developed specific to solve the PPRL problem.

In Durham et al.'s [58] recent survey on privacy-preserving string comparators, six existing comparators that can be used in PPRL for private comparison have been experimentally evaluated in terms of their complexity, correctness, and privacy. The results show the trade-offs of the six surveyed string comparators among the three properties of PPRL.

While all these surveys analyze and compare several private comparison functions, we aim to develop a taxonomy that characterizes all aspects of PPRL, and to provide a comprehensive analysis of current approaches to PPRL along this taxonomy. Illustrating the gaps in current approaches to PPRL will help to identify future research directions for PPRL.

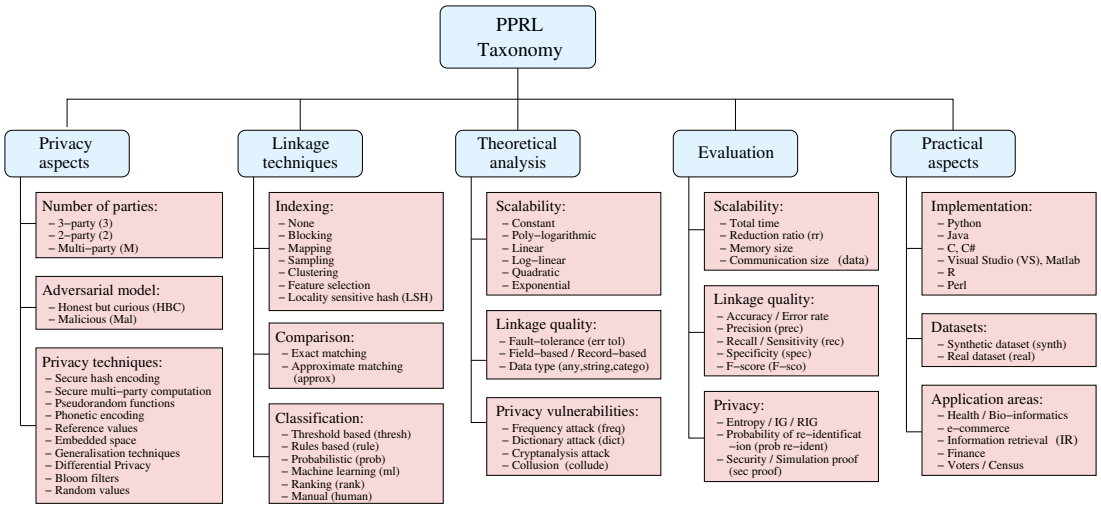


Figure 4.1: The fifteen dimensions used to characterize privacy-preserving record linkage techniques (taken from [193]). Abbreviations shown in brackets are those used in Table 4.1 on page 42.

4.2 A Taxonomy of PPRL Techniques

In this section we describe a taxonomy of PPRL techniques that includes fifteen dimensions of PPRL which we categorize into five main topics, as is illustrated in Figure 4.1. Combined, these fifteen dimensions provide a comprehensive characterization of PPRL techniques. In the following subsections we discuss each dimension in detail, and we provide an overview of the major methodologies or techniques applied in these dimensions.

4.2.1 Privacy Aspects

The privacy requirements for linking databases across organizations consider the assessment of three dimensions of PPRL techniques: how many parties are involved in a cross-organizational linkage, the adversarial model assumed, and the actual techniques employed in a PPRL approach to provide privacy and confidentiality.

4.2.1.1 Number of parties

Solutions to PPRL can be classified into those that require a third party for performing the linkage and those that do not. The former are known as 'three-party protocols' and the latter as 'two-party protocols' [24, 28, 194]. In three-party protocols, a (trusted) third party (which we call the 'linkage unit') is involved in conducting the linkage, while in two-party protocols only the two database owners participate in the PPRL process. The advantage of two-party over three-party protocols is that the former are more secure because there is no possibility of collusion between one

of the database owners and the linkage unit. However, two-party protocols generally require more complex techniques to ensure that the two database owners cannot infer any sensitive information from each other during the linkage process. A further characterization of PPRL techniques is if they can be extended to the efficient linking of data from more than two data sources (multi-party) or not.

4.2.1.2 Adversarial model

PPRL techniques proposed in the literature generally consider one of the two adversarial models that are commonly used in the field of cryptography, and especially in the area of secure multi-party computation (SMC) [73, 78, 135].

1. **Honest-but-curious behavior (HBC)**:
   HBC parties are curious in that they try to find out as much as they can about the other party's inputs while following the protocol [78, 135]. A protocol is secure in the HBC perspective if and only if all parties involved have no new knowledge at the end of the protocol above what they would have learned from the output, which is generally the record pairs classified as matches. Most of the PPRL solutions proposed in the literature assume the HBC adversarial model. Note that this adversarial model does not prevent parties from colluding with each other with the aim to learn about another party's sensitive information [135].

2. **Malicious behavior**: In contrast to HBC parties, malicious parties or adversaries can behave arbitrarily. In particular, malicious parties may refuse to participate in the protocol, not follow the protocol in the specified way, choose arbitrary values for their data inputs, or abort the protocol at any time [134]. Proving privacy under this model for evaluation of a privacy technique is more difficult compared to the HBC model, because there exist additional and potentially unpredictable ways for malicious parties to deviate from the specified steps of the protocol that are undetectable by an outside observer [22, 73, 135].

4.2.1.3 Privacy techniques

A variety of privacy techniques has been employed to facilitate PPRL. The major approaches are:

1. **Secure hash-encoding**: This technique has been one of the first to be used for PPRL [19, 60, 158, 159]. One-way hash-encoding functions [171] convert a string value into a hash-code (for example *'peter'* into *'51dc3dc01ea0'*) such that having access to only a hash-code will make it nearly impossible with current computing technology to learn its original string value. The Message Digest (like MD5) and Secure Hash Algorithms (like SHA-1 and SHA-2) are the most widely known and used one-way hash algorithms [116].

   In order to prevent dictionary attacks, where an adversary hash-encodes values from a large list of common words using existing hash-encoding functions

until a matching hash-code is found, a keyed hash-encoding approach can be used which significantly improves the security of this privacy technique. The Hashed Message Authentication Code (HMAC) function [116] is one such approach. Without knowing the secret key, a dictionary attack will not be successful. However, frequency attacks are still possible, where the frequency distribution of a set of hash-codes is matched with the distribution of known attribute values, such as surnames [136].

A major problem when using hash-encoded values for matching is, however, that only exact matches can be found [60]. Even a single character difference in a string that is encoded will lead to a completely different hash-code.

2. **Secure multi-party computation (SMC)**: The basic idea of SMC is that a computation is secure if at the end of the computation no party knows anything except its own input and the final results of the computed function [41, 73, 135]. Yao [212] first proposed the secure two-party computation problem and developed a secure solution. Goldreich et al. [74] extended this approach to several parties, and they developed a general framework for SMC. SMC employs some form of encryption schemes to allow secure computation. The two major cryptographic encryption schemes used for secure computation in the PPRL literature are commutative [1] and homomorphic [114] encryption. The secure set union, secure set intersection, and secure scalar product, are the most commonly used SMC techniques [41, 171]. A drawback of these SMC techniques is that they are computationally expensive. Several works in recent times have developed efficient SMC techniques for privacy-preserving data mining [4, 5, 64].

3. **Pseudo random functions**: A Pseudo Random Function (PRF) is a deterministic function $f : \{0,1\}^n \rightarrow \{0,1\}^n$ which is efficient (computable in polynomial time) and takes two inputs $x, k \in \{0,1\}^n$. A PRF is a secure algorithm that when given an n-bit seed k, and an n-bit argument x, it returns an n-bit string $f_k(x)$ such that it is infeasible to distinguish $f_k(x)$ for random k from a truly random function [137]. In PPRL, PRFs that have a long period and that are not predictable can be used to generate random secret values to be shared by a group of parties [69, 152, 178].

4. **Phonetic encoding**: A phonetic encoding algorithm, such as Soundex, NYSIIS or Double-Metaphone [23], groups values together that have a similar pronunciation. The main advantage of using a phonetic encoding is that it inherently provides privacy [105], reduces the number of comparisons and thus increases scalability [23], and supports approximate matching by its tolerance against data errors [23, 105]. However, they are language dependent and only a limited work has been done on non-English phonetic encodings [160, 173].

5. **Reference values**: The use of reference values, which are common to all database owners, has been applied in several PPRL approaches [104, 154, 170, 210]. Such reference lists can be constructed either with random faked values, or values

that for example are taken from a public telephone directory, such as all unique surnames and town names. This list of reference values can be used by the database owners to calculate the distances between their attribute values and the reference values. Reference values are used in Chapters 6, 7, and 8.

6. **Embedded space**: Similar to mapping-based blocking as described in Section 2.1.1, this technique embeds the attribute values into a multi-dimensional metric space [18, 170, 210] while preserving the distances between these values. It is often difficult to determine a good dimension of the metric space.

7. **Generalization techniques**: The idea behind data generalization techniques is to overcome the problem of re-identification of individual records by generalizing the data in such a way that re-identification from the perturbed data is not feasible [130, 138, 182]. k-anonymity is one data generalization technique that has been used as an effective privacy technique in PPRL [101, 104, 145]. A database is k-anonymous mapped (satisfies the k-anonymity criteria) if every combination of (masked) quasi-identifier values is shared by at least k records in the database [182]. Different notions of k-anonymous mapping are explored in [72, 129, 180, 208]. Other generalization techniques include value generalization hierarchies [91], top-down specialization [145], and binning [132]. We use k-anonymous mapping in Chapters 6 and 7 and binning in Chapter 8.

8. **Bloom filters**: A Bloom filter is a bit-string data structure of length l bits where all bits are initially set to 0. k independent hash functions, $h_1, h_2, \ldots, h_k$, each with range $1, \ldots, l$, are used to map each of the elements in a set s into the Bloom filter by setting k corresponding bit positions to 1. The Bloom filter was proposed by Bloom [17] for efficiently checking set membership [20]. Bloom filters have been used in PPRL for private matching of records as they provide a means of privacy assurance [56, 57, 103, 125, 174, 198], if effectively used [123]. We propose a Bloom filter-based two-party PPRL solution in Chapter 9.

9. **Noise addition**: Adding noise in the form of extra records to the databases that are linked is a data perturbation technique [108] which can be used to overcome the problem of frequency analysis attacks within PPRL protocols [53, 121]. However, when adding extra records there is generally a trade-off between linkage quality (due to false matches), scalability, and privacy [105]. False matches can also affect the privacy of the matched real values. Recently, **differential privacy** [61] has emerged as an alternative to **randomization noise addition** technique for PPRL. Initially, differential privacy was designed to support interactive queries and aggregate results presentation by adding noise to each statistical query result (such as *Count* or *Sum*) with the magnitude of noise depending on a privacy parameter ϵ and sensitivity of the query set Q. In recent times, it has been adapted to address microdata publication as well as PPRL [18, 92, 124].

4.2.2 Linkage Techniques

The dimensions under this topic cover techniques used in each of the required steps of the PPRL process, as illustrated in Figure 2.2.

4.2.2.1 Indexing / blocking

The techniques employed in the blocking step to facilitate record linkage solutions that scale to very large databases become more challenging if privacy concerns have to be considered. In PPRL, there is a trade-off of the blocking step not only between accuracy and efficiency, but also privacy. Several approaches have been proposed that address the scalability of PPRL solutions by adapting existing blocking techniques, such as standard blocking, mapping-based blocking, clustering, sampling, and locality sensitive hash functions, into a privacy-preserving context, as discussed in Section 3.4.

4.2.2.2 Comparison

Linkage quality is heavily influenced by how the values in records or individual attributes are compared with each other [157]. As discussed in Section 2.2, the naïve approach of exact matching of encrypted values does not provide a practical solution. Several of the approximate comparison functions that were presented in Section 2.1.2 have been investigated from a privacy preservation perspective, as described in detail in Sections 3.3 and 3.4.

4.2.2.3 Classification

The decision model used in PPRL to securely classify the compared record pairs needs to be effective in providing highly accurate results, such that the number of false negatives and false positives is minimized, while at the same time preserving the privacy of all records that are not part of matching pairs. As discussed in Section 2.1.3, a variety of classification techniques has been developed for record linkage. Details of which classification techniques have been used in PPRL were described for individual approaches in Chapter 3.

4.2.3 Theoretical Analysis

Theoretical estimates for the three main properties of PPRL allow the comparison of PPRL techniques, as well as an assessment of their expected scalability to large databases, quality of linkage results, and privacy guarantees.

4.2.3.1 Scalability

This includes the computation and communication complexities that measure the overall computational efforts and cost of communication required in the PPRL process. Generally, the big-O notation is used to specify the computation and commu-

nication complexities [155]. Given n is the number of records in a database, the big-O notation of $O(log\ n)$ represents logarithmic complexity, $O(n)$ linear complexity, $O(n\ log\ n)$ log-linear complexity, $O(n^2)$ quadratic complexity, $O(n^c)$ polynomial complexity, $O(poly\ log\ n)$ polynomial logarithmic complexity, and $O(c^n)$ exponential complexity, where $c > 1$.

4.2.3.2 Linkage quality

The quality of linkage is theoretically analyzed in terms of fault-tolerance of the matching technique to data errors and variations, whether the matching is based on individual fields or whole records, and the types of data the matching technique can be applied to. Fault-tolerance to data errors can be addressed by using approximate matching or pre-processing techniques such as spelling transformations. Records can either be compared as a whole (*record-based*) or by comparing the values of individual selected attributes (*field based*), as was discussed in Section 2.1.2. Approximate comparison functions specific to different types of data are required to link different data types, as was discussed in Section 2.1.

4.2.3.3 Privacy vulnerabilities

The privacy vulnerabilities that a PPRL technique is susceptible to provide a theoretical estimate of the privacy guarantees of that technique. The main privacy vulnerabilities include *frequency attack* and *dictionary attack* (as discussed in Section 4.2.1.3). Bloom filter-based PPRL techniques are generally also susceptible to *cryptanalysis attacks*. As Kuzu et al. [122] recently showed, depending upon the number of hash functions employed and the number of bits in a Bloom filter, using a constrained satisfaction solver allows the iterative mapping of individual hash-encoded values back to their original values.

Another vulnerability associated with three-party and multi-party approaches is *collusion* between parties. Parties involved in a PPRL protocol may work together to find out another party's data. These common privacy vulnerabilities of PPRL techniques are discussed in detail in Chapter 5.

4.2.4 Evaluation

The outcomes of a PPRL technique need to be experimentally / practically evaluated in terms of the three properties: scalability, linkage quality, and privacy.

4.2.4.1 Scalability

The measures that were discussed in Section 2.1.4 (and that will be detailed in Section 5.3.3) can be used to assess the scalability property of PPRL similar to those assessing the scalability of non privacy-preserving record linkage approaches.

4.2.4.2 Linkage quality

Assuming that truth data are available (which is not the case in many PPRL applications), the linkage quality can be assessed using the measures that are used for record linkage in a non privacy-preserving setting as was discussed in Section 2.1.4 (and will be detailed in Section 5.3.2).

4.2.4.3 Privacy evaluation

Various measures have been used to assess the privacy protection that PPRL techniques provide. Here we present the most prominent measures used.

1. **Entropy**, **Information gain (IG)** and **Relative information gain (RIG)**: Entropy measures the amount of information contained in a message [105, 177]. The entropy $H(X)$ and conditional entropy $H(Y|X)$ form the basis for the IG metric [177]. IG assesses the possibility of inferring the original message Y, given its encoded version X [105, 177]: $IG(Y|X) = H(Y) - H(Y|X)$. The lower the value for IG is, the more difficult it is to infer the original value from its encoded value. The RIG measure normalizes the scale of IG ($0.0 \leq RIG(Y|X) \leq 1.0$) with regard to the entropy of the original text Y [105], and is defined as $RIG(Y|X) = \frac{IG(Y|X)}{H(Y)}$. Since RIG values are normalized between 0.0 and 1.0, they provide a marginal scale for comparison and evaluation.
2. **Security / Simulation proof**: The proof of privacy of PPRL solutions can be evaluated by simulating the solutions under different adversarial models [22, 73, 135]. A party's view in the execution of a PPRL technique needs to be simulated given only its input and output to evaluate the privacy in terms of what the party learns from the execution. If under a certain adversarial model (honest-but-curious or malicious, as was discussed in Section 4.2.1.3) a party learns nothing from the execution except its input and output, then the technique can be proven to be secure and private.
3. **Probability of re-identification**: The probability of re-identification of values can be used as a measure to evaluate privacy against several attacks such as frequency attacks, dictionary attacks, and cryptanalysis attacks (as will be discussed in detail in Chapter 5).

4.2.5 Practical Aspects

The final three dimensions cover practical aspects of PPRL techniques including the datasets used for experimental evaluations, how a solution was implemented, and if a proposed solution was developed with a specific application area in mind.

4.2.5.1 Implementation

This dimension specifies the implementation techniques, such as programming languages and computing platforms, that have been used to prototype a PPRL tech-

nique in order to conduct its experimental evaluation. Some solutions proposed in the literature provide only theoretical proofs but they have not been evaluated experimentally, or no details about their implementation have been published.

4.2.5.2 Datasets

Experimental evaluation on one or ideally several datasets is important for the critical evaluation of PPRL techniques. Due to the difficulties of obtaining real-world data that contain personal information, synthetically generated datasets are commonly used. Several tools are available to generate synthetic data [34, 88]. However, to evaluate the practical aspects of PPRL techniques with regard to their expected performance in real-world applications, evaluations should ideally be done on datasets that exhibit real-world properties and error characteristics.

4.2.5.3 Application areas

This dimension describes if a PPRL technique has been developed with a certain application area in mind, or if it is specialized to link data from a certain application area. Some of the areas targeted include healthcare, census, e-commerce, information retrieval (IR), and finance applications.

4.3 Discussion and Research Directions

We conducted an extensive survey of existing PPRL techniques along the proposed taxonomy as summarized in Table 4.1. These PPRL techniques were described in detail in Chapter 3. In this section, we analyze the surveyed PPRL techniques as characterized in Table 4.1 with regard to the taxonomy proposed. This analysis highlights several areas of where future research in PPRL needs to focus on.

As our survey has shown, since the beginning of the development of techniques that aim to provide solutions for PPRL, there has been a large variety of techniques that have been investigated. There is a clear path of progress, starting from early techniques that solve the problem of privacy-preserving exact matching, moving on to techniques that allow approximate matching while keeping the attribute values that are matched private, and finally in the last few years focusing on techniques that address the issue of scalability of PPRL to large databases. This has also been illustrated in Figure 3.1 on Page 32.

The gaps that are identified in existing PPRL research are illustrated in Figure 4.2 which follows the same five topics of our taxonomy shown in Figure 4.1. The research questions that are represented by darker boxes with solid outlines are considered within the scope of this research study and the remaining represented by lighter boxes with dotted outlines are left out for future research which will be discussed further in Chapter 11.

Table 4.1: Characterization of the PPRL techniques surveyed in Chapter 3 (adapted from [193]). The techniques that are proposed in this thesis are marked with an asterisk (*).

PPRL technique	Privacy aspects			Linkage techniques			Theoretical analysis						Evaluation			Practical aspects		
							Scalability		Linkage quality									
	Num of parties	Adversarial model	Techniques	Indexing	Comparison	Classification	comp	comm	err tol	matching	data	Privacy	Scalability	Linkage quality	Privacy	Implementation	Datasets	Application areas
Qua98	3,M	HBC	secure hashing	block	exact	prob	quadratic	linear	√	field	any	freq,collude		rec,spec			real	health
Van00	3	HBC	secure hashing	none	exact	rule	quadratic	linear	×	record	any	freq,collude		accuracy			real	health
OKe04	3,M	HBC	pseudo random,SMC	none	exact	rule	quadratic	quadratic	×	field	any	collude			sec proof			health
Fre05	2	both	pseudo random,SMC	block	exact	rule	linear	polylog	×	field	any				sec proof			IR
Lai06	2,M	HBC	Bloom filter	none	exact	rule	linear	constant	×	record	any	cryptanalysis						
Kan08	3,M	HBC	generalization,SMC	block	exact	rule	quadratic	linear	×	field	catego	freq,collude	rr		sec proof	Java	real	health
Web12	3	HBC	secure hashing	none	exact	rule	quadratic	linear	×	record	any	freq,collude		rec,spec			real	health
Du01	3	HBC	random values,SMC	none	approx	rank	linear	linear	√	field	string	collude			sec proof			e-commerce
Ata03	2	HBC	SMC	none	approx	thresh	quadratic	quadratic	√	field	string				sec proof			health
Rav04	2	HBC	SMC	sample	approx	thresh	linear	linear	√	field	string			prec	sec proof		real	
Chu04	3	HBC	secure hashing	none	approx	thresh	exponential	exponential	√	field	string	freq,collude				Python	synth	health
Sch09	3	HBC	Bloom filter	none	approx	thresh	quadratic	linear	√	record	string	cryptanalysis		prec,rec		Java	both	
Dur10	3	HBC	Bloom filter	none	both	rule,prob	quadratic	linear	√	field	string	cryptanalysis	time	prec,rec			real	health
Li11	2	both	SMC	none	both	thresh	exponential	exponential	√	record	string		time		sec proof	C#	real	
Jon13	2	HBC	generalization	none	approx	thresh	quadratic	linear	√	record	string	freq		accuracy	prob re-ident	R	real	voters
Kum13	3	HBC	random values,recode	none	approx	human	quadratic	linear	√	field	any	freq,collude						health
Son00	2	HBC	pseudo random,SMC	block	approx	rule	linear	linear	√	field	string	freq			sec proof			IR
All05	3	HBC	secure hashing,SMC	block	approx	thresh	quadratic	linear	√	field	string	freq,collude	time	prec		Java	real	
Sca07	3	HBC	embedded space, reference values	mapping	approx	thresh	quadratic	linear	√	field	string	freq,collude	time	prec,rec	sec proof	Java	real	
Ina08	3	HBC	generalization,SMC	block	approx	thresh	quadratic	linear	√	field	catego	freq,collude	rr	rec			real	
Pan09	3	HBC	reference values	cluster	approx	thresh	quadratic	linear	√	field	string	freq,collude	time	prec,rec			real	health
Yak09	2	HBC	embedded space,SMC	mapping	approx	thresh	quadratic	linear	√	field	string		time	prec,rec			real	
Ina10	2	HBC	differential privacy,SMC	block	approx	thresh	quadratic	linear	√	field	catego		rr	rec	sec proof		real	
*Vat11	2	HBC	reference values, generalization	block	approx	thresh	linear	linear	√	field	string	freq	time	prec,rec	prob re-ident	Python	real	
Haw11	3	HBC	random values	feature selection	approx	thresh	quadratic	linear	√	record	string	collude	data time	F-sco		Matlab VS	real	
Moh11	2,M	both	generalization	block	approx	ml	log-linear	linear	√	field	catego	freq	time	err rate	sec proof		real	finance
Kar11a	3	HBC	Phonetic	block	approx	thresh	quadratic	linear	√	field	string	freq,collude	time	prec,rec	IG,RIG	Java	both	
Kar11b	3	HBC	Phonetic, random values	block	approx	thresh	quadratic	linear	√	field	string	collude	time	acc	IG,RIG	Java	both	
Dur12	3	HBC	Bloom filter	LSH	approx	prob	quadratic	linear	√	record	string	collude	time	prec,rec	prob re-ident	Perl	real	health
*Vat12	2	HBC	Bloom filter	block	approx	thresh	quadratic	linear	√	record	string	cryptanalysis	time,rr,memory	rec	prob re-ident	Python	real	
Kar12	3	HBC	generalization, reference values	cluster	approx	thresh	quadratic	linear	√	field	string	freq,collude	time	prec,rec			both	
Bon12	3	HBC	embedding space,diff erential privacy	mapping	approx	thresh	quadratic	linear	√	field	string	freq,collude	time	F-sco	sec proof		real	
Kuz13	3	HBC	differential privacy,SMC	cluster	approx	thresh	quadratic	linear	√	field	any	collude	time,rr,data	prec,rec,pc	sec proof		real	
Sch13	3	HBC	Bloom filter	block	approx	thresh	linear	linear	√	record	string	cryptanalysis	time			Java	synth	
*Vat13a	3	HBC	reference values, generalization	cluster	approx	thresh	linear	linear	√	record	string	freq,collude	time,rr	pc	DR	Python	both	
*Vat13b	2	HBC	reference values, generalization	window	approx	thresh	linear	linear	√	record	string	freq	time,rr	pc	DR	Python	both	
Kar13	3	HBC	Bloom filters	LSH	approx	thresh	quadratic	linear	√	record	string	cryptanalysis	time,rr	prec,rec		Java	real	
Wen14	2	HBC	Bloom filter,SMC	LSH	both	thresh	linear	linear	√	record	string	cryptanalysis	time	prec,rec	sec proof	C	synth	

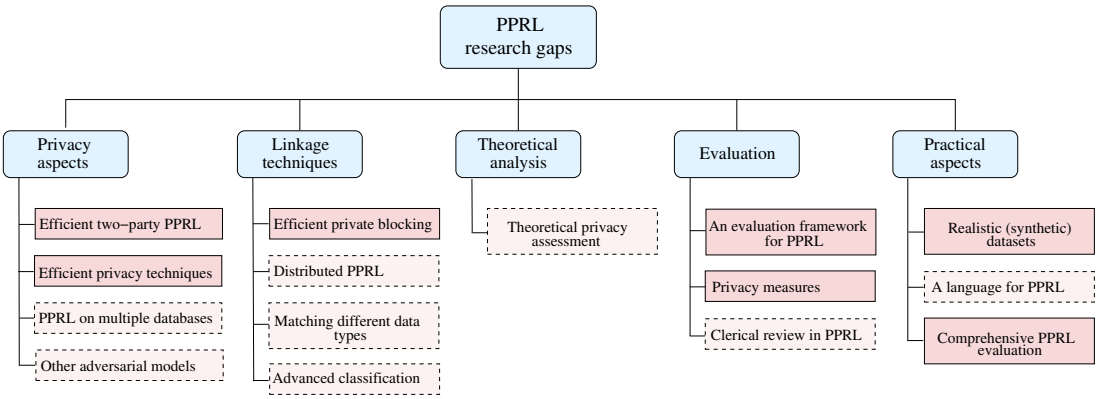


Figure 4.2: The gaps identified in existing PPRL research to outline research directions. The research gaps represented by darker boxes with solid lines are addressed in this thesis, while the lighter boxes with dotted lines are left for future work.

4.3.1 Privacy Aspects

With regard to privacy, several topics require further attention in order to make PPRL more applicable for practical applications.

- **Efficient two-party PPRL**: Most work in PPRL require a (trusted) third party to perform linkage which is not always available in real-world applications. The two-party solutions proposed in the literature often use expensive SMC privacy techniques to ensure that no private information can be inferred from the data exchanged between the two database owners, which are not practical due to computational cost. Therefore, two-party PPRL solutions that employ efficient privacy techniques are required for practical PPRL applications where no third party is available. We address this problem in Chapters 7, 8 and 9.

- **Efficient privacy techniques**: As the characterization of PPRL techniques in Table 4.1 has shown, many different privacy techniques have been explored over the past nearly two decades to address the various challenges posed by the requirements of PPRL. More advanced privacy techniques have been developed in the second and third generations while first generation techniques are mainly based on secure hash-encoding only. More research is needed to investigate the use of efficient and other advanced scalable privacy techniques for private blocking and private matching and classification in PPRL that provide sufficient privacy protection to work in combination with or even replace the expensive SMC-based techniques. This is addressed in our work by using efficient perturbation-based privacy techniques such as reference values [154], Bloom filters [17], and k-anonymous mapping [72] in Chapters 6, 7, 8, and 9.

- **PPRL on multiple databases**: Most work in PPRL (and record linkage in general) thus far has concentrated on linking data from two database owners

only. Only a small number of approaches have investigated how to efficiently link databases from more than two data sources [101, 125, 145, 152, 158]. As the scenarios in Section 1.2 have shown, however, linking data from more than two sources is commonly required. Recent work by Sadinle et al. [168] extends the Fellegi and Sunter model to link more than two databases in a non privacy-preserving context.

- **PPRL for other adversarial models**: Most solutions proposed so far assume the HBC adversarial model. However, this is not sufficient in many real-world applications. PPRL under different adversarial models such as the covert model [8] and accountable computing [95] need to be developed.

4.3.2 Linkage Techniques

Research in non-PPRL in recent years has developed various advanced techniques that provide improved scalability and linkage quality. Thus far, however, most of these techniques have not been investigated in a privacy-preserving setting.

- **Efficient blocking**: Most work in PPRL that has investigated scalability through some form of blocking (or indexing) technique has employed the basic standard blocking approach [65]. As explained in Section 2.1.1, this technique is not efficient and has quadratic complexity when the databases are large. Mapping-based blocking [96] is a second technique that has been employed in PPRL [170, 210]. The use of locality sensitive hashing (LSH) has recently been proposed to improve the scalability of PPRL techniques [56]. Other efficient blocking techniques such as the sorted neighbourhood, or suffix-array-based techniques, need to be explored in a privacy-preserving setting. Chapters 6 and 7 contribute to this problem by using the sorted neighborhood approach.

- **Distributed PPRL**: Distributing computations in PPRL among computational resources can scale up the process with respect to large scale data volumes. Distributed computing in PPRL needs to consider the privacy aspect, making the task challenging. There has not been much work done in this direction. Karapiperis and Verykios [107] proposed a distributed framework for PPRL based on LSH. More work needs to be carried out addressing this aspect of scalability to develop and implement practical PPRL solutions.

- **Matching different data types**: PPRL solutions in the second and third generations consider approximate comparison, mostly for string data type only. Research is required to develop approximate comparison functions that are tailored to numeric, date, age, and time attributes, and even for those containing geographic and other complex types of information [29].

- **Advanced classification**: As Table 4.1 shows, most current approaches to PPRL employ a simple threshold or rule-based deterministic approach to classify the compared record pairs. Only limited work has been conducted that investigates

the application of advanced classification techniques that have been developed for record linkage in the past decade, such as machine learning or graph-based collective classification approaches, in a privacy-preserving context [145, 209, 213]. This constitutes a significant gap between the state-of-the-art techniques in non-PPRL techniques and those employed in PPRL, and provides ample opportunities for research to significantly improve PPRL techniques.

4.3.3 Theoretical Analysis

Analyzing PPRL solutions in terms of privacy, linkage quality, and scalability in order to understand the expected performances of the solutions with regard to these three properties is important and has some theoretical challenges unaddressed.

- **Theoretical privacy assessment**: While the analysis of scalability of PPRL algorithms with regard to their communication and computation requirements is based on standard approaches such as the big-*O* notation [155], and the analysis of linkage quality can be assessed by the type of data that can be matched, and if matching is exactly or approximately, the theoretical assessment of the privacy achieved within PPRL is currently the least matured aspect.

4.3.4 Evaluation

The evaluation of implementations of PPRL techniques with regard to their scalability, linkage quality, and privacy preservation, poses some unique challenges.

- **An evaluation framework for PPRL**: There is currently no framework available for PPRL that facilitates the comparative evaluation of different PPRL techniques with regard to privacy, scalability, and linkage quality. Researchers have used a variety of evaluation measures and datasets (both real and synthetic), which makes comparing existing techniques difficult. We address this gap by proposing an evaluation framework for PPRL solutions in Chapter 5.

- **Privacy measures**: While the two properties of scalability and linkage quality have standard sets of measures that have been widely used for evaluation, privacy does not have such a standard set of measures to allow for comparative evaluation. A standard set of privacy measures is required that quantifies the amount of privacy provided by a privacy-preserving solution. We propose a set of measures for empirical privacy evaluation of PPRL solutions in Chapter 5.

- **Clerical review in PPRL**: Current PPRL techniques only address how to assess linkage quality and completeness to a very limited degree. Given in a practical linkage situation the true match status of the compared record pairs are unlikely to be known, and in a PPRL scenario even the actual record attribute values cannot be inspected (because this would reveal private information), measuring the linkage quality and completeness is difficult [11, 33]. Hence, research directions are required for privacy-preserving interactive record linkage

through active learning systems or crowdsourced systems [6, 141, 196]. Recent work in PPRL proposes an interactive solution with human-machine interaction to improve the quality of linkage results [121].

4.3.5 Practical Aspects

Several gaps exist in the PPRL literature regarding the practical aspects.

- **Realistic (synthetic) datasets**: Since real-world datasets are often not available and/or accessible due to privacy and confidentiality concerns, researchers are often dependent on synthetic datasets for evaluating their algorithms. While there have been several tools developed that allow the generation of datasets, a practical way of generating datasets that exhibit similar characteristics as real data (such as data errors, variations, and dependencies between attributes) is essential to conduct an effective empirical evaluation. We contribute to this direction by developing a flexible data generation and corruption tool [35, 183] to generate and/or corrupt synthetic datasets with realistic data characteristics, which will be discussed in Chapter 5.
- **A language for PPRL**: Researchers have used various languages to prototype their algorithms for evaluation, making it difficult to compare different algorithms. A language for PPRL will need to facilitate the detailed specifications of all building blocks of the PPRL process in the form of abstract representations, such as XML schemas. This will make it possible for researchers to implement their novel algorithms and techniques, and integrate them so as to evaluate them comparatively.
- **Comprehensive PPRL evaluation**: So far it seems that no single PPRL technique has outperformed all other techniques in the three properties of linkage quality, privacy preservation, and scalability to large datasets. However, the lack of comprehensive studies that compare existing techniques within the same framework and on different types of data, means that it is currently not possible to determine which technique(s) perform better than others on data with different characteristics and of different sizes. We contribute to this research avenue by conducting a comprehensive evaluation of several PPRL solutions in terms of all three properties of PPRL in Chapter 10.

4.4 Summary

We have identified fifteen dimensions that allowed us to characterize PPRL techniques, and to generate a taxonomy of such techniques. This proposed taxonomy can be used as a comparison and analysis tool for PPRL techniques. Through this taxonomy we have identified various shortcomings of current approaches to PPRL that suggest several future research directions in this field. We will address some of these identified shortcomings in the remaining chapters of this thesis.

Chapter 5

Evaluation Framework

As the research directions identified in the previous chapter (Section 4.3) stated, (1) developing an evaluation framework for privacy-preserving record linkage (PPRL) with a standard set of measures for assessing the three main properties of scalability, linkage quality, and privacy, and (2) using realistic datasets are the two important research problems in order to practically evaluate and implement PPRL solutions in real-world applications. Recognizing this, we propose an evaluation model in Section 5.2 based on which we define evaluation measures for the three properties in Section 5.3. We present details of the datasets in Section 5.4, the linkage studies in Section 5.5, and the computing platform in Section 5.6 that will be used to evaluate our proposed algorithms.

5.1 Introduction

Over the years, various solutions for PPRL have been proposed as reviewed in Chapter 3. Privacy is addressed in these solutions using two different types of general approaches: (1) secure multi-party computation (SMC) techniques [41, 78, 135] and (2) data perturbation techniques [108, 109, 199]. The former approach is generally more expensive with regard to the computation and communication complexity though it provides strong privacy guarantees, while the latter uses efficient techniques and, as opposed to SMC techniques, in many cases it reveals a certain amount of information without compromising the privacy of sensitive data. However, due to the presence of partially revealed information, such perturbation techniques can be vulnerable to various types of attack.

It is important to note that the objective of PPRL is different from that of privacy-preserving data publishing [70, 179] or of statistical data disclosure [54, 90]. Privacy-preserving data publishing masks a dataset in such a way that no identifying information about individuals can be inferred from the published dataset, while PPRL aims to identify matching records in two or more datasets without disclosing any sensitive information that can be used to identify individual records (and thus the entities they refer to) in the datasets. Therefore, in data publishing sensitive attributes which may contain some (masked) sensitive values (for example, an attribute containing disease values) are disclosed possibly along with the (masked) quasi-identifiers

(QIDs) that contain personal identifying information such as names and addresses. In PPRL, on the other hand, only the (masked) QIDs are disclosed (only to the parties involved in the process) to allow the identification of matching records.

Various privacy models have been used for data publishing and different attacks have been studied in privacy preserving data publishing, including minimality attacks [207], deFinetti's theorem [112], and composition attacks [71]. However, most of these attacks are not applicable to PPRL since they use information from the (masked) sensitive attributes as well. Without sensitive attribute values disclosure, such attacks would not be possible.

Several attack methods have been developed to investigate the privacy guarantees of perturbation-based PPRL solutions. The main attacks and vulnerabilities of PPRL defined in the literature include:

- **Dictionary attack:** In dictionary attacks, it is assumed that the adversary knows the masking function (e.g. one-way hash function such as SHA and MD5 [171]) and potential parameter values used in a PPRL protocol, so that the adversary can mask a large list of common (global) values using the same masking function and parameter values as used in the PPRL protocol until a matching masked value is found. A keyed masking approach can overcome this problem by using a secret key for masking [116]. The Hashed Message Authentication Code (HMAC) function [116] is one such approach. Without knowing the secret key, a dictionary attack is unlikely to be successful.

- **Frequency attack:** Frequency attacks are still possible on the keyed masking approach (without knowing the secret key), where the frequency distribution of a set of masked values matches with the distribution of known global values [136].

  As is examined in [126], original values can be identified in a pseudo-anonymiza tion-based PPRL solution by using frequency distribution analysis of anonymized values. Experimental results showed that exact identification, without any prior knowledge, is difficult but the characteristics of the dataset and the quality of prior knowledge influence the likelihood of

- **Cryptanalysis attack:** Generally, Bloom filter-based PPRL techniques [56, 174, 188] are also susceptible to cryptanalysis attacks [122], where the bit distribution in a Bloom filter allows an adversary to learn the characteristics of hash functions that are used to map the values from records (e.g. q-grams) into the Bloom filter. This is similar to a frequency attack on bits and consequently on the values or q-grams which are mapped to those bit positions.

  Kuzu et al. [122] proposed a constraint satisfaction cryptanalysis attack on Bloom filters in PPRL, where the Bloom filter encodings can be iteratively mapped to values in a global dataset, and certain values can then be identified using the properties of the hash functions and the frequency distribution of values.

- **Composition attack:** Given auxiliary information (also called background knowledge [71]) about the individual datasets that are linked and / or certain records in the datasets, a composition attack can be successful by combining knowledge from more than one independent masked datasets to learn sensitive values of certain records [71].

  An attack on distance preserving perturbation techniques is investigated in [185] where the original data values can be re-identified with high level of confidence if knowledge about mutual distances between data objects is available.

- **Collusion:** Another vulnerability associated with three-party and multi-party solutions is the collusion between some of the parties involved in the protocol (one of several or a set of database owners and the third party) with the aim to learn the other database owner's data. Different types of scenarios might occur with regard to collusion, as they will be discussed in detail in Section 5.2.

Linkage studies or linkage attacks defined in the statistical disclosure community [54] are general terms for attack methods, that link a masked dataset to an external global dataset with known values using any subset of the previously discussed attacks in order to re-identify records and / or attribute values (known as identity or attribute disclosure, respectively, as will be explained in Section 5.3.1) in the masked dataset. We consider linkage attack methods based on frequency attack, cryptanalysis attack, and collusion for privacy evaluation of PPRL solutions.

Based on such re-identification attacks, PPRL solutions can be evaluated for privacy guarantees. However, as characterized in Chapter 4, most of the PPRL solutions developed so far have not been properly evaluated in terms of the privacy aspect. Some PPRL solutions provide theoretical proofs of the privacy aspect which makes the comparative practical evaluation of solutions difficult. As Rudin and Wagstaff explained in [167], in the communities of machine learning, data mining, and statistics, relatively little effort is made on practical evaluation and deployment of novel algorithms, and these communities should prioritize applications of algorithms that have impact to science and society. The same holds for the area of PPRL as well.

A general framework with a set of standard and normalized measures is therefore required to conduct such practical evaluation and comparison of PPRL solutions with respect to the three main properties of PPRL, scalability, quality, and privacy. Cormode et al. [46] recently proposed a unifying framework for evaluating empirical privacy and empirical utility of several privacy techniques by using the measures of prediction accuracy and relative query results, respectively. However, the prediction accuracy measure used in this work provides only the average empirical privacy based on the accuracy of the classifier that aims to find correlations between each quasi identifier and the sensitive attribute (which is also not available in PPRL as discussed above). Given this lack of standard privacy measures and evaluation methods for PPRL, we therefore propose a comprehensive evaluation framework that includes a wide range of numerical measures for empirical evaluation of all three properties, and that enables quantifying and interpreting the performances of different PPRL solutions on the same scale.

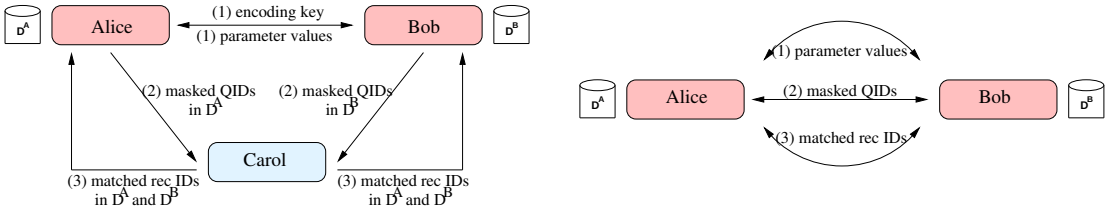


Figure 5.1: General three-party (left) and two-party (right) settings in PPRL solutions with linkage databases $\mathbf{D}^A$ and $\mathbf{D}^B$ and the data flow between the parties (taken from [190]). The numbers correspond to the order of the data flow in the protocols.

5.2 Evaluation Model

In this work, we consider an evaluation model for PPRL on two data sources only. Assume *Alice* and *Bob* are two database owners with their respective databases $\mathbf{D}^A$ and $\mathbf{D}^B$ (generally referred as $\mathbf{D}$), who participate in a PPRL protocol to identify matching records in their databases that correspond to the same real-world entities under the privacy-preserving setting. Existing PPRL techniques can be categorized based on their need (or not) of a third party for performing record linkage [24, 28, 194]. General settings of three-party and two-party protocols are illustrated in Figure 5.1. In three-party protocols, a trusted third party, *Carol*, is involved in conducting the linkage, while in two-party protocols only the two database owners participate in the PPRL process. As was discussed in Section 5.1, three-party protocols are often not sufficient in many real-world applications due to the absence of a trusted third party, since there is a risk of collusion between one of the database owners and the third party with the aim to learn the other database owner's sensitive data. Two-party protocols do not rely on a trusted third party but they generally require more complex techniques to ensure that the two database owners cannot infer any sensitive information from each other during the linkage process.

The internal adversaries in a PPRL protocol are the parties involved in the process (*Alice*, *Bob*, and / or *Carol*). We assume that the parties involved follow the honest but curious behavior (HBC) [78, 135], in that they try to find out as much as possible about the data of the other parties while following the protocol. So far most developed PPRL techniques adapt the HBC threat model, as surveyed in Chapter 4. It is important to note that the HBC threat model does not prevent collusion between parties [135]. There have been few PPRL techniques proposed for the malicious threat model [135] as well, where adversaries may behave arbitrarily. Proving privacy under the malicious model is more difficult because there exist several and potentially unpredictable ways for malicious parties to deviate from the protocol [22, 73, 135].

Two different general philosophies are adopted to preserve privacy and confidentiality of person-level data, which are restricted access and restricted data [54, 67]. To obtain effective results of privacy-preserving tasks, it is often preferred to have uncontrolled access to restricted data rather than restricted access to data [67]. Often, restricted data is achieved in PPRL by first decoupling personal (quasi-)identifying

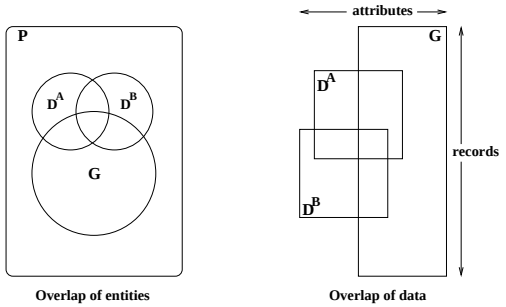


Figure 5.2: Overlaps of entities (left) and data (right) in the linkage databases $\mathbf{D}^A$ and $\mathbf{D}^B$ and the global dataset $\mathbf{G}$ (taken from [190]). $\mathbf{P}$ is the total assumed population.

attributes (QIDs) from sensitive attributes [121] and then by transforming a database ($\mathbf{D}$) into a masked version ($\mathbf{D}^M$) in order to protect the actual sensitive values in the database while preserving certain information to perform effective linkage.

Privacy evaluation requires assessing the risk of disclosure by calculating the probability that an adversary can correctly identify a value in a released dataset [54]. Such re-identification studies can be done through a linkage attack, as described in Section 5.1, using an available dataset, for example a publicly available global dataset such as a telephone book or an electoral roll. In this thesis we assume the adversary is using a linkage attack for evaluating the privacy of PPRL solutions.

We assume that the adversary has access to a global dataset $\mathbf{G}$ that contains $N = |\mathbf{G}|$ unique values or combinations of values (for example, combinations of surname and first name values) of the population $\mathbf{P}$ from which the databases $\mathbf{D}^A$ and $\mathbf{D}^B$ are also drawn. This is reasonable because generally personal identifying attributes, such as names and addresses, are used for linkage and in many countries this background information is partially available in public resources (e.g. North Carolina (NC) voter registration data [31]). The individual databases that are used for the linkage ($\mathbf{D}^A$ and $\mathbf{D}^B$) can be considered as horizontal partitions of $\mathbf{G}$ (i.e. records overlap), while $\mathbf{G}$ can be a vertical partition of the linkage databases (attributes overlap). An overview of the overlaps of records and attributes in the datasets $\mathbf{G}$, $\mathbf{D}^A$, and $\mathbf{D}^B$ is illustrated in Figure 5.2.

We only consider insider attacks (which involve the internal adversaries who are the database owners and / or the third party) for privacy evaluation. We deem insider attacks to be the worst case because an insider adversary can be assumed to have more information than any external adversary, including knowledge about the PPRL protocol used, masking methods, and parameter values of the linkage techniques and algorithms used. It is important to note that a frequency attack might still be possible by an external adversary without this information. The possible scenarios for insider attacks in three-party and two-party protocols are:

- **Three-party protocols**

    1. In the first scenario, we assume that *Alice*, *Bob* and *Carol* do not collude with each other. This case is much harder to attack because *Carol* does not

know the encoding key and / or the parameter values used in the protocol and *Alice* and *Bob* do not have access to the actual or masked values in each other's database. In this case, only a frequency attack might be possible by *Carol* depending on the PPRL protocol used.

2. In the second scenario, one of the database owners (*Alice* or *Bob*) gets the other database owner's data (*Bob*'s or *Alice*'s, respectively) by colluding with the third party *Carol*. This is a worst case assumption because if two parties collude in such a way, then the privacy of the party that is not involved in the collusion cannot be assured. However, many three-party protocols assume a trusted third party (as reviewed in Chapter 3) to reduce this risk of collusion. An alternative is to re-design a three-party protocol into a two-party protocol, which is one of the important research aims of this thesis.
3. Similar to the above scenario, *Carol* colludes with *Alice* or *Bob* in order to get the (secret) encoding key. Thereby it can conduct a dictionary attack using the key, and so can decode both *Alice*'s and *Bob*'s data. Instead of one of the database owners, the third party gets both database owners' data in this type of collusion. However, the colluding database owner in many cases would not like to reveal the (secret) encoding key because that would compromise the privacy of its own data as well.
4. The first scenario, where no collusion between parties happens, is the best possible assumption. However, collusion can still happen in a HBC protocol [135]. The second and third scenarios are the worst case assumptions and they may be too unrealistic. Therefore, in this fourth scenario we assume that *Carol* knows only the masking function(s) and the parameter values used (and not the encoding key), either by colluding with *Alice* or *Bob*, or assuming or estimating parameter values with some background knowledge. *Carol* can perform an attack depending on the protocol, for example a cryptanalysis attack [122], with this knowledge to infer *Bob*'s or *Alice*'s values.

- **Two-party protocols**

1. No collusion is obviously possible in two-party protocols. However, similar to the fourth scenario in three-party protocols which was described above, *Alice* and *Bob* know the masking function(s) and the parameter values used in the protocol, and as a result they can perform attacks on the exchanged (masked) data between them to infer actual values from each other's data.

In the remainder of this thesis, we assume that *Carol* knows the masking function(s) used in a PPRL protocol and knows or predicts the parameter values used in the protocol (fourth scenario for three-party protocols) to evaluate privacy of three-party protocols, similar to any two-party protocols. This assumption of an adver-

sary's background knowledge (or partial knowledge) has been used in many attack methods that have been proposed in the literature [71, 122, 126, 185].

5.3 Evaluation Measures

The evaluation of a PPRL technique needs to be conducted in terms of the three properties of privacy, quality, and scalability. Quality and scalability correspond to the effectiveness and efficiency of a linkage process and they can be assessed based on available standard measures (such as precision, recall, reduction ratio, etc.) that will be discussed in Sections 5.3.2 and 5.3.3, respectively. However, the privacy protection provided by a PPRL technique is comparatively more difficult to assess. In the following Section 5.3.1, we present evaluation measures that can be used to evaluate the privacy aspect of PPRL. While the privacy measures based on information gain (see Section 5.3.1.1) have previously been used in PPRL [56, 105], the statistical disclosure risk measures based on probability of suspicion are novel.

5.3.1 Privacy Measures

Privacy is normally measured as the risk of disclosure of information to the parties involved in a PPRL protocol. As defined in the glossary on statistical disclosure control [89], if an entity's confidential information can be identified in the disclosed (masked) data with an unacceptably narrow estimation, or if it can be exactly identified with a high level of confidence, then this raises a privacy risk of disclosure. A practical way of assessing disclosure risk is to conduct re-identification studies by linking values from a masked dataset to an external global dataset $\mathbf{G}$ [54].

We categorize the types of disclosure into record level or identity disclosure, and attribute level disclosure [55, 67, 89]. Identity disclosure occurs when a record with multiple attribute values from the masked dataset $\mathbf{D}^M$ can be linked to an entity with the same masked attribute values in $\mathbf{G}^M$, which allows re-identification of the entity. It is important to note that a rare value (that only occurs in one or a small number of entities) for a single attribute could also lead to re-identification of the entity represented by that value by spontaneous recognition [55]. On the other hand, attribute level disclosure allows an attribute value (characteristics) of an entity from $\mathbf{D}^M$ to be accurately re-identified.

Our method to evaluate privacy is to simulate attacks (as described in Section 5.1) on protected data in the masked dataset ($\mathbf{D}^M$) by linking them to the masked version ($\mathbf{G}^M$) of the known unprotected (publicly available) data in $\mathbf{G}$ [149]. A disclosure risk (DR) measurement that boils down to a numerical value to quantify the privacy protection of a PPRL technique based on such a simulation attack allows to compare the privacy guarantees of several PPRL techniques.

The resulting DR measures should be numerical values that are normalized between 0.0 and 1.0, where $DR = 0.0$ means no disclosure at all and $DR = 1.0$ means a provable disclosure (i.e. exact and correct re-identification). These normalized values can also be specified as degrees of privacy as illustrated in Figure 5.3, following the

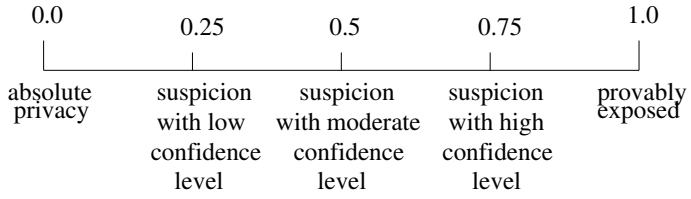


Figure 5.3: Degrees of privacy (adapted from [166]): Degrees range from absolute privacy, where the adversary cannot re-identify the actual value from the masked data, to provably exposed, where the adversary can provably re-identify the actual value.

work on degrees of connectivity or routing anonymity proposed by Reiter et al. [166]. In the following, we first consider linkage using a single attribute in defining the privacy evaluation model, and then we extend the model to include multiple attributes.

5.3.1.1 Disclosure risk of linkage using a single attribute

If an attribute value a^M of a record R^M in a masked dataset ($R^M \in \mathbf{D}^M$) matches with exactly one value for the same attribute in $\mathbf{G}^M$, then there is a provably exposed risk of disclosure of a^M, because the masked value a^M can be identified with this one-to-one match. A value a^M that matches with a small number of values in $\mathbf{G}^M$ has a risk of suspicion with a high probability, while a value a^M that matches with possibly many values in $\mathbf{G}^M$ has a disclosure risk with a low probability. Absolute privacy is attained with values a^M that match with either no values in $\mathbf{G}^M$ (i.e. no background information is available), or with all the values in $\mathbf{G}^M$, or with a user-specified acceptable number of values k (as discussed below).

Given n_g is the number of global values in $\mathbf{G}^M$ that are matched with an attribute value a^M in the masked dataset $\mathbf{D}^M$, the probability of suspicion of the corresponding value a^M is calculated as $1/n_g$. We then normalize this probability into the 0.0 to 1.0 interval, where 1.0 indicates provably exposed risk and 0.0 represents absolute privacy, as defined in Equation 5.1 (with $N = |\mathbf{G}^M|$).

$$P_s(a^M) = \frac{1/n_g - 1/N}{1 - 1/N} \tag{5.1}$$

Statistical disclosure risk measures: Using the probability of suspicion (P_s) values calculated for each of the values a^M in an attribute in $\mathbf{D}^M$, we present five different statistical disclosure risk (DR) measures to calculate the overall disclosure risk of the entire masked dataset $\mathbf{D}^M$.

As a running example, Table 5.1 shows the P_s values for a small made-up dataset of $n = 50$ values. This dataset contains, for example, five values of an attribute with $P_s = 1.0$, which means that these five attribute values match with only one attribute value out of $1,000$ in $\mathbf{G}^M$ (we assume $\mathbf{G}^M$ contains $1,000$ values of the same attribute),

Table 5.1: Probability of suspicion (P_s) of values a^M in an attribute in a small (fictional) example masked dataset $\mathbf{D}^M$. The total number of a^M values is $n = 50$, and the total number of global values for the same attribute in $\mathbf{G}^M$ is $N = 1,000$. Values are sorted according to their $P_s(a_i^M), 1 \leq i \leq n$ (adapted from [190]).

ID	P_s	ID	P_s	ID	P_s	ID	P_s	ID	P_s	ID	P_s	ID	P_s	ID	P_s	ID	P_s
1	1.0	2	1.0	3	1.0	4	1.0	5	1.0	6	0.5	7	0.5	8	0.5	9	0.5	10	0.5
11	0.5	12	0.5	13	0.5	14	0.5	15	0.5	16	0.33	17	0.33	18	0.33	19	0.33	20	0.33
21	0.33	22	0.25	23	0.25	24	0.2	25	0.2	26	0.2	27	0.2	28	0.2	29	0.2	30	0.1
31	0.1	32	0.1	33	0.1	34	0.1	35	0.1	36	0.01	37	0.01	38	0.01	39	0.01	40	0.01
41	0.002	42	0.002	43	0.002	44	0.002	45	0.0	46	0.0	47	0.0	48	0.0	49	0.0	50	0.0

ten attribute values that match with two global values ($P_s = 0.5$), and six attribute values that match with either no values or all the 1,000 values in $\mathbf{G}^M$ ($P_s = 0.0$).

1. Maximum risk (DR_{Max}): This measure allows us to define the maximum risk of disclosure of the masked dataset. It corresponds to the maximum value for the probability of suspicion P_s of attribute values a^M in the masked dataset, as explained in Equation 5.2.

$$DR_{Max} = \max_{a^M \in \mathbf{D}^M} (P_s(a^M)) \tag{5.2}$$

   In the example given in Table 5.1, the DR_{Max} is calculated as $DR_{Max} = 1.0$. This explains that the masked dataset has a maximum risk of 1.0 of any sensitive value being disclosed, i.e. there exists at least one attribute value in $\mathbf{D}^M$ that matches to a single value in $\mathbf{G}^M$.

2. Marketer risk (DR_{Mark}): It is important to know how many values in a masked dataset can be exactly re-identified. This risk is known as marketer risk and it evaluates the risk of disclosure from the perspective of a marketer adversary who wishes to re-identify as many values as possible in the disclosed dataset [48]. Marketer risk is measured as the proportion of values in $\mathbf{D}^M$ that have provably exposed risk of disclosure ($P_s = 1.0$) with one-to-one mapping in $\mathbf{G}^M$. DR_{Mark} for the running example in Table 5.1 is $5/50 = 0.1$ calculated using Equation 5.3 (as there are five of the fifty values having $P_s = 1.0$).

$$DR_{Mark} = |\{a^M \in \mathbf{D}^M : P_s(a^M) = 1.0\}|/n \tag{5.3}$$

   where $P_s(a^M)$ is the probability of suspicion of a value a^M in $\mathbf{D}^M$ and $n = |\mathbf{D}^M|$.

3. Mean risk (DR_{Mean}): The mean risk calculates the average of probability of suspicion values to evaluate the average disclosure risk. DR_{Mean} is calculated

using Equation 5.4. A value in the example masked dataset illustrated in Table 5.1 has an average probability of 0.28 of being re-identified, i.e. in average a value in $\mathbf{D}^M$ can be matched to around four values in $\mathbf{G}^M$.

$$DR_{Mean} = \frac{1}{n} \sum_{a^M \in \mathbf{D}^M} P_s(a^M) \tag{5.4}$$

4. Median risk (DR_{Med}): The median risk takes into account the distribution of probabilities of suspicions in the masked dataset and it gives the center of the distribution of disclosure risk values. DR_{Med} is calculated as shown in Equation 5.5, assuming $P_s(a^M)$ values are sorted in ascending order. DR_{Med} for the running example (with $n = 50$) results in $1/2 \times [P_s(a^M_{25}) + P_s(a^M_{26})] = (0.2 + 0.2)/2 = 0.2$.

$$DR_{Med} = \begin{cases} 1/2 \times [P_s(a^M_{n/2}) + P_s(a^M_{n/2+1})] & n \text{ is even} \\ P_s(a^M_{(n+1)/2}) & n \text{ is odd} \end{cases} \tag{5.5}$$

5. User acceptance (UA) mean risk (DR_{UAM}): If the users / data respondents of the linkage accept that the data will not be at a disclosure risk if a value a^M in their masked dataset matches with more than a certain number of values (k unique values) in the global dataset, then we can eliminate the risk of disclosing those masked values that are below the respective probability of suspicion, as the probabilities of suspicion of those values would be in the low confidence level, as shown in Figure 5.3. This approach is based on the concept of $(k, 1)$-anonymization mapping [72], where any value in a masked dataset is consistent with at least k original values and thus provides $(k, 1)$-anonymization privacy constraints. Ramachandran et al. [163] and Ferro et al. [66] proposed similar approaches to identify vulnerable records in a dataset that match with at most k global records in public data.

   The mean disclosure risk calculation can then be applied using Equation 5.4 after removing or setting to 0.0 the probabilities of suspicions that are acceptable by the users. For our running example, if the acceptable minimum number of global values that match with a single value in the masked dataset is set to $k = 4$ ($P_s = 0.25$), then in Table 5.1 we can set the probability of suspicion for the last 27 values (those with $P_s < 0.25$) to $P_s = 0.0$, and DR_{UAM} would then be calculated as $DR_{UAM} = 0.24$ using Equation 5.4.

We illustrate the distribution of P_s values in the example dataset shown in Table 5.1 and the calculated statistical disclosure risk measures in Figure 5.4. In Figure 5.5, we also present the distribution of P_s values in a real North Carolina (NC) voter dataset [31] (to be described in detail in Section 5.4) and the disclosure risk

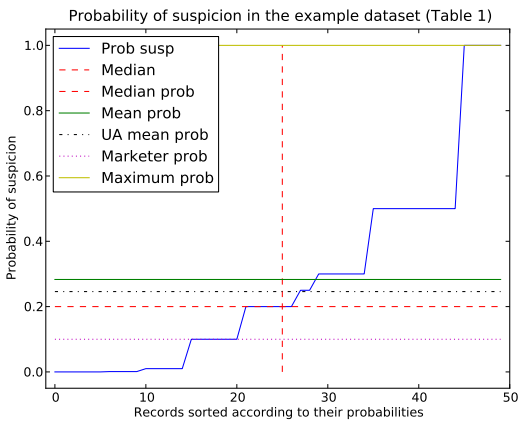


Figure 5.4: Distribution of probability of suspicion (P_s) values in the example dataset shown in Table 5.1 and the calculated statistical disclosure risk measures (presented in Section 5.3.1.1). The acceptable minimum number of global values that match with a single value is set to $k = 4$ ($P_s = 0.25$) for DR_{UAM} calculation (taken from [190]).

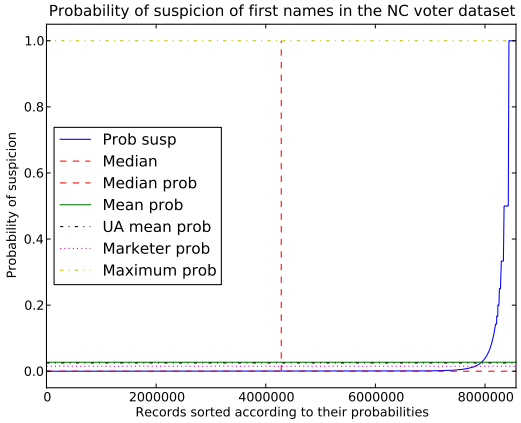


Figure 5.5: Distribution of probability of suspicion (P_s) of first name attribute values in the hash-encoded NC voter dataset [31] and the calculated disclosure risk measures for a simple dictionary attack on hash-encoded values using the same dataset as the global dataset. We set $k = 50$ for DR_{UAM} calculation (taken from [190]).

measures calculated for a simple dictionary attack on hash-encoded first name values using the same original NC voter dataset as the global dataset. As can be seen from these two figures, this set of statistical disclosure risk measures provide numerical and statistical information (maximum, mean, median, marketer, and UA mean) on the risk of disclosing a masked dataset.

Information theory measures: The standard information theory measures, such as information gain (IG) and relative information gain (RIG) [177], can also be used as DR measures based on a simulation attack on the masked dataset using the original dataset as the global dataset. IG assesses the possibility of inferring values in the original dataset $\mathbf{D}$, given its masked version $\mathbf{D}^M$ [177]. These information theory measures have been used for privacy evaluation in PPRL before [56, 105]. However, there are some limitations of these measures.

The first limitation is that the global dataset can only be assumed to be the same as the original linkage dataset ($\mathbf{G} \equiv \mathbf{D}$), while our statistical DR measures, proposed in the previous section, are independent of the choice of the global datasets. The second is that the IG measures provide only the overall total information gain from the masked dataset while our DR measures provide statistical summary information of the disclosure risk. We use a small example dataset shown in Table 5.2 to illustrate the calculation of IG and RIG.

Following the notation used by Durham [56] and Karaksidis et al. [105], the entropy $H(D)$ of a dataset $\mathbf{D}$ is defined as:

Table 5.2: Disclosure risk calculation of a small example dataset using IG and RIG. The global dataset is the same as the original dataset ($\mathbf{G} \equiv \mathbf{D}$) and the total number of global values in $\mathbf{G}$ is $N = n = 100$ (taken from [190]).

Original values in $\mathbf{D}$	Prob of values in $\mathbf{G}$ (n_g/N)	$log_2(n_g/N)$
peter	$30/100 = 0.3$	-0.522
pete	$20/100 = 0.2$	-0.464
smith	$50/100 = 0.5$	-0.5
$\mathbf{H(D) = -\sum(n_g/N)log_2(n_g/N) = 1.48}$		

Masked values in $\mathbf{D^M}$	Prob of values in $\mathbf{G}^M$ (n_g^M/N)	$H(\mathbf{D}\|\mathbf{D}^M = a^M)$
p360	$50/100 = 0.5$	$0.6 \times log_2 0.6 + 0.4 \times log_2 0.4 = 0.48$
s530	$50/100 = 0.5$	$1.0 \times log_2 1.0 = 0.0$
$\mathbf{H(D\|D^M) = -\sum(n_g^M/N) \times H(D\|D^M = a^M) = 0.48}$		

$$H(\mathbf{D}) = -\sum_{a \in \mathbf{D}} (n_g/N) log_2(n_g/N) \tag{5.6}$$

where n_g denotes the number of global values in $\mathbf{G}$ that match with a value a in $\mathbf{D}$, and N is the total number of values in $\mathbf{G}$. $H(D)$ is calculated for the example dataset with three made-up values (shown in Table 5.2) to 1.48, as explained in the left three columns in the table.

The conditional entropy of a dataset $\mathbf{D}$ given $\mathbf{D}^M$, $H(\mathbf{D}|\mathbf{D}^M)$, is defined as [105]:

$$H(\mathbf{D}|\mathbf{D}^M) = -\sum_{a^M \in \mathbf{D}^M} (n_g^M/N) H(\mathbf{D}|\mathbf{D}^M = a^M) \tag{5.7}$$

where n_g^M is the number of masked global values in $\mathbf{G}^M$ that match with a masked value a^M in $\mathbf{D}^M$, and N is the total number of values in $\mathbf{G}^M$. $H(\mathbf{D}|\mathbf{D}^M)$ for the running example is 0.48, as shown in the right three columns in Table 5.2. The entropy and conditional entropy form the basis for the information gain (IG) metric [177]. IG between $\mathbf{D}$ and $\mathbf{D}^M$ is defined as [105]:

$$IG(\mathbf{D}|\mathbf{D}^M) = H(\mathbf{D}) - H(\mathbf{D}|\mathbf{D}^M) \tag{5.8}$$

The running example results in $IG = 1.48 - 0.48 = 1.0$. The lower the value for IG is, the more difficult it is for an adversary to infer the original dataset from a masked dataset. The relative IG (RIG) measure normalizes the scale of IG ($0.0 \leq RIG(\mathbf{D}|\mathbf{D}^M) \leq 1.0$) with regard to the entropy of the original dataset $\mathbf{D}$ [105], and is defined as $RIG(\mathbf{D}|\mathbf{D}^M) = IG(\mathbf{D}|\mathbf{D}^M)/H(\mathbf{D})$. This is calculated as $RIG = 1.0/1.48 = 0.67$ for the running example dataset. Since RIG values are normalized between 0.0 and 1.0, they provide a marginal scale for comparison and evaluation.

5.3.1.2 Disclosure risk of linkage using multiple attributes

Record level (or identity) disclosure is possible when multiple attributes are used for linking, as it is generally the case. Disclosure risk calculation for linking on multiple

attributes can be done in three ways depending on the information available in the global dataset **G**.

1. The first case is if the global dataset contains combinations of individual values for all attributes (m attributes) used in the linkage and / or blocking, and each combination refers to one single entity, then the disclosure risk calculation is similar to the single attribute disclosure calculation. For each record R^M in the masked dataset $\mathbf{D}^M$, the number of global records n_g in $\mathbf{G}^M$ that have the matching (masked) values in the same attributes of R^M is calculated and the probability of suspicion of R^M then is $P_s(R^M) = 1/n_g$. An example would be if a combination of masked values of 'amilia' for the first name attribute and 'smith' for the last name attribute of a record R^M in $\mathbf{D}^M$ matches with $n_g = 2$ combinations / records in $\mathbf{G}^M$ that have the same masked values in the corresponding two attributes, then the probability of suspicion of R^M is calculated as $P_s(R^M) = 1/2 = 0.5$. This disclosure risk is higher than when only a single attribute is used in linkage, since multiple attributes (more information) of a record are compared with the entities in **G** that also have the same combination of attribute values (which could likely allow for an identity disclosure).

   The distributions of probability of suspicion values in a real NC voter dataset [31] and the calculated disclosure risk measures for a dictionary attack on hash-encoding of multiple attributes are shown in Figure 5.6. As the figure illustrates, when multiple attributes are used in linkage the disclosure risk becomes higher compared to the risk when only a single attribute is used, as was shown in Figure 5.5. The probability of suspicion and the disclosure risk values become higher with more attributes used. The number of unique combinations of attribute values of first name and city is smaller than the number of unique combinations of first name and last name which results in lower disclosure risk values for the former, as can be seen in Figures 5.6(a) and 5.6(b), respectively. The probability of suspicion of the four attributes first name, last name, city, and zipcode provide a marketer risk of $DR_{Mark} = 0.84$, as shown in Figure 5.6(d). This is similar to the results by Sweeney [181] which showed that around 90% of the population of the USA have a unique combined value of the three attributes zipcode, gender, and date of birth.

2. The second case is where the global dataset **G** contains combinations of attribute values as in case 1, but a certain subset of attribute values of a record R^M in $\mathbf{D}^M$ do not match with any values in the corresponding attributes in $\mathbf{G}^M$. For example, a masked first name value of 'amilia' in $\mathbf{D}^M$ matches with $n_{g_1} = 2$ masked first name values in $\mathbf{G}^M$ but the corresponding (masked) last name value 'dickson' in $\mathbf{D}^M$ does not match with any global values ($n_{g_2} = 0$). In such a case, we calculate the probability of suspicion as $P_s(R^M) = 1/(n_{g_1} \times N) = 1/(2 \times 1,000) = 0.0005$, by considering all the global values in $\mathbf{G}^M$ as possible matches ($N = 1,000$ in this example) for masked values that match with zero global values.

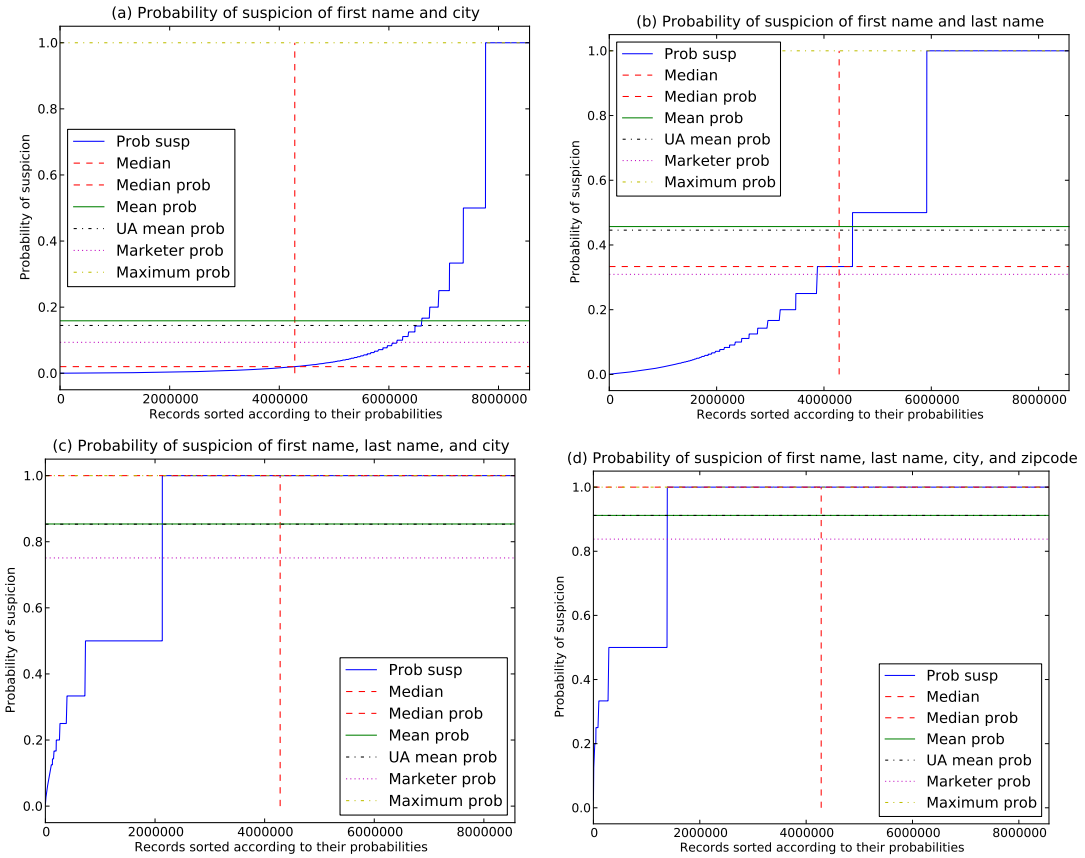


Figure 5.6: Distributions of probability of suspicion (P_s) of (a) first name and city, (b) first name and last name, (c) first name, last name, and city, and (d) first name, last name, city, and zipcode attribute values in the hash-encoded NC voter dataset [31] and the calculated disclosure risk measures for a simple dictionary attack on hash-encoded values using the same dataset as the global dataset. We set $k = 50$ for DR_{UAM} calculation (taken from [190]).

3. In the third case, the combinations of attribute values are not available in $\mathbf{G}^M$ (i.e. $\mathbf{G}^M$ consists of individual lists of global values for each attribute, but not the combinations of different attribute values). In this case, we multiply the number of global values that match with each attribute of a record R^M in $\mathbf{D}^M$ individually, in order to calculate the total number of global values that match with the record R^M. The probability of suspicion for R^M in this case would be $P_s(R^M) = 1/(n_{g_1} \times n_{g_2} \times \cdots \times n_{g_m})$, where m is the number of attributes used for the linkage. For example, if a record R^M in $\mathbf{D}^M$ with masked values of 'amilia' and 'smith' for the first name and last name attributes, respectively, matches with $n_{g_1} = 2$ global records in $\mathbf{G}^M$ that have the same (masked) first name value, and $n_{g_2} = 10$ global records that have the same (masked) last name value, then the probability of suspicion of that record is $P_s(R^M) = 1/(2 \times 10) = 1/20 = 0.05$.

5.3.2 Linkage Quality Measures

PPRL has to deal with the trade-off between privacy protection and the quality of linkage. Achieving more privacy generally means losing more data quality due to information lost in the protected / masked data as compared to the original data, and thus losing more quality of the linkage results. In practice, measuring the linkage quality is often difficult, because no truth data with known match status are available in many real-world applications [33]. However, the linkage quality can be assessed in a pilot study using synthetic data (representing real data characteristics) with known match status [35], or using the manual classification results obtained by clerical review in a record linkage process [29].

The quality of linkage in PPRL depends on both the quality of blocking as well as the quality of comparison and classification results (the three main steps in the PPRL process as is illustrated in Figure 2.2). The measures that are commonly used in information retrieval and data mining, such as precision, recall, and f-measure [140, 161], can be used to assess the quality of private matching and classification results. The quality of blocking can be measured using the pairs completeness and pairs quality measures [29, 62]. Based on the classification of the number of true matches (TM), false matches (FM), false non-matches (FN), true non-matches (TN), true matches included in the candidate record pairs generated by blocking (BM), and true non-matches included in the candidate record pairs (BN), the quality of linkage measures are defined as given below.

1. Precision: Precision is the fraction of record pairs classified as matches by a decision model that are true matches: $Precision = TM/(TM + FM)$.

2. Recall: Recall is the fraction of true matches that are correctly classified as matches by a decision model: $Recall = TM/(TM + FN)$.

3. F-measure: The F-measure or F-score is the harmonic mean of Precision and Recall, calculated as $F\text{-}measure = 2 \times (Precision \times Recall)/(Precision + Recall)$.

4. Pairs completeness (PC): Pairs completeness measures the effectiveness of a blocking technique in the record linkage process: $PC = BM/(TM + FN)$. This measure is similar to Recall.

5. Pairs quality (PQ): Pairs quality measures the efficiency of a blocking technique and is similar to the Precision measure: $PQ = BM/(BM + BN)$.

5.3.3 Scalability Measures

The third aspect of PPRL that makes the linkage process scalable to large real-world databases is dependent on the complexity of the protocol. The number of record pairs that are compared and classified using a PPRL technique determines the complexity of the protocol. A naïve pair-wise comparison of two databases is of quadratic complexity in the size of the databases [30]. Private blocking techniques [56, 104, 124]

are used in the first step of PPRL to reduce this large number of comparisons by removing pairs that are unlikely to refer to matches without comparing them in detail in the next step.

The efficiency of a blocking technique can be measured using reduction ratio (RR) [29, 62], which provides a value that indicates by how much a blocking technique is able to reduce the number of candidate record pairs that are being generated compared to all possible record pairs. A higher RR value means a blocking technique is more efficient in reducing the number of candidate record pairs that are being generated. RR is calculated as $RR = 1.0 - (BM + BN)/(TM + FN + FM + TN)$.

The complexity of techniques (or algorithms) used in PPRL has also an impact on the scalability of the protocol. Generally the complexity of algorithms is measured using the big-O notation [155] and practically evaluated in terms of efficiency using measures that are dependent on the computing platform and the networking infrastructure used, such as the total runtime, the memory space required to perform the linkage, and the size of messages or data communicated between parties in the protocol. The challenge with these platform dependent measures is how to normalize them into the 0.0 to 1.0 interval, to allow comparison of several PPRL solutions. A possible way to evaluate runtime, for example, is to calculate the average time required for a candidate record pair to be compared and classified using the most computationally intensive PPRL technique, and then multiply this value by the total number of candidate record pairs ($n^A \times n^B$, if no blocking is applied). This would give an upper bound for expected runtime. Then we can run all the PPRL solutions that need to be evaluated on the same computing platform, and measure their runtime. Using the upper bound calculated, the resulting runtime values can then be normalized between 0.0 and 1.0.

5.3.4 Overall Evaluation Score

A generic score can be calculated to evaluate PPRL techniques in terms of the three properties using the measures discussed in the above sections. For example, given the measures for disclosure risk (DR), linkage quality (LQ), and scalability (S), the overall evaluation score can be computed as the weighted average of the three measures.

$$score = \alpha(1 - DR) + \beta(LQ) + (1 - \alpha - \beta)(S) \qquad 0 \leq \alpha + \beta \leq 1 \tag{5.9}$$

Different weights for the three properties can be used depending on the importance of the properties with respect to application or user preferences. This final numerical score indicates the viability of a specific PPRL solution in terms of privacy, linkage quality, and scalability. A graphical representation of the three properties of PPRL provides more insights into the analysis and comparison of different PPRL techniques. Three-dimensional plots can be used to define the three properties along the three axes of the graphs to compare PPRL solutions, as will be shown in Chapter 10.

Table 5.3: The number of records in the datasets used for experiments, and the number of records that occur in both datasets of a pair (i.e. the number of true matches).

Dataset	Alice	Bob	Overlap
OZ-1730 No-mod / Mod	1,730	1,730	849 (50%)
OZ-17,294 No-mod / Mod	17,294	17,294	8,536 (50%)
OZ-172,938 No-mod / Mod	172,938	172,938	86,476 (50%)
OZ-1,729,379 No-mod / Mod	1,729,379	1,729,379	864,231 (50%)
NC	481,315	480,701	333,403 (70%)

5.4 Datasets Used

We used two real-world databases to empirically evaluate and compare the performances of our proposed PPRL solutions and several existing state-of-the-art PPRL solutions, as will be presented in the following chapters, using our proposed evaluation framework.

1. **OZ:** The first dataset is a real Australian telephone database that contains 6,917,514 records. We extracted four attributes commonly used for record linkage: given name (with 78,336 unique values), surname (with 404,651 unique values), suburb (town) name (13,109 unique values), and postcode (2,632 unique values). To generate datasets of different sizes, we sampled 0.1%, 1%, 10% and 100% of records in the full database twice each for *Alice* and *Bob*, and stored them into pairs of files such that 50% of records appeared in both files of a pair.

   The record pairs that occur in both datasets are exact matches. These datasets are labelled as 'No-mod' for no modification. To investigate the performance of PPRL solutions in the context of 'dirty data' (where attribute values contain errors and variations), we generated another series of datasets where we modified each attribute value by applying a randomly selected character edit operation (insert, delete, substitute, or transposition) [34]. These datasets are labelled as 'Mod' for modification. This leads to a much reduced number of exact matching record pairs and allows us to evaluate the quality of solutions in terms of the accuracy of approximate matching.

2. **NC:** The second dataset we used is a large real-world voter registration database from North Carolina (NC) in the US [31], containing records of over 8 million voters. We downloaded this dataset every two months since October 2011 to build a longitudinal dataset. We extracted four attributes (first name, surname, city, and zipcode) of 629,362 voters, such that 314,644 were represented by a single record and 314,718 by two or more records (up-to 6), where duplicate records contain both typographical errors, and actual variations and changes of values. We split this dataset (randomly) into two containing 481,315 and 480,701 records for *Alice* and *Bob*, respectively. Because voter registration numbers identify unique voters we can calculate the linkage quality. Table 5.3 provides an overview of the datasets we generated.

Table 5.4: The number of records in the corrupted datasets used for experiments, and the number of records that occur in both datasets of a pair (i.e. the number of true matches). Three levels of corruptions are applied which are labelled as 'Light-mod', 'Med-mod', and 'Heavy-mod'.

Dataset	Alice	Bob	Overlap
OZ Cor-4,611 Light / Med / Heavy	4,611	4,611	2,305 (50%)
OZ Cor-46,116 Light / Med / Heavy	46,116	46,116	23,058 (50%)
OZ Cor-461,167 Light / Med / Heavy	461,167	461,167	230,583 (50%)
NC Cor-5,488 Light / Med / Heavy	5,488	5,488	2,744 (50%)
NC Cor-54,886 Light / Med / Heavy	54,886	54,886	27,443 (50%)
NC Cor-548,860 Light / Med / Heavy	548,860	548,860	274,430 (50%)

3. **OZ Cor and NC Cor:** To simulate real-world 'dirty' data [29, 83] that exhibit data errors, such as data entry errors, phonetic variations, typographical mistakes, measurement or format variations, scanning and Optical Character Recognition (OCR) errors, and speech recognition errors of dictated values, we applied several corruption functions to the two above described databases (OZ and NC) and generated another set of corrupted datasets. These datasets allow us to evaluate the performance of PPRL algorithms (especially the quality of approximate matching) in a real-world setting. We used our flexible data Generation and Corruption of personal data tool (GeCo) [35] to corrupt the OZ and NC databases. The GeCo tool is available online: http://dmm.anu.edu.au/geco [183]. We applied four different corruption functions as given below:

   (a) **Character edit corruptor:** This function applies one of the four edits at a randomly selected position in an attribute value: insert a new character, delete the character, substitute with a new character, or transpose the character with one of its neighboring characters. Probabilities are set as equal for each of these edits except for the postcode / zipcode attribute where only the substitute edit is applied with probability of 1.0 (since zipcode / postcode values have fixed length of numerical characters).

   (b) **OCR corruptor:** This function replaces a character sequence in an attribute value with a new character sequence that has similar shape, such as '5' and 's' or 'm' and 'rn', to model OCR errors.

   (c) **Keyboard corruptor:** This function simulates typing errors by randomly replacing a character with a neighbouring character according to a keyboard layout matrix, such as 'a' and 's' in QWERTY keyboard layout. Probabilities for selecting a replacement in a row or column are set to 0.5 for each.

   (d) **Phonetic corruptor:** This function simulates phonetic errors by replacing a sequence of characters in an attribute value with a new sequence of characters that sounds similar, such as 'ph' and 'f', or 'rie' and 'ry'.

We applied all four corruption functions with equal probability (of 0.25) to all attributes except the postcode / zipcode attribute to which only the character edit corruptor and OCR corruptor are applied with probabilities set to 0.5.

For each dataset, we generated three variations with three different levels of corruptions, which are lightly modified (that corrupts only one attribute), moderately modified (corrupts two attributes), and heavily modified (corrupts all four attributes). The attributes are selected for corruption randomly with 0.25 of probability. The different levels of corruptions in the datasets allow us to evaluate how these corruptions affect the performance of the solutions. The set of corrupted datasets we generated for the experimental study are shown in Table 5.4.

It is important to note that we assume in this study that the datasets do not contain any missing values in the attributes used for linkage and / or blocking. However, real-world datasets do contain missing values due to various reasons [29]. Addressing the problem of missing values by preprocessing the datasets before conducting PPRL is out of the scope of this study.

5.5 Linkage Attacks

We present the methods for linkage attacks on our proposed solutions using an external global dataset, for privacy evaluation of the solutions, in the relevant chapters.

As explained by Duncan et al. [54], a drawback of using external datasets for risk calculation in disclosure control, is that the results are dependent on the choice of global datasets. Conducting linkage studies using a very large external dataset as the global dataset would require longer runtime and more computational resources which might not be practical for empirical evaluation. In addition, an external global dataset might not be available for privacy evaluation. In the worst case scenario, the global dataset $\mathbf{G}$ can be considered to be equivalent to the linked database $\mathbf{D}$ (i.e. $\mathbf{G} \equiv \mathbf{D}$). Conducting linkage studies of attacks such as frequency attacks, cryptanalysis attacks, and collusion using the masked dataset ($\mathbf{D}^M$) and the original dataset $\mathbf{D}$ as the global dataset would provide the highest disclosure risk in this worst case scenario. If a specific privacy technique provides sufficient privacy guarantees under such a worst case assumption, then the privacy technique would provide sufficient privacy in a real-world setting as well, because the global dataset available to an adversary is highly likely to be different from the original dataset. If $\mathbf{G}$ is larger than $\mathbf{D}$, then there would possibly be many global values in $\mathbf{G}^M$ that match a masked value in $\mathbf{D}^M$, which therefore results in lower disclosure risk. On the other hand, if $\mathbf{G}$ is smaller than $\mathbf{D}$, there might be masked values in $\mathbf{D}^M$ that do not match with any global values in $\mathbf{G}^M$, again resulting in lower disclosure risk.

We consider the worst case assumption of $\mathbf{G} \equiv \mathbf{D}$ (though they are not practical in real applications, they provide a baseline for privacy comparison of PPRL techniques) in this thesis for privacy evaluation and comparison of several PPRL techniques in Chapters 6, 7, 8, 9, and 10. However, the proposed framework can be used with

any choice of global datasets (as long as all the techniques are compared for privacy against attacks using the same global dataset).

5.6 Computing Platform Used

We prototyped all the solutions presented in the remaining chapters using the Python programming language (version 2.7.3), due to its flexibility and efficiency for rapid prototype development model. We also used the Freely extensible biomedical record linkage (Febrl) [32] tool for implementing the approximate string comparison functions and phonetic encoding function. All tests were run on a compute server with two 64-bit Intel Xeon (2.4 GHz) CPUs, 128 GBytes of main memory and running Ubuntu 12.04. The programs and (the small) test datasets are available upon request.

5.7 Summary

In this chapter, we have provided the details of the experimental setup used to conduct an effective and practical empirical study of our PPRL solutions which will be presented in the next four chapters. We have proposed a comprehensive evaluation framework for PPRL solutions that enables assessment and comparison of different solutions in terms of the three main properties of PPRL, which are scalability, linkage quality, and privacy.

Scalability and quality of PPRL solutions can be assessed using the standard measures that have been used in the literature. However, numerical measures to quantify the privacy guarantees provided by a solution need to be defined. We have defined five different disclosure risk measures that can be used to measure the privacy by simulating frequency linkage attacks using an external global dataset. Then we presented the details of datasets and computing platform used for the empirical study of our research.

Future work includes extending the evaluation framework to address the problem of privacy-preserving linking of multiple sources and to consider different adversarial models such as the covert model [8] or accountable computing [95] for privacy evaluation. A limitation with the proposed disclosure risk measures for privacy evaluation is that they strongly depend on the attack methods used and the fine tuning of parameters of those attack methods. More research is required in the direction of developing efficient and effective attack methods for privacy evaluation.

Chapter 6

Three-Party Private Blocking

Addressing the scalability aspect of privacy-preserving record linkage (PPRL) using efficient private blocking techniques has been identified as one of the important research directions in Chapter 4. In this chapter, we propose an efficient three-party private blocking solution based on the sorted neighborhood approach that combines the k-anonymous mapping and reference values privacy techniques, as will be described in Section 6.2. We analyze the solution in Section 6.3 with respect to complexity, privacy, and quality, and in Section 6.5 we validate these analyses through an empirical study based on a linkage attack proposed in Section 6.4. Finally we summarize our findings in Section 6.6.

6.1 Introduction

The scalability challenge of PPRL has been addressed by several recent approaches that adapt existing blocking techniques, such as standard blocking [65], mapping-based blocking [96], clustering [44], and locality sensitive hashing [113], into a privacy-preserving context. One popular blocking technique used in traditional record linkage is the sorted neighborhood approach [52, 84], where database records are sorted according to their sorting key values (SKVs - values of an attribute or a combination of attributes used to sort the records) over which a sliding window is moved. Candidate pairs are generated from the records that are within the current window.

The sorted neighborhood approach is very efficient compared to other blocking techniques in that its resulting number of candidate record pairs is $O((n^A + n^B)w)$, compared to $O((n^A \cdot n^B)/b)$ for other blocking techniques [30], where n^A and n^B are the respective number of records in the two databases $\mathbf{D}^A$ and $\mathbf{D}^B$ to be linked, b is the number of blocks generated, and w is the size of the window. However, the use of sorted neighborhood methods for blocking has only received a little attention in the PPRL context [106].

Karakasidis et al. [106] proposed a sorted neighborhood-based private blocking approach using k-nearest neighbor (or k-medoids) clustering to group similar (candidate) records into the same block individually by the database owners (which is less efficient than the sorted neighborhood approach in terms of computation complexity), followed by using the sorted neighborhood approach by the third party to group

Table 6.1: Notation used in this chapter.

$\mathbf{D}^A$, $\mathbf{D}^B$	Databases held by database owners *Alice* and *Bob*, respectively
$\mathbf{R}$	Publicly available reference dataset
$\mathbf{R}'$	Lists of reference values selected from $\mathbf{R}$ by *Alice* and *Bob*
n^A, n^B	Number of records in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
n_R, n	Number of reference values in $\mathbf{R}'$, and number of records in databases
k	Minimum number of records in a block
$\mathbf{S^A}$, $\mathbf{S^B}$	Set of sorted neighborhood clusters (SNC) in *Alice*'s and *Bob*'s databases, respectively
$\mathbf{O}^A$, $\mathbf{O}^B$	Set of k-anonymous clusters in *Alice*'s and *Bob*'s databases, respectively
$\mathbf{C}$	Set of candidate record pairs
$sim(\cdot,\cdot)$	Function used to calculate similarities between two reference values ($0 \leq sim(\cdot,\cdot) \leq 1$)
s_t	Minimum similarity threshold value to determine a pair of reference values as similar, $0 \leq s_t \leq 1$

candidate blocks from both database owners. In contrast, we aim to develop more efficient technique by using only the sorted neighborhood approach for the private blocking of databases.

In this chapter, we propose an efficient three-party blocking technique for PPRL based on the sorted neighborhood approach using a combination of two privacy techniques: k-anonymous mapping [72, 182] and public reference values [154]. The aim of this approach is to efficiently create k-anonymous blocks represented by reference values from which candidate pairs are generated, without revealing any information that can be used to infer individual records and their attribute values. We propose two versions of k-anonymous mapping to generate k-anonymous blocks. The first is based on similarity between reference values (which we call **SNC-3PSim**) and the second on the size of blocks (**SNC-3PSize**). In the following section we describe our protocol in detail.

6.2 Proposed Solution

In this section we describe the steps of our sorted neighborhood clustering (SNC)-based three-party private blocking protocol. Assume *Alice* and *Bob* are the two owners of their respective databases $\mathbf{D}^A$ and $\mathbf{D}^B$, and *Carol* is the trusted third party. *Alice* and *Bob* share the sorted reference list $\mathbf{R}'$ containing n_R reference values selected from the publicly available reference dataset $\mathbf{R}$.

Reference values have been used in PPRL as a privacy technique for mapping the attribute values in a database into a masked version such that the distances between the actual values are preserved in the masked version [104, 154, 170, 210]. Such reference values can either be constructed with random faked values, or they can be extracted from a publicly available dataset, for example, all unique surnames taken from a public telephone directory or electoral roll (such as the NC voter dataset [31] as described in Section 5.4).

Figure 6.1 illustrates the three-party setting for private blocking, and Table 6.1 summarizes the notation we use in this chapter. We illustrate the steps with an example consisting of two small databases with given names and surnames, as shown in Figure 6.2. The two database owners *Alice* and *Bob* perform the following steps, as illustrated in Figures 6.2 to 6.4 (taken from [189]).

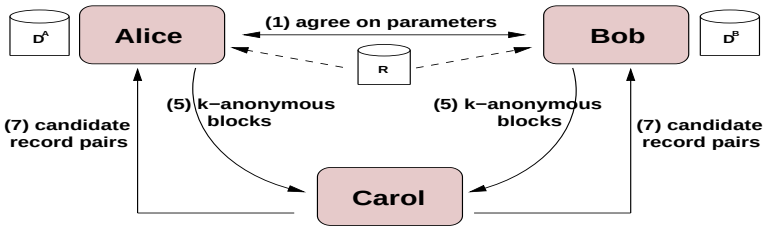


Figure 6.1: Outline of the proposed three-party private blocking protocol (taken from [189]). The numbers given correspond to the protocol steps described in Section 6.2 that involve an exchange of data between parties.

D^A

RecID	Surname	GivenName	SKV
RA1	millar	robert	millarrobert
RA2	marten	amyas	martenamyas
RA3	melar	gail	melargail
RA4	miller	robart	millerrobart
RA5	morley	william	morleywilliam
RA6	philips	colin	philipscolin
RA7	smith	alisen	smithalisen
RA8	sampson	taylor	sampsontaylor

Reference values

R'

1	*millar*
2	*myler*
3	*robinson*
4	*smith*

D^B

RecID	Surname	GivenName	SKV
RB1	millar	robert	millarrobert
RB2	marris	roberto	marrisroberto
RB3	malar	gayle	malargayle
RB4	perris	charles	perrischarles
RB5	robbins	william	robbinswilliam
RB6	robertson	amy	robertsonamy
RB7	samuell	tailor	samuelltailor
RB8	smeeth	rupert	smeethrupert
RB9	smith	alison	smithalison

Figure 6.2: Example databases held by *Alice* ($\mathbf{D}^A$) and *Bob* ($\mathbf{D}^B$) with surname and given name attributes and their SKVs, and a list of reference values ($\mathbf{R}'$) along with their position values, used to illustrate the protocol described in Section 6.2.

1. They agree upon the list of attributes to be used as the sorting keys, the reference dataset $\mathbf{R}$, the number of reference values to be used n_R, the minimum number of records in a block k, a similarity (approximate string comparison) function $sim(\cdot,\cdot)$ to compare reference values ($0 \leq sim(\cdot,\cdot) \leq 1$), and the minimum similarity threshold s_t used to decide if two blocks are to be merged in the SNC-3PSim approach described in Step 4.

2. *Alice* and *Bob* each randomly selects and sorts n_R ($n_R \leq |\mathbf{R}|$) reference values in lexicographical order. The value for n_R can be chosen as $n_R = \min(|\mathbf{D^A}|, |\mathbf{D^B}|)/k$, so that each block will contain roughly around k database records. It is important to note that both *Alice* and *Bob* have the same list of sorted reference values $\mathbf{R}'$ at the end of this step and *Carol* does not know $\mathbf{R}'$. A secret random seed shared by *Alice* and *Bob* (but not known to *Carol*) can be used to select the same set of values from $\mathbf{R}$ into $\mathbf{R}'$ by both *Alice* and *Bob*.

3. *Alice* and *Bob* individually insert their records based on the records' SKVs into the sorted list of reference values to create SNC blocks, as shown in Figure 6.3. An inverted index data structure can be used to efficiently insert records where the keys are the reference values and the corresponding values contain a list of SKVs that are lexicographically sorted before the reference value.

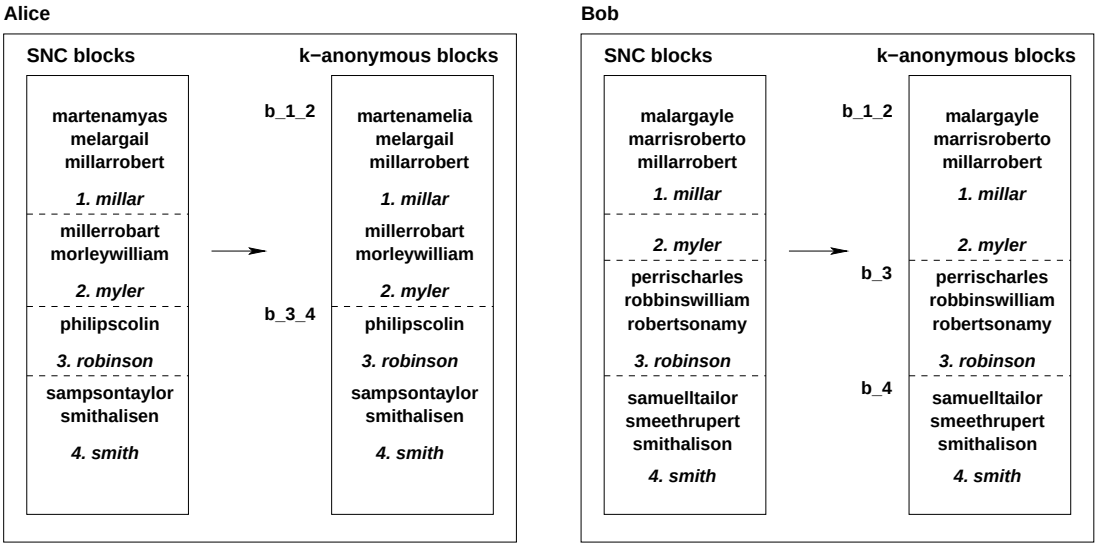


Figure 6.3: Insertion of SKVs into the sorted list of reference values (protocol step 3) where each block is represented by one reference value (shown in italics font), and merging of blocks to create k-anonymous blocks (protocol step 4) where each block is represented by one or more reference values (in this example, $k = 3$).

Algorithm 6.1 : Size-based merging

Input: **S**: Set of SNC blocks $(b : [v_1, \ldots, v_l])$
k: Minimum number of elements in a block
Output: **O**: Set of k-anonymous blocks
$('b\_x \ldots \_(x+y)' : [v_1, \ldots, v_k])$

1: $ids = []; sizes = []$ {[] is an empty list}
2: **for** $(r, c) \in$ **S do**
3: $ids += [r]; sizes += [len(c)]$
4: **end for**
5: min_size = $min(sizes)$
6: **while** min_size $< k$ **and** $len(ids) > 1$ **do**
7: min_size_block = **S**.$getID(len(b)$ =min_size)
8: block_vals = **S**[min_size_block]
9: $i = ids.getindex$(min_size_block)
10: prev_block = ids[i-1]; next_block = ids[i+1]
11: **if** len(**S**[prev_block]) $<$ len(**S**[next_block]) **then**
12: block_id =min_size_block +prev_block
13: block_vals += **S**[prev_block]
14: **else**
15: block_id = min_size_block + next_block
16: block_vals += **S**[next_block]
17: **end if**
18: $update(sizes)$; min_size = $min(sizes)$
19: **O**[block_id] =block_vals
20: **end while**

Algorithm 6.2 : Sim-based merging

Input: $\mathbf{R}'$: List of sorted reference values $[r_1, \ldots, r_{n_R}]$
S: Set of SNC blocks $(b : [v_1, \ldots, v_l])$
k: Minimum number of elements in a block
s_t: Minimum similarity threshold
$sim(\cdot, \cdot)$: Similarity comparison function
Output: **O**: Set of k-anonymous blocks
$('b\_x \ldots \_(x+y)' : [v_1, \ldots, v_k])$

1: $i = 0$
2: **while** $i < n_R$ **do**
3: block_vals = []; block_id = 'b'
4: num_vals = 0; $j = 0$
5: sim_val = 0.0
6: **while** (num_vals $< k$ **and** $i + j < n_R$) or (sim_val $\geq s_t$ **and** $i + j < n_R$) **do**
7: $r_i = \mathbf{R}'[i + j]$; $b = \mathbf{S}[r_{i+j}]$
8: num_vals += $len(b)$
9: block_vals += b
10: sim_val = $sim(r_{i+j}, r_{i+j+1})$
11: block_id += $str(i + j)$ + '_'
12: j+= 1
13: **end while**
14: **O**[block_id] = block_vals
15: i+= j
16: **end while**

4. The next step is to create k-anonymous blocks. After inserting records into the sorted list of reference values there will be n_R SNC blocks each represented by one reference value. However, to provide k-anonymous privacy characteristics,

each database owner has to perform k-anonymous mapping [182] by merging their blocks in such a way that each block contains at least k database records. Block IDs are assigned to the merged (k-anonymous) blocks such that they consist of the position values of the reference values that reside in the merged blocks. We use block IDs of the form '$b\_x\_(x{+}1)\cdots\_(x{+}y)$', where x is the position value of the first reference value in the corresponding block, and $(y+1)$ is the number of reference values in that block. The merging of blocks to create k-anonymous blocks is shown in Figure 6.3. This merging process can be done in two different ways, as illustrated in Algorithms 6.1 and 6.2 (individually by *Alice* and *Bob*).

(a) **SNC-3PSize:** Blocks are merged until the minimum size of the blocks becomes greater than or equal to k by iteratively identifying the smallest block. Algorithm 6.1 provides an overview of this method. In lines 2-5, we find the block with the smallest number of elements, and if this number is less than k (line 6) we merge it with the smaller of its two neighboring blocks. $getID(.)$ and $getIndex(.)$ are functions used to get the ID and the index of a block in the set of SNC blocks (**S**), respectively, while $len(.)$ is a function used to calculate the length of a block. We repeat this merging (lines 6-20) until the size of the smallest block is at least k. The values in the merged block are stored in the output set of k-anonymous blocks (**O**) in line 19 along with the block ID. This method results in similar number of records in most blocks. However, true matches might be missed depending on the value for k, because the similarity between reference values is not considered.

(b) **SNC-3PSim:** Blocks are merged until the number of elements in each of them becomes greater than or equal to k and the similarity between reference values of adjacent blocks becomes less than the threshold s_t. This approach follows recent work on adaptive sorted neighborhood for duplicate detection [52]. Algorithm 6.2 shows the main steps involved in this method. The blocks are merged until their size becomes greater than or equal to k (line 6 in Algorithm 6.2). If the size of the (merged) block is at least k, we compute the similarity of the next block's reference value r_{i+j+1} with the current block's reference value r_{i+j}, and if this similarity value $sim(r_{i+j+1}, r_{i+j})$ is greater than or equal to s_t, then we continue to merge the next block with the current block. Lines 6-13 show this loop of merging. In line 14, the values in the merged block are stored in the output set of k-anonymous blocks (**O**) with the block ID. This method is more likely to insert similar values into one block, at the cost that the resulting larger blocks will generate more candidate record pairs.

5. Once the k-anonymous blocks are created, they need to be sent to a third party, *Carol*, to generate candidate record pairs. The values in the blocks are replaced by their (encrypted) record IDs.

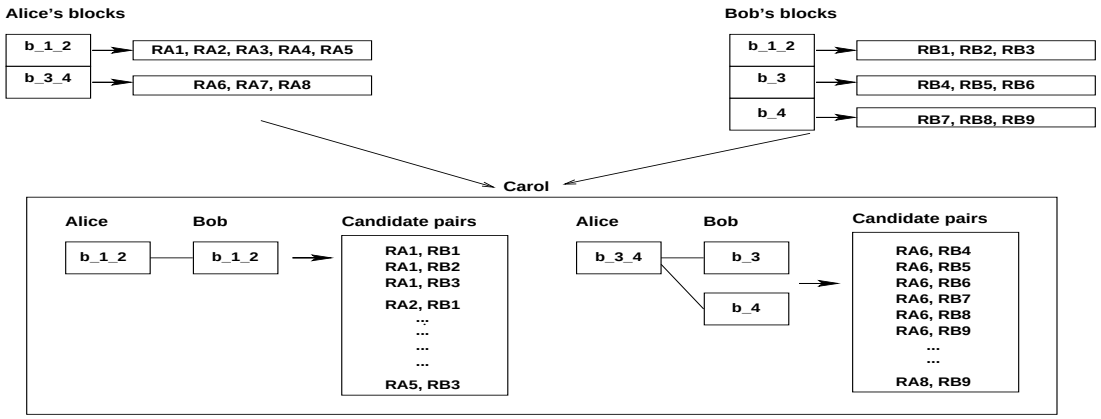


Figure 6.4: The merging of corresponding blocks from *Alice* and *Bob* to generate candidate record pairs as conducted by *Carol* in Step 6 of the protocol.

Carol receives the *k*-anonymous blocks from *Alice* and *Bob* and performs the following steps:

Algorithm 6.3 : Generating candidate record pairs

Input: $\mathbf{O}^A$: *Alice*'s set of *k*-anonymous blocks
$\mathbf{O}^B$: *Bob*'s set of *k*-anonymous blocks
Output: **C**: Set of candidate record pairs

```
1: for (i^A, b^A) ∈ O^A do
2:     ref_pos_vals_alice = get_ref_pos_vals(i^A)
3:     bob_blocks = []; b^B = []
4:     for ref_pos ∈ ref_pos_vals_alice do
5:         bob_blocks += O^B.getIDs(ref_pos)
6:         b^B += O^B[bob_blocks]
7:     end for
8:     for alice_rec_ID ∈ b^A do
9:         for bob_rec_ID ∈ b^B do
10:            C += [alice_rec_ID, bob_rec_ID]
11:        end for
12:    end for
13: end for
```

6. *Carol* finds corresponding blocks from *Alice* and *Bob* based on the reference position values in the block IDs to generate candidate record pairs, as illustrated in Figure 6.4 and Algorithm 6.3. From the block IDs, *Carol* extracts the position values of the reference values that reside in the corresponding blocks using $get\_ref\_pos\_vals(\cdot)$ (line 2). In lines 3-7, *Carol* finds for each of *Alice*'s blocks all of *Bob*'s blocks that need to be merged by extracting the position values from *Alice*'s block IDs. *Carol* then performs a nested loop (lines 8-12) over *Alice*'s blocks and *Bob*'s corresponding blocks, and stores the record pairs from *Alice*'s and *Bob*'s records in the output set of candidate record pairs **C**.

7. *Carol* sends the record IDs of the candidate pairs **C** back to *Alice* and *Bob* which then employ a private matching and classification protocol on the generated candidate record pairs [56, 154, 174] (as will be proposed in Chapters 8 and 9).

6.3 Analysis of the Protocol

In this section we analyse our SNC-based three-party private blocking approach in terms of complexity, privacy, and quality of blocking.

6.3.1 Complexity

Assuming both databases contain n records ($n = n^A = n^B$) and n_R reference values are selected from the reference dataset, sorting these n_R reference values is of $O(n_R \; log \; n_R)$ complexity, and inserting the n database records into the sorted list of reference values is of $O(n)$ complexity. At the end of this step (step 3), there will be n_R SNC blocks each represented by one reference value. Merging these blocks to create k-anonymous blocks in step 4 requires a loop over n_R blocks, which results to $O(n_R)$ merged blocks. Sending the k-anonymous blocks to *Carol* is of $O(n)$ communication complexity. *Carol* performs a loop over the blocks (a maximum of n_R blocks) in step 6 to merge and generate candidate record pairs from *Alice*'s and *Bob*'s records. The computation complexity of this step is $O(n_R^2)$.

The overall complexity of our approach is linear in the size of the databases n and quadratic in the number of reference values n_R. The number of generated blocks will be on average n/k. Assuming each block contains k records, the number of candidate record pairs generated by our approach is $\frac{n}{k} \times k^2 = n \; k$.

6.3.2 Privacy

We assume that all parties that participate in the protocol follow the honest but curious (HBC) adversarial model [78], in that they follow the protocol while trying to find out as much as possible about the data from the other party. We analyze the privacy of the protocol, by evaluating what can be learned by each of the parties from the data they communicate with each other during the protocol.

In step 5, *Alice* and *Bob* send their k-anonymous blocks (with encrypted record identifiers) to *Carol* to generate candidate blocks. Since each block consists of at least k records, it is difficult for *Carol* to perform a frequency linkage attack (as will be presented in Section 6.4) to infer individual records. This is because each record in a k-anonymous block is consistent (or similar) with at least k records in the same block resulting in a maximum disclosure risk of $DR_{Max} = 1/k$ (as was discussed in Section 5.3.1). The value for k has to be chosen carefully. A higher value for k provides stronger privacy guarantees but more candidate record pairs will be generated.

Furthermore, *Carol* does not know the list of reference values used ($\mathbf{R}'$), and therefore she cannot learn the blocking details such as k-anonymous mapping. *Alice* and *Bob* cannot learn anything about each other's data as they do not communicate any data between them. However, as with other three-party protocols, collusion between the third party and one of the database owners with the aim to identify the other database owner's data, is a privacy risk in this approach as well [78].

In step 7, *Carol* sends back the candidate record pairs to *Alice* and *Bob* to perform linkage using a private matching and classification technique [56, 154, 174], which

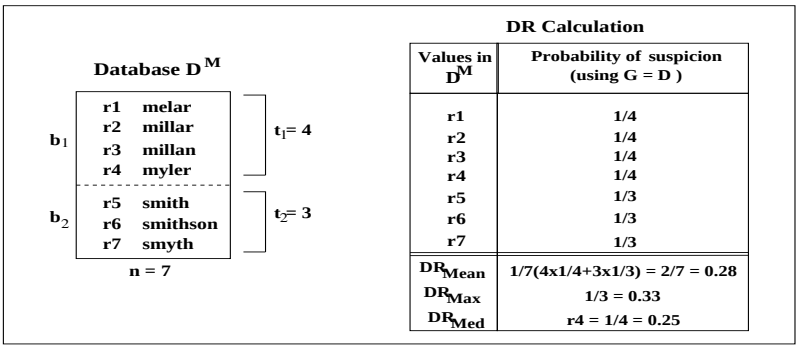


Figure 6.5: An attack method for three-party private blocking solutions [56, 104, 124, 189] using $\mathbf{G} \equiv \mathbf{D}$ and the statistical disclosure risk (DR) measures (presented in Section 5.3.1) calculated for the linkage attack. The example dataset is made-up for illustrative purposes. Records $r_1 \ldots r_4$ are consistent with 4 records in the same block b_1 of size $t_1 = 4$ resulting in probability of suspicion of $P_s = 1/4$, while records $r_5 \ldots r_7$ are consistent with 3 records in the same block b_2 of $t_2 = 3$ resulting in $P_s = 1/3$. The total number of records in $\mathbf{D}$ is $n = 7$ (taken from [190]).

should not reveal any sensitive information (this step is outside of our protocol).

6.3.3 Quality

A good blocking technique should be able to group all similar records into the same block - i.e., effectiveness measured by pairs completeness (PC), while keeping the number of candidate record pairs generated as small as possible - i.e., efficiency measured by reduction ratio (RR) [30]. SNC-3PSim retrieves more similar records compared to SNC-3PSize as the similarity between reference values is used in SNC-3PSim to determine the maximum size of a block. This results in higher PC and lower RR for SNC-3PSim compared to SNC-3PSize.

The value of k also determines the PC and RR of blocking. A higher value for k results in higher PC but lower RR. An optimal k needs to be set such that high values for both PC and RR are achieved while k guarantees sufficient privacy as well.

6.4 Linkage Attack

Based on the evaluation model proposed in Chapter 5, we now describe the frequency linkage attack for our SNC-3P approaches using an external global dataset $\mathbf{G}$ (which is assumed to be the same original dataset $\mathbf{D}$ in the worst case as discussed in Section 5.5) for privacy evaluation.

Generally in three-party private blocking techniques, only the number of blocks (n_B) and the size of each block ($t_i = |b_i|, 1 \leq i \leq n_B$) are revealed to the third party that participates in the protocol. In the masked (blocked) database $\mathbf{D}^M$, a record r is consistent or similar with $t_i - 1$ other records in the same block b_i where r resides. For our SNC-3P approaches, $t_i \geq k$. If r is consistent with t_i records (including r) in the local (masked) database then there would be at least t_i global matching values

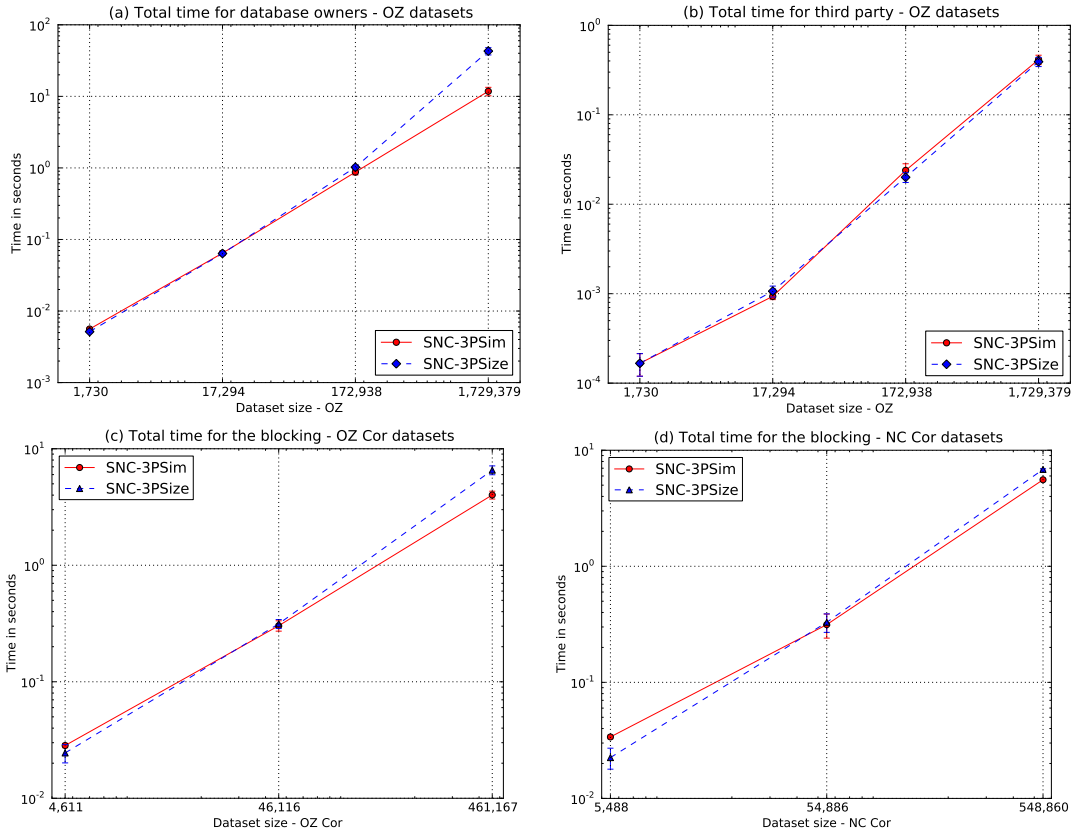


Figure 6.6: Total time of the SNC-3P blocking approaches required by (a) database owners, and (b) third party on the OZ datasets, (c) total blocking time on the OZ Cor datasets, and (d) total blocking time on the NC Cor datasets averaged over the results of all variations of each dataset.

n_g ($\geq t_i$) in $\mathbf{G}$. Therefore the probability of suspicion (P_s) of a record r in private blocking is $P_s = 1/t_i$ ($\geq 1/n_g$) under the worst case assumption ($\mathbf{G} \equiv \mathbf{D}$).

The general attack method and disclosure risk calculation for three-party private blocking solutions on a small made-up (of 7 records) dataset with two blocks b_1 of size $t_1 = 4$ and b_2 of $t_2 = 3$ are illustrated in Figure 6.5. The maximum disclosure risk is $DR_{Max} = 1/min(t_i) = 1/3$, because each masked value in the blocked database $\mathbf{D}^M$ is consistent with at least 3 values in the database. Lower DR values calculated under the worst case validates that a specific private blocking solution would provide sufficient privacy guarantees against frequency attacks in the actual setting.

6.5 Experimental Evaluation

In this section, we present and discuss the results of the experimental evaluation study of our SNC-3P based approaches conducted on the datasets described in Section 5.4 using the evaluation framework proposed in Chapter 5. The default parameters were set as $s_t = 0.9$ and $k = 100$ (this gives best results in terms of all three

properties of PPRL, as will be shown in Figure 6.9). Different values for k in the range of $[3, 10, 20, 50, 100]$ were also used to evaluate the performance of the SNC-3P blocking against k. A combination of two attributes was used as the sorting key: given name / first name, and surname / last name.

Figure 6.6 shows the total blocking time required for private blocking of the SNC-3P approaches on different datasets. As can be seen from the figure, the SNC approach (both variations of SNC-3PSim and SNC-3PSize) is very efficient (in terms of blocking time) and is also almost linear in the size of the databases. SNC-3PSim is faster in blocking the databases by the database owners, as shown in Figure 6.6 (a), because the SNC-3PSize involves several iterations until the minimum bock size becomes at least k, as was explained in Algorithm 6.1. Both SNC-3PSim and SNC-3PSize require almost same runtime by the third party, as shown in Figure 6.6 (b). Hence SNC-3PSim requires shorter time in total than SNC-3PSize, especially on the largest datasets (see Figures 6.6 (c) and 6.6 (d)).

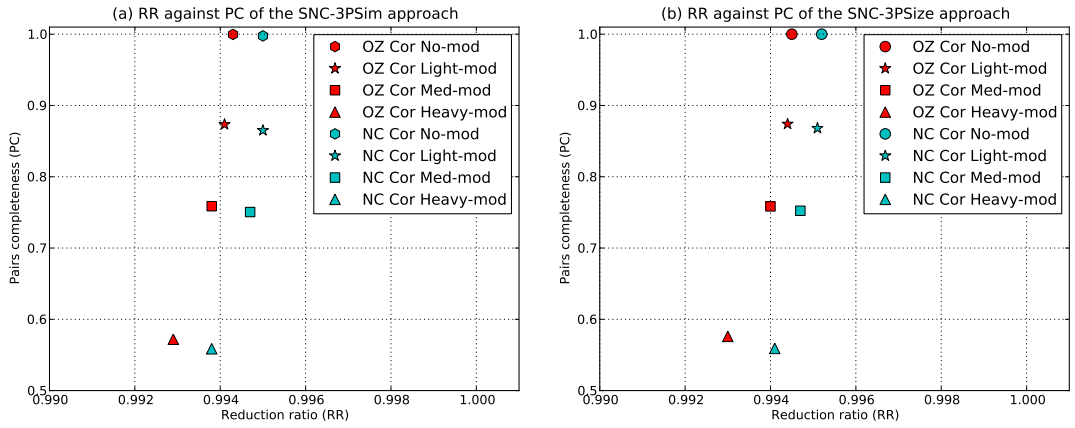


Figure 6.7: A comparison of reduction ratio (RR) against pairs completeness (PC) of the (a) SNC-3PSim and (b) SNC-3PSize solutions on the OZ Cor-46,116 and NC Cor-54,886 datasets with No-mod, Light-mod, Med-mod, and Heavy-mod variations.

A comparison of the efficiency and effectiveness of the SNC-3P blocking measured by reduction ratio (RR) and pairs completeness (PC), respectively, is presented in Figure 6.7 on the OZ Cor-46,116 and NC Cor-54,886 datasets with different levels of data modifications (corruptions). Different levels of corruptions applied to the datasets (as was described in Section 5.4) allow us to evaluate the performance of approximate matching of our solutions in the presence of data errors. As shown in the figure, both approaches achieve high RR and PC when no modification is applied to the datasets, and then the values decrease as the level of modifications increases. Both approaches of SNC-3P achieve similar values for RR and PC. However, as we discussed in Section 6.3.3, a slightly higher PC and a lower RR are achieved by the SNC-3PSim approach, comparatively.

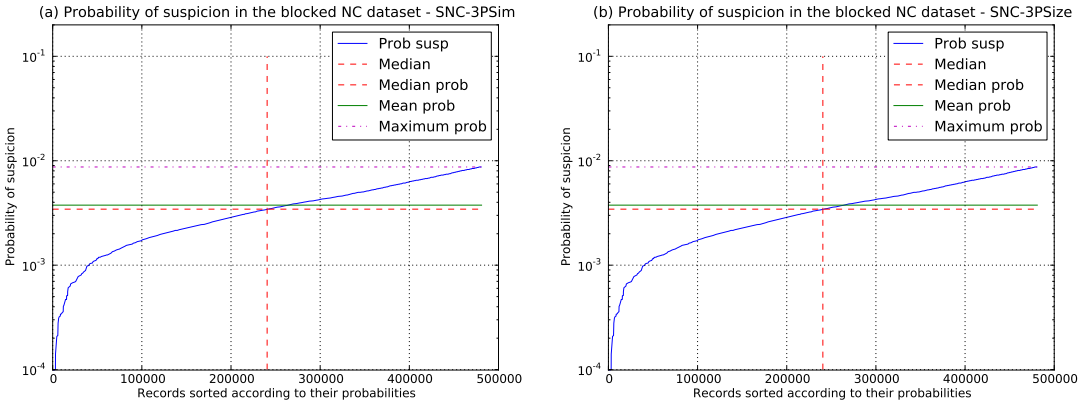


Figure 6.8: A comparison of probability of suspicion (P_s) values of the blocked NC dataset generated by the (a) SNC-3PSim and (b) SNC-3PSize approaches and the calculated disclosure risk (DR) measures on the NC dataset.

We next study the distribution of P_s values in the masked and blocked NC dataset by the SNC-3P blocking solutions, and the DR measures (proposed in Section 5.3.1) calculated for the frequency linkage attack (described in Section 6.4). The DR results illustrated in Figure 6.8 show the privacy aspects of both SNC-3P approaches by providing lower values for the disclosure risk even in the worst case assumption.

Finally, we investigate the performance of our solution against different values of k in Figure 6.9. The SNC-3P approaches achieve high values for both PC and RR even when $k = 100$, which gives a strong privacy guarantee (Figure 6.9 (a)). As discussed in Section 6.3.3, the effectiveness of blocking (PC) increases with k while efficiency (RR) decreases. PC is slightly higher for SNC-3PSim compared to SNC-3PSize (though the difference is not significant) due to the effectiveness of SNC-3PSim as was discussed in Section 6.3.2.

Figure 6.9 (b) shows the scalability (in terms of blocking time) of our approach with different values of k. Interestingly, the total time required for blocking decreases with k. This is because when k gets larger, a smaller number of larger sized blocks are generated, and thus it takes shorter time for k-anonymous mapping and for generating candidate record pairs. The difference in total time against k is considerable with the SNC-3PSize approach, as illustrated in Figure 6.9 (b).

The DR measures calculated based on the linkage attack given in Section 6.4 against different values of k is illustrated in Figure 6.9 (c). As discussed in Section 6.3.2, disclosure risk decreases when k becomes larger, because a masked value in the blocked dataset will have at most $1/k$ of probability of suspicion and this value decreases with k. These empirical results show that privacy (measured by DR) and quality (measured by PC) of the SNC-3P approaches increase with k while scalability or efficiency (measured by RR) decreases with k, though at a smaller cost.

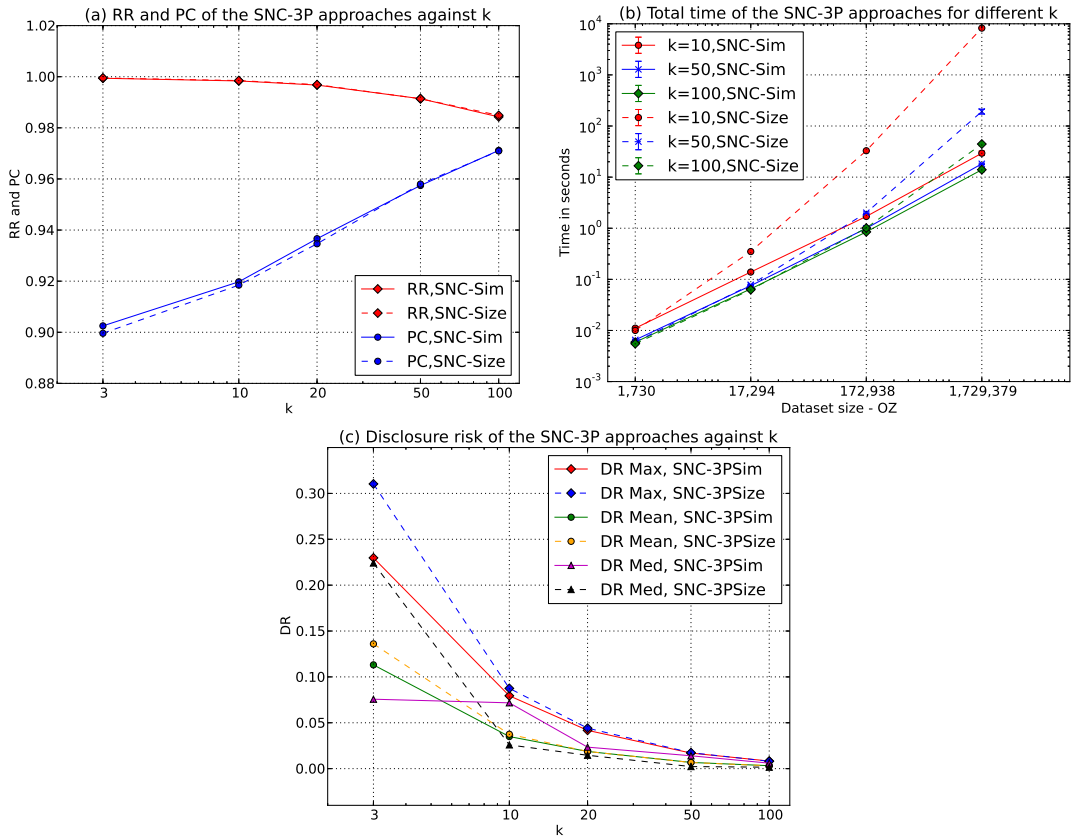


Figure 6.9: (a) Reduction ratio (RR) and pairs completeness (PC) values for the SNC-3P blocking approaches on the OZ-17,294 Mod dataset, (b) total blocking time against the dataset size averaged over the results of all variations of each dataset, and (c) disclosure risk values on the OZ-17,294 Mod dataset, for different values of k.

6.6 Summary

In this chapter, we have proposed an efficient three-party private blocking technique that can be used to make privacy-preserving record linkage applications scalable to large databases. Our method is based on the sorted nearest neighborhood clustering approach, and uses a combination of the privacy techniques reference values and k-anonymous mapping. Experiments conducted on large real-world databases validate that our approach is scalable to large databases and effective in generating quality candidate record pairs while preserving k-anonymous privacy characteristics.

However, as discussed earlier, three-party solutions are often susceptible to collusion between parties. In the next chapter, we aim to study how the sorted neighborhood clustering can be used for private blocking in a two-party context.

Chapter 7

Two-Party Private Blocking

As discussed in the previous chapters, one main threat with three-party solutions is the possibility of collusion between parties to identify the private data of another party. In this chapter, we introduce a novel two-party private blocking technique for privacy-preserving record linkage (PPRL) based on the efficient sorted neighborhood clustering, as will be described in Section 7.2. Similar to the SNC-3P private blocking approach proposed in the previous chapter, privacy is addressed by k-anonymous mapping and public reference values. We analyze our two-party solution in Section 7.3 and empirically evaluate in Section 7.5 based on a linkage attack presented in Section 7.4. Finally we summarize our work in Section 7.6.

7.1 Introduction

Private blocking aims to generate candidate record pairs from two databases without revealing any sensitive information that can be used to infer individual records and their attribute values in the databases. As discussed in Chapter 6, the sorted neighborhood approach is considered to be a very efficient technique compared to other blocking techniques in terms of the number of candidate record pairs generated [52].

As reviewed in Chapter 3, majority of the proposed private blocking solutions require a trusted third party to perform the blocking. Such three-party solutions are often not reliable due to the privacy risk of collusion between the third party and one of the database owners with the aim to learn about the other database owner's private or confidential information. We therefore propose an efficient two-party private blocking technique based on the sorted neighborhood approach using a combination of two privacy techniques: k-anonymous mapping [72, 182] and public reference values [154].

7.2 Proposed Solution

As we did in the previous chapter, we assume again two database owners, *Alice* and *Bob*, with databases $\mathbf{D}^A$ and $\mathbf{D}^B$, participate in the protocol to perform private blocking on their databases. *Alice* and *Bob* have access to a publicly available reference

Table 7.1: Notation used in this chapter.

$\mathbf{D}^A$, $\mathbf{D}^B$	Databases held by database owners *Alice* and *Bob*, respectively
$\mathbf{R}$	Publicly available reference dataset
$\mathbf{R}^A$, $\mathbf{R}^B$	Lists of reference values independently selected from $\mathbf{R}$ by *Alice* and *Bob*, respectively ($\mathbf{R}^A \neq \mathbf{R}^B$)
k, w	Minimum number of records in a block, and size of the window
n^A, n^B	Number of records in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
n_R^A, n_R^B	Number of reference values in $\mathbf{R}^A$ and $\mathbf{R}^B$ used by *Alice* and *Bob*, respectively
n_R, n	Total number of reference values used, and number of records in databases
n_r, n_e	Number of reference values in each block, and number of reference values exchanged from each block
A_i, B_i	The i^{th} k-anonymous block of *Alice* and *Bob*, respectively
W_i	The i^{th} window created by the sliding window
v_i	Sorting key value (SKV) of i^{th} record in the databases
r_i	Reference value in the i^{th} position in the sorted reference list
$sim(\cdot,\cdot)$	Function used to calculate similarities between two reference values r_i and r_j, $(0 \leq sim(\cdot,\cdot) \leq 1)$
s_t	Minimum similarity threshold value to determine a pair of values as similar, $(0 \leq s_t \leq 1)$

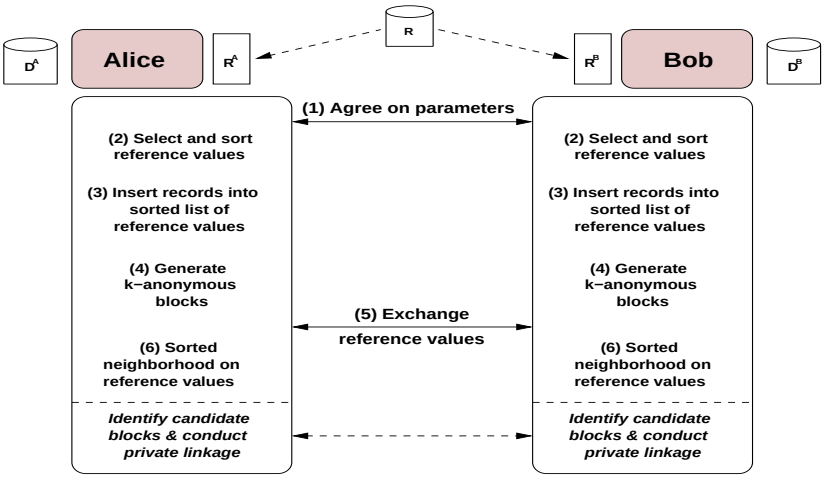


Figure 7.1: Outline of the proposed two-party private blocking protocol (taken from [192]). The numbers given correspond to the protocol steps explained in Section 7.2.1.

dataset $\mathbf{R}$ that contains reference values in the same domain as the values used for the sorting key. All values in $\mathbf{D}^A$ and $\mathbf{D}^B$ are sensitive, and only $\mathbf{R}$ is shared between *Alice* and *Bob*. The aim of this approach is to identify candidate record pairs from $\mathbf{D}^A$ and $\mathbf{D}^B$ by using the values in $\mathbf{R}$ without revealing any information about the sensitive values in $\mathbf{D}^A$ and $\mathbf{D}^B$. Figure 7.1 illustrates the two-party setting and the outline of our solution, and Table 7.1 summarizes the notation we use in this chapter.

Initially local blocks are independently generated by *Alice* and *Bob* using sorted neighborhood-based k-anonymous clustering. Each of the generated k-anonymous blocks contains at least k database records and n_r reference values ($n_r \geq 1$). To identify the candidate blocks from both databases a certain number of reference values (n_e) from each block are exchanged between *Alice* and *Bob*. These exchanged reference values represent the sorting key values (SKVs) of the records in the corresponding blocks. The sorted nearest neighborhood approach is applied on the list of exchanged reference values from both databases, to find the candidate blocks from the reference values that fall into the same window.

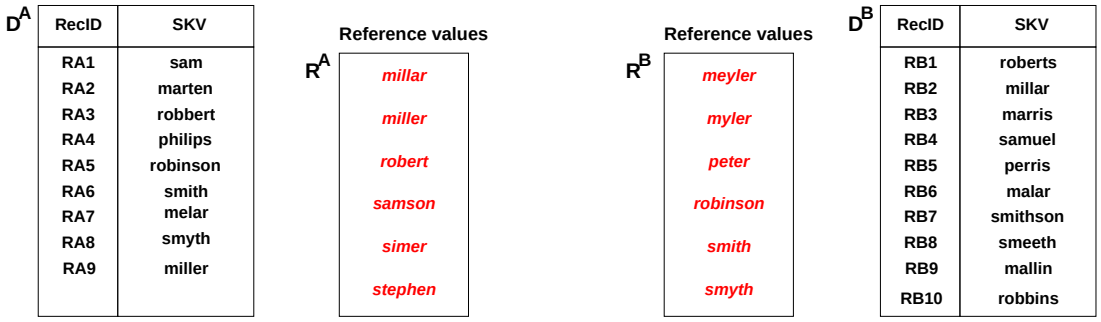


Figure 7.2: Example databases held by *Alice* ($\mathbf{D}^A$) and *Bob* ($\mathbf{D}^B$) with SKVs based on surname attribute, and the lists of reference values ($\mathbf{R}^A$ and $\mathbf{R}^B$) for *Alice* and *Bob*, respectively, used to illustrate the protocol described in Section 7.2.1.

7.2.1 Protocol Description

In this section we describe the steps involved in our two-party sorted neighborhood clustering (SNC)-based private blocking approach, and illustrate the protocol with an example consisting of two small databases, as shown in Figure 7.2. The protocol performs the following steps, as illustrated in Figures 7.2 to 7.4 (taken from [192]):

1. The first step is for the database owners, *Alice* and *Bob*, to agree upon the attributes to be used as the sorting key, the minimum number of elements in a block k, the size of the window w, a similarity function $sim(\cdot,\cdot)$ to compare reference values ($0 \leq sim(\cdot,\cdot) \leq 1$), and the minimum similarity threshold s_t which defines if two blocks should be merged or not. The $sim(\cdot,\cdot)$ function used here is an approximate string comparison function [29] that calculates how similar two reference values (which are assumed to be strings) are.

2. *Alice* and *Bob* each individually selects and sorts a certain number of reference values (n_R) from the reference dataset $\mathbf{R}$. We will provide details of this selection process in Section 7.2.2. The value for n_R can be chosen as $n_R = n/k * n_r$, where n is the number of records in the database to be blocked, so that each block will roughly contain n_r reference values. Since *Alice* and *Bob* do this step independently, they will end up with different lists of sorted reference values ($\mathbf{R}^A$ and $\mathbf{R}^B$, respectively). Due to the selection process, some reference values might occur in both $\mathbf{R}^A$ and $\mathbf{R}^B$.

3. *Alice* and *Bob* then individually insert their database records based on the records' SKVs into their sorted list of reference values. This step generates SNC-based blocks that contain one reference value in each block and it's corresponding list of SKVs which are lexicographically sorted before the reference value (see SNC blocks in Figure 7.3). An inverted index data structure can be used to perform this blocking efficiently.

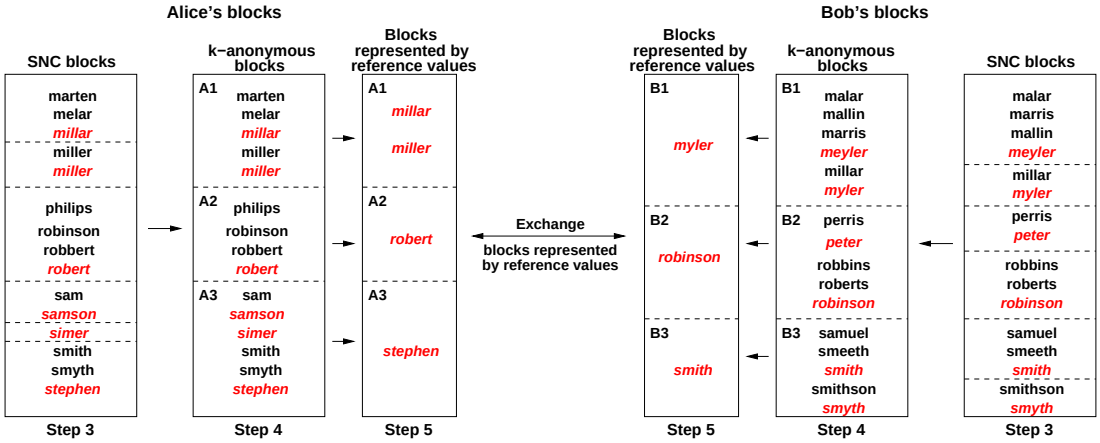


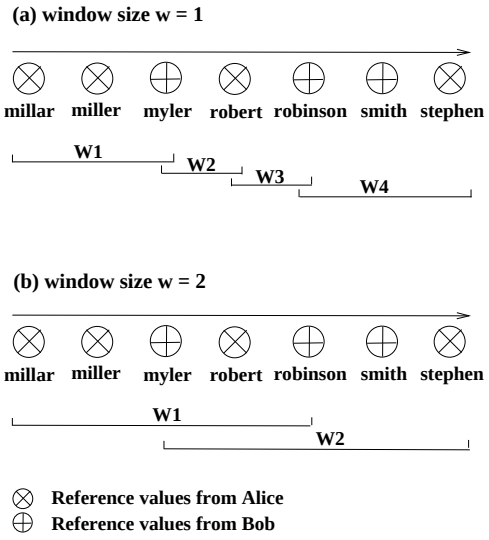
Figure 7.3: Insertion of SKVs into the list of sorted reference values to generate SNC blocks (protocol step 3) and merging of blocks to generate k-anonymous blocks (protocol step 4), where each block is represented by one or several reference values and contains the SKVs that are sorted near the reference values. The reference values that are selected to represent each block are then exchanged between *Alice* and *Bob* (protocol step 5). In this example, $k = 3$ and $s_t = 0.8$. Reference values are shown in italic font.

Algorithm 7.1 : Generating k-anonymous blocks (protocol step 4)

Input: $\mathbf{R}'$: List of sorted reference values $[r_1, \ldots, r_{n_R}]$
$\mathbf{S}$: Set of SNC blocks $(r_1 : [v_1, \ldots, v_l])$
k: Minimum number of elements in a block
s_t: Minimum similarity threshold
$sim(\cdot, \cdot)$: Similarity comparison function
Output: $\mathbf{O}$: Set of k-anonymous blocks $((r_1, \ldots, r_{n_r}) : [v_1, \ldots, v_k])$

```
1:  i = 0
2:  while i < n_R do
3:      block_vals = []; ref_vals = []
4:      num_vals = 0; sim_val = 0.0; j = 0
5:      while (num_vals ≤ k and i + j < n_R) or
                (sim_val ≥ s_t and i + j < n_R) do
6:          r_i = R'[i + j]; c = S[r_{i+j}]
7:          num_vals += len(c)
8:          block_vals += c
9:          sim_val = sim(r_{i+j}, r_{i+j+1})
10:         ref_vals += r_i
11:         j += 1
12:     end while
13:     O[(ref_vals )] = block_vals
14:     i += j
15: end while
```

4. The next step is to merge the SNC blocks such that each block contains at least k database records. This provides k-anonymous privacy characteristics, as each record in the database can be seen as similar to at least $k - 1$ other records. Algorithm 7.1 (which is executed independently by *Alice* and *Bob*) shows the main steps involved in the merging of SNC blocks to create k-anonymous blocks. The k-anonymous blocks are generated by merging the SNC blocks until the number of records in the blocks becomes (or is) at least k (lines 5-12). The similarity



window	Alice's reference values	Bob's reference values	Candidate blocks
W1	millar, miller	myler	(A1,B1)
W2	robert	myler	(A2,B1)
W3	robert	robinson	(A2,B2)
W4	stephen	robinson	(A3,B2)
		smith	(A3,B3)

window	Alice's reference values	Bob's reference values	Candidate blocks
W1	millar, miller	myler	(A1,B1)
	robert	robinson	(A1,B2)
			(A2,B1)
			(A2,B2)
W2	robert	myler	(A2,B3)
	stephen	robinson	(A3,B1)
		smith	(A3,B2)
			(A3,B3)

Figure 7.4: The sorted nearest neighborhood approach on the exchanged reference values using a sliding window of size (a) $w = 1$ and (b) $w = 2$, and (on the right side) their corresponding candidate blocks generated (protocol step 6).

between reference values can also be considered using the $sim(\cdot,\cdot)$ function. If the similarity between two reference values in two different blocks is greater than or equal to the minimum similarity threshold value s_t, then the two blocks are merged together. This reduces the chances of missing true candidate record pairs that have similar SKVs [52, 211]. The first block of Bob (B1) in Figure 7.3 is an example for this similarity-based merging. Though the size of the block represented by the reference value 'meyler' is equal to $k = 3$, the next block represented by the reference value 'myler' is merged with this block since the similarity between these two reference values is greater than or equal to s_t ($sim('meyler','myler') = 0.9 \geq s_t$, where $s_t = 0.8$).

5. Once the k-anonymous blocks are generated, reference values corresponding to each block need to be exchanged between *Alice* and *Bob*. These reference values represent each block in their databases. The number of reference values (n_e) exchanged from each block can be 1 or more ($n_e \geq 1$). The privacy of the protocol depends on the number of reference values that are exchanged. The larger this number from each block is, the higher the accuracy but the lower the privacy. This is discussed in more detail in Section 7.3.

6. Using the exchanged reference values the candidate blocks from *Alice* and *Bob* can be identified to generate candidate record pairs. The sorted nearest neighborhood approach is used to achieve this goal, as explained in Algorithm 7.2. The reference values from *Alice* and *Bob* are merged and sorted first (line 1) and then the sorted nearest neighborhood method is applied on the sorted list of reference values using a sliding window of size w to identify the candidate

reference values that fall in the same window (lines 3-13). The value for w represents the minimum number of reference values that must be included in the window from each database owner. The blocks represented by these candidate reference values are determined as candidate blocks (lines 14-18). This process is illustrated in Figure 7.4 for $w = 1$ and $w = 2$.

Algorithm 7.2 : Generating candidate blocks (protocol step 6)

Input: $\mathbf{R}^A$: List of Alice's reference values $[r_1, \ldots, r_{n_A}]$
$\mathbf{R}^B$: List of Bob's reference values $[r_1, \ldots, r_{n_B}]$
w: Size of the window
Output: $\mathbf{C}$: Candidate blocks $((r_1^A, r_1^B), \ldots, (r_l^A, r_l^B))$

```
1: R = R^A ∪ R^B; sort(R)
2: i = 0
3: while i < len(R) do
4:     alice_refs = []; bob_refs = []
5:     j = 0
6:     while (len(alice_refs) ≤ w and i + j < len(R)) or
           (len(bob_refs) ≤ w and i + j < len(R)) do
7:         if R[i + j] ∈ R^A then
8:             alice_refs += R[i + j]
9:         else if R[i + j] ∈ R^B then
10:            bob_refs += R[i + j]
11:        end if
12:        j += 1
13:    end while
14:    for r^A ∈ alice_refs do
15:        for r^B ∈ bob_refs do
16:            C += (r^A, r^B)
17:        end for
18:    end for
19:    i+= j
20: end while
```

The candidate record pairs are generated from all the records in the corresponding candidate blocks of *Alice* and *Bob*. For example, the candidate pair of blocks (A3, B3) in Figure 7.4 generates the following candidate record pairs: (RA1, RB4), (RA1, RB7), (RA1, RB8), (RA6, RB4), (RA6, RB7), (RA6, RB8), (RA8, RB4), (RA8, RB7), and (RA8, RB8). A private matching and classification technique [56, 154, 174] (which is outside and independent of our protocol) can then be applied on each resulting candidate block individually to obtain the detailed similarities of individual record pairs (as will be proposed in Chapters 8 and 9).

7.2.2 Selecting Reference Values

Reference values are used in our approach as a privacy technique to conduct private blocking between two sensitive databases. Such reference values can be constructed either with random faked values, or values extracted from a public reference dataset, for example, all unique surnames taken from a public telephone directory or electoral roll (such as the NC voter dataset [31], as was described in Section 5.4). The aim of this approach is to find the candidate blocks using the reference values instead of the actual values in the databases.

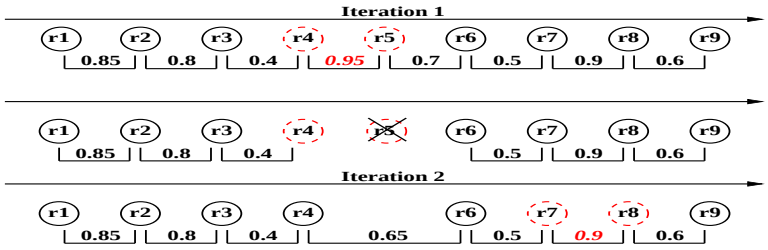


Figure 7.5: The reference values selection method proposed in Section 7.2.2. A value in the highest similarity pair is removed in each iteration according to its similarity with neighbors.

The list of reference values used should be effective in blocking the databases by placing similar SKVs into the same blocks and different SKVs in different blocks. In addition, the reference values exchanged should not be close to each other in the lexicographically sorted list of values in a public global dataset, **G**, as this might reduce the *k*-anonymous privacy protection, as will be discussed in Section 7.3. This means that the reference values selected from **R** need to be evenly spread to represent all blocks in the databases.

We propose a reference values selection method to select and exchange appropriate reference values that are not similar/close to each other and represent all blocks. The similarity or closeness of reference values can be calculated by using an extended version of the Dice-coefficient similarity function [29] to compare only the first few characters of the strings (we name this as 'Dice-short'). We only compare the first few characters of the reference value strings as the reference values are sorted in lexicographical order. Assume r_i and r_j are two reference values and r_i is 10 characters long and r_j is 12 characters long. We only compare the first $min(r_i, r_j)/2 = 5$ characters of r_i and r_j to check if these two strings are not lexicographically sorted close to each other.

Algorithm 7.3 : Selecting reference values

Input: $\mathbf{R}'$: List of reference values $[r_1, \ldots, r_{n_{R'}}]$
n_R: Number of reference values, $n_R < n'_R$ $(n'_R = |R'|)$
$sim(\cdot,\cdot)$: Dice-short similarity comparison function
Output: $\mathbf{R}'$: List of selected reference values $[r_1, \ldots, r_{n_R}]$

```
1: sort(R'); sim_pairs = {}
2: for i ∈ range(len(R') − 1) do
3:     sim_pairs[(R'[i], R'[i + 1])]=sim(R'[i], R'[i + 1])
4: end for
5: while len(R') ≥ n_R do
6:     [r_i, r_j] = max(sim_pairs)
7:     i = R'.getIndex(r_i); j = R'.getIndex(r_j)
8:     if sim(R'[i − 1], r_i) ≥ sim(R'[j + 1], r_j) then
9:         R'.remove(r_i)
10:    else
11:        R'.remove(r_j)
12:    end if
13: end while
```

The proposed selection method is explained in Algorithm 7.3. This is run by *Alice* and *Bob* independently in Step 2 of the protocol. The process starts by initially selecting more than n_R reference values from **R** into $\mathbf{R}'$. Each pair of consecutive reference

values in the sorted list of $\mathbf{R}'$ are then compared using the Dice-short similarity function. Pruning of similar reference values from $\mathbf{R}'$ is conducted in an iterative way, such that one of the reference values in the closest or most similar pair is removed at each iteration, depending on the similarity between their neighboring reference values, until the number of reference values in $\mathbf{R}'$ becomes n_R. Figure 7.5 illustrates this iterative pruning of similar reference values.

Further, the reference values of *Alice* and *Bob* that are sorted next to each other in a window (in step 6 of our protocol) should be similar or close to each other in order to be considered as true candidate blocks. Again, the Dice-short similarity function can be used to calculate how similar *Alice*'s and *Bob*'s reference values in a window are, to determine the corresponding blocks as candidate blocks.

7.3 Analysis of the Protocol

In this section we analyze our SNC-2P protocol in terms of complexity, privacy, and quality of blocking.

7.3.1 Complexity

Assume the number of records in both databases is n ($n = n^A = n^B$) and n_R reference values are selected by both *Alice* and *Bob* from the reference dataset $\mathbf{R}$ ($n_R = n_R^A = n_R^B$). Sorting these n_R reference values is of $O(n_R\ log\ n_R)$ complexity, and inserting the n database records into their sorted list of reference values is of $O(n\ log\ n_R)$ complexity. Insertion of records into the sorted list of reference values in protocol step 3 results in n_R SNC blocks each represented by one reference value. Merging the SNC blocks to create k-anonymous blocks in step 4 of the protocol requires a loop over n_R blocks, which is of $O(n_R)$ complexity, and results in an average of (n/k) k-anonymous blocks each represented by n_r reference values ($n_r \geq 1$) and each containing a minimum of k database records.

Alice and *Bob* then exchange n_e reference values ($1 \leq n_e \leq n_r$) from each of their k-anonymous blocks (a total of n_R reference values) in protocol step 5. The communication complexity of this step is therefore $O(n_R)$. In protocol step 6, sorting the exchanged reference values is of $O(2n_R\ log\ 2n_R)$ complexity and applying the sorted nearest neighborhood approach on this sorted list of reference values is of $O(2n_R)$ complexity. The overall complexity of our approach is linear in the size of the databases n and log-linear in the number of reference values n_R used. Assuming each block contains k records and the size of the sliding window (minimum number of reference values a window comprises from each database owner) is w, the number of candidate record pairs generated by our approach is $(n/k) \times (k^2 \times w) = n\ k\ w$.

7.3.2 Privacy

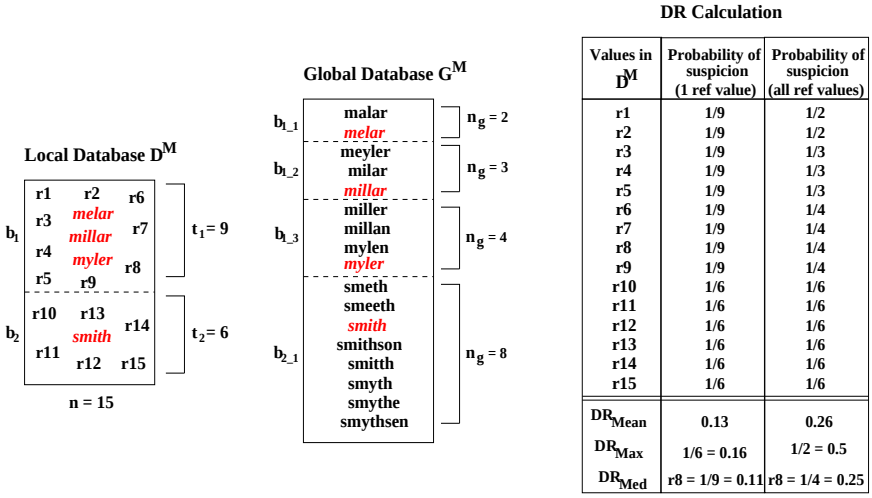
We assume that both *Alice* and *Bob* follow the honest but curious (HBC) adversarial model [78], in that they follow the protocol while trying to find out as much as

possible about the data from the other party. To analyze the privacy of the protocol, we need to evaluate what can be learned from the data they communicate with each other during the protocol. In step 5 of our protocol, *Alice* and *Bob* exchange a certain number of reference values (n_e) from each block (no sensitive actual values in the databases are exchanged), and this might leak some information regarding the blocks in their databases depending on the value for n_e.

- **Case 1: $\mathbf{n_e = 1}$**: Since each block consists of at least k records, revealing only one reference value from each block guarantees k-anonymous privacy (similar to the SNC-3P approaches proposed in Chapter 6). This type of privacy has successfully been used in previous private blocking solutions [91, 104]. k-anonymous mapping makes it difficult to perform a frequency attack to infer individual records in the blocks. The value for k has a trade-off between privacy and computational complexity. A higher value for k provides stronger privacy guarantees but more candidate record pairs will be generated. One drawback of representing a block by one reference value only is the possible loss of some true candidate blocks when applying the sorted nearest neighborhood method on the exchanged reference values in step 6 of our protocol, because a single reference value is not sufficient to represent all the SKVs in a block.

- **Case 2: $\mathbf{n_e > 1}$**: If several reference values are exchanged from a block then k-anonymous privacy is not guaranteed. Conducting a frequency linkage attack (as will be described in Section 7.4) using a global dataset $\mathbf{G}$ with known values in the same domain as used for the sorting key can reveal frequency information for the reference values exchanged from the blocks. This information can be used by an adversary to infer the frequency distribution of the sensitive SKVs in the corresponding blocks. Assume a block is represented by reference values $r = r_1, \ldots, r_e$ and their frequency distribution of individual block sizes in $\mathbf{G}$ is learned as $\mathbf{f} = f_1, \ldots, f_e$. Revealing only one reference value (r_i) discloses that there are k records sorted near r_i, and thus the disclosure risk is $1/k$. But revealing several reference values discloses more information, namely that there are $f_i \times k / \sum f_i$ records sorted near reference value r_i, $i = 1 \ldots e$. This reduces the k-anonymous privacy to $min(\mathbf{f}) \times k / \sum f_i$. For example, if three reference values, r_1, r_2 and r_3, in a block are exchanged and their individual size frequency distribution of blocks in $\mathbf{G}$ is $f_1 = 1$, $f_2 = 3$, and $f_3 = 4$, then this reveals that there are $k/8$ records sorted near r_1, $3k/8$ near r_2, and $4k/8$ near r_3. Disclosure risk (maximum) is increased to $1/(k/8) = 8/k$ with the $k/8$ records sorted near r_1 from $1/k$ in case 1. Privacy is therefore reduced to $k/8$.

Therefore, a larger number of reference values exchanged from each block will reduce the privacy of the protocol. At the end of private blocking, candidate blocks are found and private linkage can be conducted on each block pair individually by using a private matching and classification technique [56, 154, 174], which should not reveal any sensitive information (this step is outside of our protocol).



DR Calculation

Values in D^M	Probability of suspicion (1 ref value)	Probability of suspicion (all ref values)
r1	1/9	1/2
r2	1/9	1/2
r3	1/9	1/3
r4	1/9	1/3
r5	1/9	1/3
r6	1/9	1/4
r7	1/9	1/4
r8	1/9	1/4
r9	1/9	1/4
r10	1/6	1/6
r11	1/6	1/6
r12	1/6	1/6
r13	1/6	1/6
r14	1/6	1/6
r15	1/6	1/6
DR_{Mean}	0.13	0.26
DR_{Max}	1/6 = 0.16	1/2 = 0.5
DR_{Med}	r8 = 1/9 = 0.11	r8 = 1/4 = 0.25

Figure 7.6: An attack method for the SNC-2P private blocking solution. Disclosure risk is less when only one reference value is exchanged compared to when several reference values are exchanged from each block. For example, $P_s = 1/9$ for $r_1, r_2, \cdots, r_9$ in block b_1 when one reference value is disclosed, because each record is consistent with 9 records that are in the same block. If all three reference values ('melar', 'millar', and 'myler' that are assumed to have the frequency distribution of individual block sizes of $2:3:4$ in **G**) are exchanged, then $P_s = 1/2$ for two records, $P_s = 1/3$ for three records, and $P_s = 1/4$ for four records in the block b_1.

7.3.3 Quality

The quality of blocking is defined in terms of effectiveness measured by pairs completeness (PC), i.e., all similar records should be grouped into the same block which generates candidate record pairs that include all true matching record pairs, and efficiency measured by reduction ratio (RR), i.e., the number of candidate record pairs generated should be as small as possible [30].

The size of the window w plays a major role in deciding the quality of blocking. A higher value for w is more likely to group more nearest blocks as candidate blocks. This results in higher PC and lower RR. The value for k also determines the PC and RR of blocking. A higher value for k results in higher PC but lower RR. Optimal k and w need to be set such that high values for both PC and RR are achieved while k guarantees sufficient privacy as well.

7.4 Linkage Attack

In this section we present the frequency linkage attack using an external global dataset **G** for privacy evaluation of our SNC-2P solution based on the evaluation model proposed in Chapter 5. A private blocking protocol that reveals more information than the number of blocks (n_B) and their sizes ($t_i = |b_i|, 1 \leq i \leq n_B$) during the protocol, will provide more information on the distribution of blocks and their val-

ues. Our SNC-2P protocol reveals reference values from each block, and as discussed in Section 7.3.2, the more reference values are exchanged from a block between the database owners the more information is disclosed about the block, as illustrated in Figure 7.6. For example, if three reference values, 'melar', 'millar', and 'myler', in block b_1 that contains nine records ($t_1 = 9$), are exchanged and their size frequency distribution of individual blocks in **G** is 'melar' (b_{1_1}) = 2, 'millar' (b_{1_2}) = 3, and 'myler' (b_{1_3}) = 4, then this reveals that there are $2t_1/9 = 2$ records sorted near 'melar', $3t_1/9 = 3$ near 'millar', and $4t_1/9 = 4$ near 'myler'. The minimum block size now becomes 2 with the two records sorted near 'melar', and the maximum disclosure risk is therefore increased to $DR_{Max} = 1/2$ from $DR_{Max} = 1/t_1 = 1/9$, i.e., when only one of three reference values is exchanged from b_1.

7.5 Experimental Evaluation

We conducted an empirical study of our SNC-2P approach on the datasets described in Section 5.4 using the evaluation framework proposed in Chapter 5. The default parameters were set as $k = 100$, $s_t = 0.8$, $w = 2$, and $n_e = 50\%$ (this setting gives best results in terms of all three properties of PPRL, as will be shown in Figures 7.10, 7.11, and 7.12). n_R was set to number of records$/k \times 10$ so that each block will roughly contain $n_r = 10$ reference values. Different values for k, w, and n_e were also used to evaluate the performance of the SNC-2P blocking against k, w, and n_e, respectively. A combination of two attributes was used as the sorting key: given name / first name, and surname / last name.

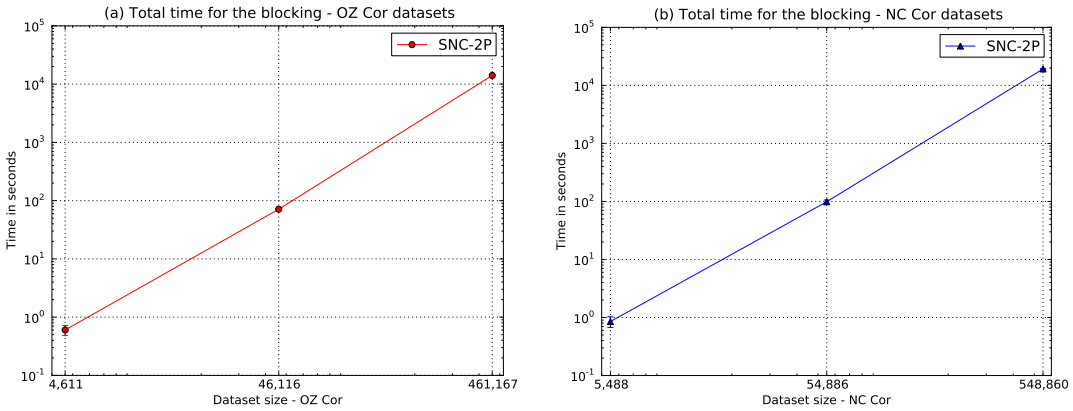


Figure 7.7: Total time of the SNC-2P blocking approach on the (a) OZ Cor datasets, and (b) NC Cor datasets, averaged over the results of all variations of each dataset.

Figure 7.7 shows the total blocking time required for private blocking of the SNC-2P approach on the OZ Cor and NC Cor datasets. As can be seen from the figure, the SNC-2P approach has an almost linear complexity in the size of the databases and is scalable to large databases.

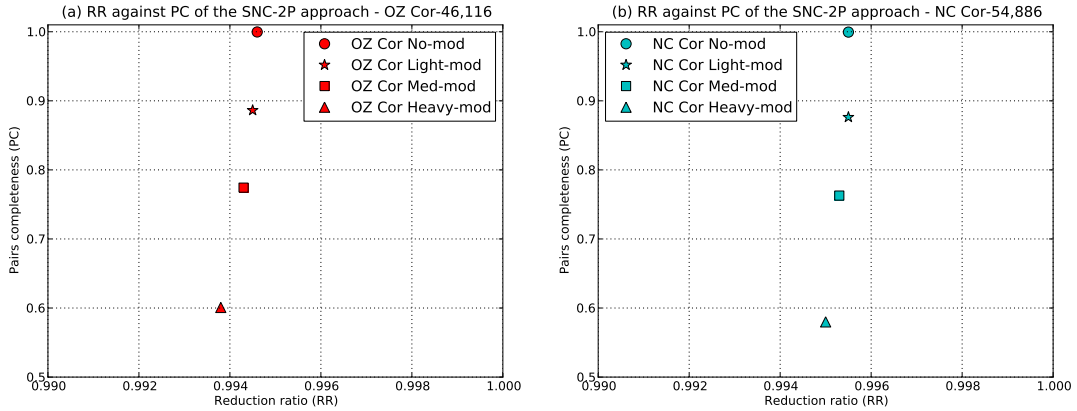


Figure 7.8: A comparison of reduction ratio (RR) against pairs completeness (PC) of the SNC-2P solution on the (a) OZ Cor-46,116 and (b) NC Cor-54,886 datasets with No-mod, Light-mod, Med-mod, and Heavy-mod variations.

A comparison of RR and PC of the SNC-2P blocking is presented in Figure 7.8 on the OZ Cor-46,116 and NC Cor-54,886 datasets with different levels of data modifications (corruptions). Different levels of corruptions applied to the datasets (as was described in Section 5.4) allow to evaluate the performance of approximate matching of our protocol in the presence of data errors. As shown in the figure, the SNC-2P approach achieves high RR and PC when no modification is applied to the datasets, and then the values decrease as the level of modifications increases.

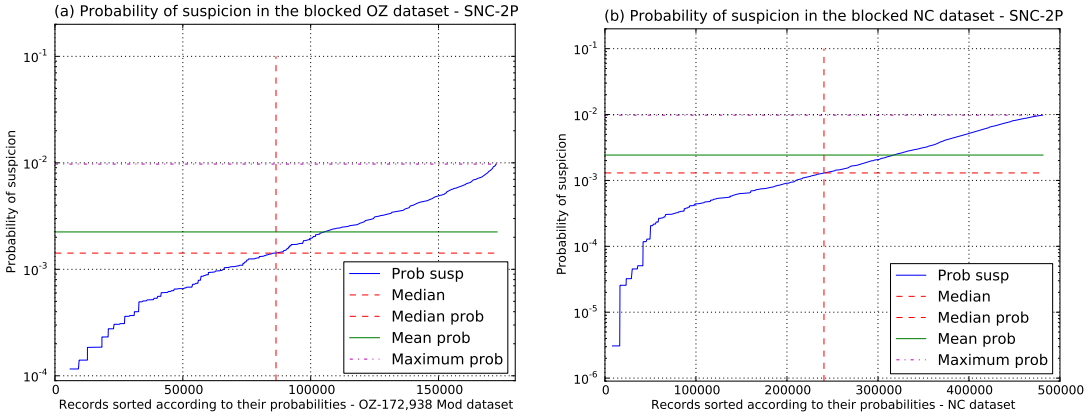


Figure 7.9: A comparison of probability of suspicion (P_s) values of the blocked datasets generated by the SNC-2P approach and the calculated disclosure risk (DR) measures on the (a) OZ-172,938 Mod, and (b) NC datasets.

We then present the distribution of P_s values in the masked and blocked OZ-172,938 Mod and NC datasets by the SNC-2P blocking solution, and the DR measures (proposed in Section 5.3.1) calculated for the frequency linkage attack (described in

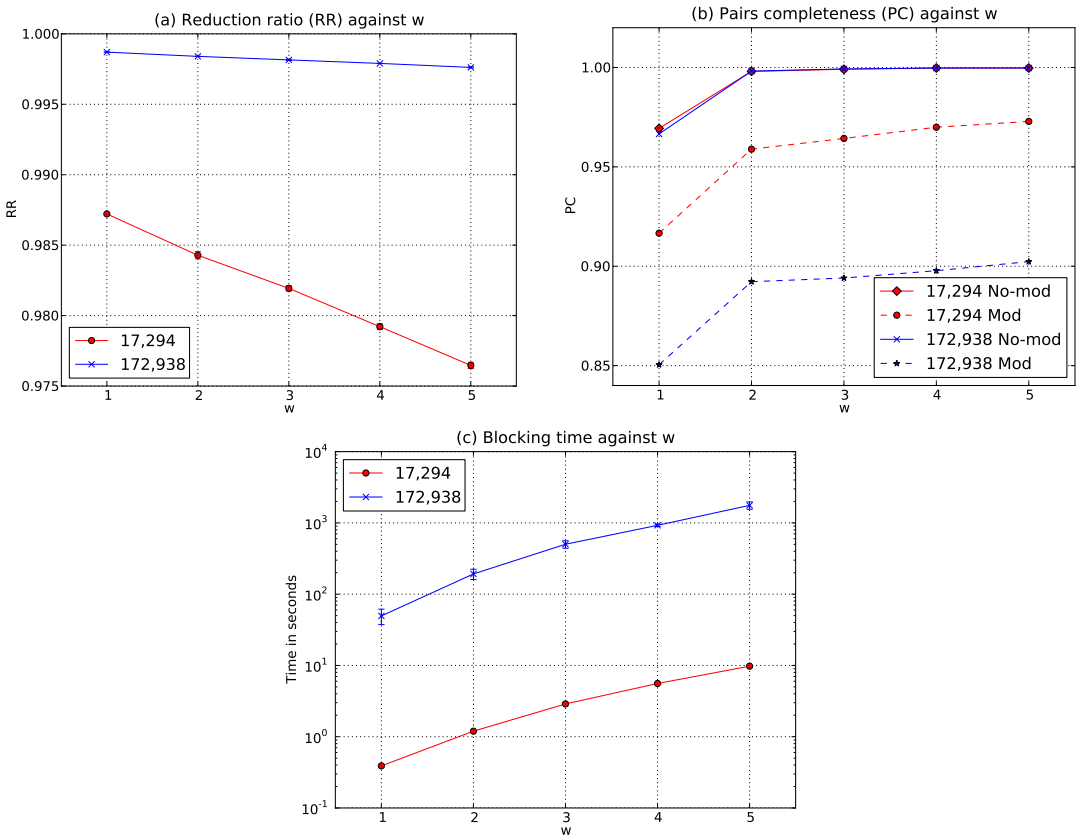


Figure 7.10: (a) Reduction ratio (RR) (b) Pairs completeness (PC) and (c) total blocking time of the SNC-2P approach on the OZ-17,294 and OZ-172,938 datasets, with different values for window size (w).

Section 6.4). The lower DR results achieved by the SNC-2P blocking approach, as illustrated in Figure 7.9, show the privacy aspects of our approach.

Figure 7.10 shows RR, PC, and total time required for blocking (averaged over the results of both database owners over all variations of each dataset) of the SNC-2P blocking approach with different window sizes w. As expected, PC and time for blocking increase with w while RR decreases. This is because when w increases more candidate blocks will be generated which results in more candidate record pairs. Hence, the efficiency of blocking (evaluated by RR and blocking time) decreases with w while effectiveness (evaluated by PC) increases, as discussed in Section 7.3.3. Since there is a drastic improvement in PC when $w = 2$ with a smaller increase in blocking time and a smaller decrease in RR, we choose this as the best default parameter setting. With non-modified datasets we achieve high PC (of nearly 1.0) and with modified datasets the value is reduced. As it turns out, there is no difference in RR and blocking time with modified and non-modified datasets, and we therefore only report the averaged results.

In Figure 7.11, we investigate the performance of SNC-2P solution in terms of RR, PC, and total blocking time with different values for the privacy parameter k.

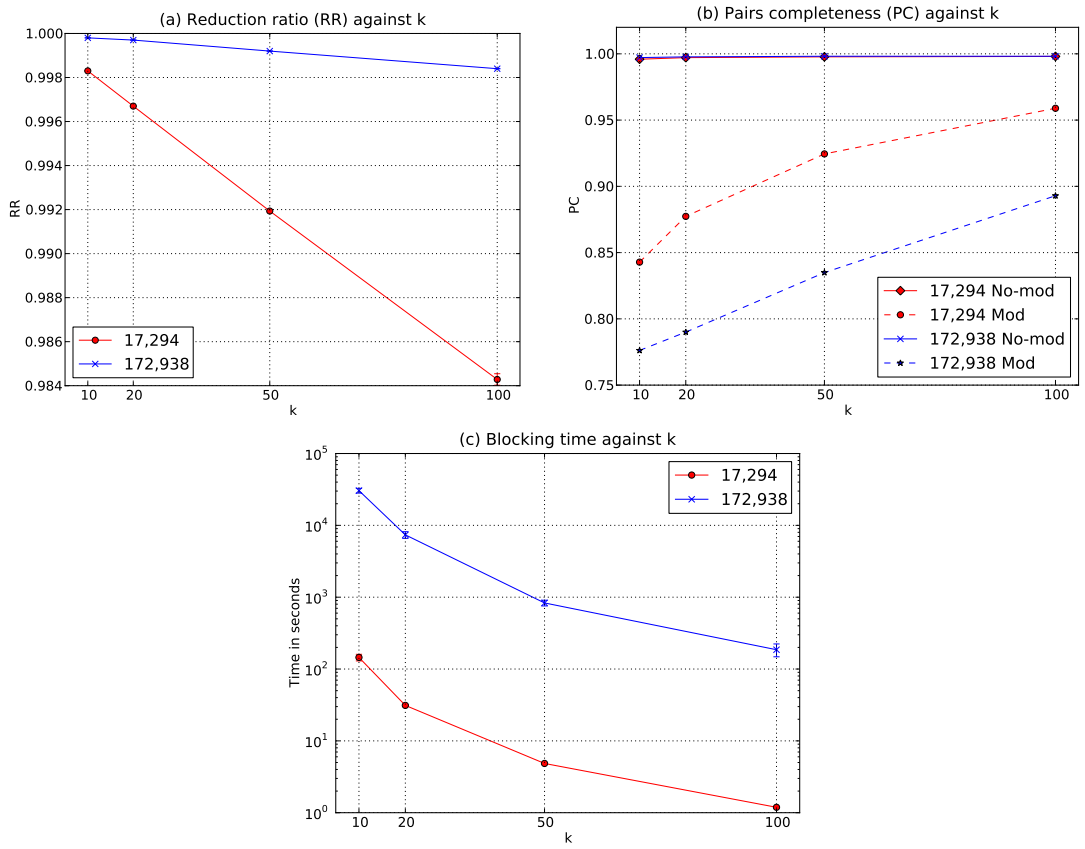


Figure 7.11: (a) Reduction ratio (RR) (b) Pairs completeness (PC) and (c) total blocking time of the SNC-2P approach on the OZ-17,294 and OZ-172,938 datasets, with different values for privacy parameter (k).

As discussed in Section 7.3.3, RR decreases with k while PC increases. As can be seen from the figure, the SNC approach achieves high values for both RR and PC even when $k = 100$. Hence, we choose $k = 100$ as the default parameter setting of our approach. Interestingly, the blocking time reduces with k. This is because the number of resulting blocks (n/k) becomes smaller as k gets larger.

Finally we study how the exchange of reference values determines the scalability, privacy, and quality of blocking in Figure 7.12. Scalability of blocking is measured by total blocking time and RR, and it reduces with the percentage of reference values exchanged (Figures 7.12 (a) and 7.12 (b)). Quality of blocking (measured by PC) significantly increases when the percentage of exchanged reference values (n_e) is increased from 10% to 50% with almost no reduction in RR, as shown in Figure 7.12 (b). However, as discussed in Section 7.3.2, the privacy of the protocol (measured by the DR measures calculated based on the linkage attack given in Section 7.4) becomes lower with more reference values exchanged as the disclosure risk of our approach increases with n_e (see Figures 7.12 (c) and 7.12 (d)). Using the Dice-short reference values selection method, as discussed in Section 7.2.2, increases the privacy of the

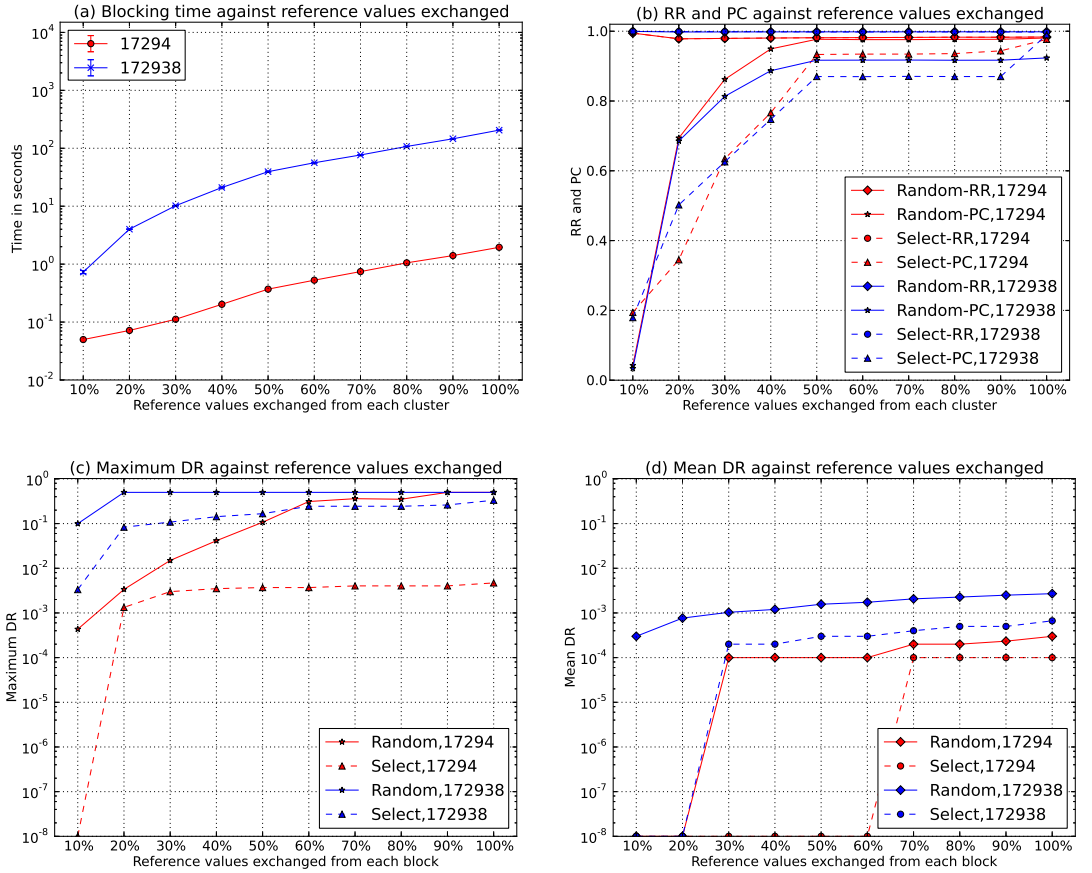


Figure 7.12: (a) Total blocking time (b) reduction ratio (RR) and pairs completeness (PC), (c) maximum disclosure risk values (DR_{Max}), and (d) mean disclosure risk values (DR_{Mean}) of the SNC-2P approach on the OZ-17,294 and OZ-172,938 datasets, when different percentage of reference values are randomly selected ('Random') or appropriately selected using the Dice-short similarity function ('Select'), as described in Section 7.2.2, and exchanged from each block (n_e).

protocol without compromising much the quality of blocking (labeled as 'Select' in the figures). The maximum disclosure risk on the 172,938 dataset is reduced to 0.24 from 0.5 with the proposed reference values selection method when $n_e = 50\%$.

7.6 Summary

In this chapter, we have proposed an efficient two-party private blocking technique based on the sorted neighborhood clustering (SNC) approach that can be used to develop scalable privacy-preserving record linkage (PPRL) applications. Our method uses a combination of two privacy techniques, *k*-anonymous mapping and public reference values. Experiments conducted on real-world datasets validate that our approach is scalable to large databases while being effective in generating quality candidate record pairs and preserving *k*-anonymous privacy characteristics.

As discussed earlier, reference values play a major role in determining the privacy of the protocol and quality of the candidate blocks generated. We introduced one method for selecting appropriate reference values. As future work, we aim to identify other reference values selection methods that can be used to improve the quality of blocking without compromising privacy.

In addition, tackling the problem of finding the optimal values for the SNC-based private blocking approaches including the privacy parameter k and the window size w in the trade-off of the three properties of PPRL using some statistical modelling procedure requires further research.

We also aim to study how the quality of blocking can be improved by running the SNC approach with several iterations using different attributes as sorting keys (similar to the traditional approach [84]) without compromising privacy of the solution. Investigating how other blocking techniques, such as q-gram-based blocking and suffix array-based blocking [29], can be used for private blocking in two-party contexts is another direction for future research.

Chapter 8

Two-Party Reference Values-based Private Matching and Classification

Developing efficient and two-party algorithms for approximate matching and classification in privacy-preserving record linkage (PPRL) has been identified as an important research direction in Chapter 4. In this chapter, we propose a novel two-party protocol for PPRL that addresses the three main properties of PPRL, scalability, privacy, and linkage quality. Our protocol uses the privacy technique of reference values in a two-party setting, as described in Section 8.2. In Section 8.3 we analyze the solution in terms of the three properties, and in Section 8.5 we conduct an empirical study in order to validate these analyses based on a linkage attack proposed in Section 8.4. Finally, we summarize our findings in Section 8.6.

8.1 Introduction

In the absence of unique identifiers for the entities stored in databases, exact or approximate similarity matching techniques are generally applied to the common quasi-identifiers (QIDs, such as name, address and date of birth) for the identification of matching record pairs from different databases [205]. Linking records by comparing the masked QID attribute values with a standard hash-encoding cryptographic technique [41, 78, 135] in a three-party protocol is a naïve solution for private matching and classification [37, 152]. The attribute values match exactly if the corresponding encoded values match, and the third party can identify exactly matching records without knowing the actual attribute values. However, as discussed in Chapter 4, a limitation of this naïve approach is that only exact comparisons of values are possible. A small variation in an attribute value results in a completely different encoded value. In practical applications, the exact matching of QID values is not always sufficient due to variations or typographical and other types of errors in real-world data [83]. Therefore, an approach for approximate matching of values and effective classification in PPRL is required.

There have been several approaches proposed for approximate private matching and classification in PPRL [58, 102, 184, 194]. However, many of these approaches require a trusted third party for linkage which is not always available in a real-

Table 8.1: Notation used in this chapter.

$\mathbf{D}^A$, $\mathbf{D}^B$	Databases held by database owners *Alice* and *Bob*, respectively
RA_i, RB_j	A record in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
A, a	Attributes common to $\mathbf{D}^A$ and $\mathbf{D}^B$ that are used for linking, an attribute $a \in A$
v, v_i, v_j	An individual attribute value
r, r_i, r_j	A reference value
$block_a(\cdot,\cdot)$	Function used to block/index attribute a
b	A blocking key value (BKV): $b = block_a(\cdot)$
c	A compound BKV (CBKV): $c = [block_{a_1}(\cdot), \ldots block_{a_{\lvert A\rvert}}(\cdot)]$
$sim_a(\cdot,\cdot)$	Function used to calculate similarities between values in attribute a
s_t	Minimum similarity threshold to determine a pair of values as similar, $0 \leq s_t \leq 1$
s_m	Minimum similarity threshold to determine the similarity range $[s_m - 1.0]$
k, d	Number of bins, maximum number of bin difference to find the matching bin combinations
$enc(\cdot, h)$	Function and key used to hash-encode values
$\mathbf{BI}_a$, **BLI**	Block Index for attribute $a \in A$, Block List Index
$\mathbf{RLI}_a$, $\mathbf{SI}_a$	Reference List Index for attribute $a \in A$, Similarity Index for attribute $a \in A$
MBC, **MBR**, **MIL**	Matching Bin Combinations, Matching Bins of Records, Match ID List

world application. Generally, most two-party solutions employ secure multi-party computation (SMC)-based privacy techniques (as reviewed in Chapter 3), which are expensive in terms of the computation and communication complexities (and therefore not scalable) while providing stronger privacy guarantees, in order to ensure that the two database owners cannot learn anything from the exchanged data.

In this chapter, we address this problem by proposing a novel two-party private matching and classification solution using efficient perturbation-based privacy techniques. Our protocol is based on (1) the use of reference values that are available to both database owners, and allow them to calculate the similarities independently between their attribute values and the reference values; and (2) the binning of these calculated similarity values to allow their secure exchange between the two database owners. Our protocol also addresses the three main properties of PPRL, scalability, linkage quality, and privacy, which makes it viable in real-world applications.

8.2 Proposed Solution

We first discuss the use of reference values for private matching and classification in a two-party setting in Section 8.2.1, followed with Section 8.2.2 by a step by step description of our protocol.

8.2.1 Reference Values in Two-Party Protocol

The use of reference values has previously been proposed for PPRL in a three-party framework by Pang et al. [154]. Such reference values are assumed to be publicly available and known to both database owners. They can be constructed either by random faked values, or they can be extracted from an external dataset, for example, unique names, postcodes, and suburb names extracted from a public telephone directory. Reference values are used by the database owners to calculate the similarities between their attribute values and the reference values. These similarities are then sent to a third party that can link the records based on the triangular inequality

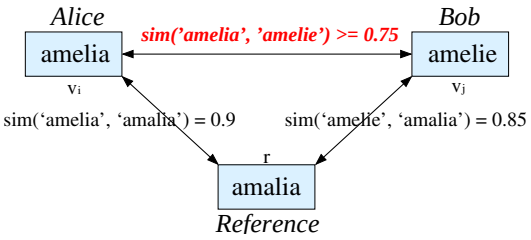


Figure 8.1: Reference value-based similarity calculation in a three-party setting using the triangular inequality property of distance metrics (Adapted from [193]). The actual similarity values calculated with the reference value ($sim(v_i, r)$ and $sim(v_j, r)$) are sent by *Alice* and *Bob* to a third party that can calculate the lower bound of $sim(v_i, v_j)$, as shown in Equation 8.1.

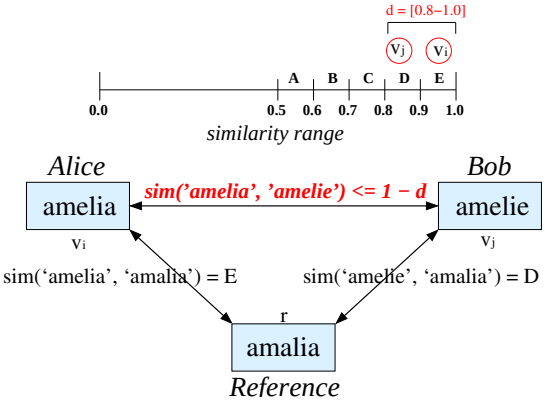


Figure 8.2: Reference value-based similarity calculation in a two-party setting using the reverse triangular inequality property of distance metrics and binning of similarity values. Binned similarity values calculated with the reference value ($sim(v_i, r)$ and $sim(v_j, r)$) are exchanged between *Alice* and *Bob* to calculate the difference d between $sim(v_i, r)$ and $sim(v_j, r)$, as shown in Equation 8.2.

property of the distance metrics, as explained in Equation 8.1. A reference value, r, is the value known to both database owners *Alice* and *Bob*, while the attribute values, v_i and v_j, are the sensitive values that are only known to the corresponding database owner (v_i by *Alice* and v_j by *Bob*).

$$\begin{aligned} dist(v_i, r) + dist(v_j, r) &\geq dist(v_i, v_j) \\ (1 - sim(v_i, r)) + (1 - sim(v_j, r)) &\geq (1 - sim(v_i, v_j)) \\ 1 - sim(v_i, r) - sim(v_j, r) &\geq -sim(v_i, v_j) \\ sim(v_i, r) + sim(v_j, r) - 1 &\leq sim(v_i, v_j) \end{aligned} \tag{8.1}$$

Assume $dist(v_i, v_j)$ is the normalised metric distance between two values v_i and v_j ($0.0 \leq dist(v_i, v_j) \leq 1.0$), and $sim(v_i, v_j) = 1.0 - dist(v_i, v_j)$ is the corresponding similarity between the two values. Similarity values are assumed to be normalised, such that $0.0 \leq sim(v_i, v_j) \leq 1.0$. For an exact match of the two values the similarity function results in $sim(v_i, v_j) = 1.0$ and for two totally different values it results in $sim(v_i, v_j) = 0.0$. A distance-based similarity function mainly holds four properties [79]: non-negativity ($dist(v_i, v_j) \geq 0.0$), identity of indiscernibles ($dist(v_i, v_i) = 0.0$), symmetry ($dist(v_i, v_j) = dist(v_j, v_i)$), and triangular inequality. The triangular inequality property states that the direct distance between two values v_i and v_j is always less than or equal to the combined distance when going through a third value r: $dist(v_i, v_j) \leq dist(v_i, r) + dist(v_i, r)$. A reference value can be used as

a third value (r) to calculate the similarity between the actual attribute values (v_i and v_j). Any distance metric can be used in this approach.

In the three-party approach, the similarities between attribute values and reference values (i.e. $sim(v_i, r)$ and $sim(v_j, r)$) are calculated individually by the database owners, and the results are sent to a third party. The third party can calculate the left hand side (LHS) of Equation 8.1 by calculating the combined similarity value ($sim(v_i, r) + sim(v_j, r) - 1$). The third party then classifies all record pairs as matches that have $sim(v_i, r) + sim(v_j, r) - 1 \geq s_t$, where s_t is a threshold value. If the LHS of Equation 8.1 is at least s_t, then obviously the right hand side (RHS) of the equation, that is the actual similarity value $sim(v_i, v_j)$ between two attribute values v_i and v_j, is also at least s_t and therefore the pair (v_i,v_j) can be classified as a match. This is illustrated in Figure 8.1. However, the results of an empirical evaluation of this approach conducted in [10] show inadequate linkage quality in terms of precision and recall. Increasing the size of the reference list (i.e. using more than one r for similarity calculation of a pair of v_i and v_j) improves the linkage quality to some extent but it leads to long runtime.

In our two-party approach, we use the reverse triangular inequality property of distance metrics, which is illustrated in Equation 8.2, to privately calculate the similarity of two attribute values without exchanging the actual attribute values.

$$\begin{aligned} \big|dist(v_i, r) - dist(v_j, r)\big| &\leq dist(v_i, v_j) \\ \big|(1 - sim(v_i, r)) - (1 - sim(v_j, r))\big| &\leq (1 - sim(v_i, v_j)) \\ \big|-sim(v_i, r) + sim(v_j, r)\big| &\leq (1 - sim(v_i, v_j)) \\ 1 - \big|sim(v_j, r) - sim(v_i, r)\big| &\geq sim(v_i, v_j) \end{aligned} \tag{8.2}$$

From the reverse triangular inequality property, we can see that the value for $sim(v_i, v_j)$ (RHS) becomes higher and gets closer to 1.0 if and only if the values for $sim(v_i, r)$ and $sim(v_j, r)$ (LHS) become equal to each other, with r being an value from the reference list. This implies that if the difference between the similarity values of two values with a value from the reference list is small, then the two values should be similar to each other. We illustrate this approach in Figure 8.2.

The scalability property of the linkage process can be addressed by blocking the records in the databases using a private blocking technique, as proposed in Chapters 6 and 7. There have also been several other private blocking techniques proposed in the literature [3, 56, 103, 104, 124]. For illustrative example, we consider blocking the records based on a (phonetic) encoding function [103], such as Soundex [23]. The database owners then use public reference lists to assign one or several reference values for each block. These reference values are used by the database owners to calculate the similarities between their attribute values and the reference values in each block. The similarity of each attribute value in a block is calculated by comparing the value only with the list of reference values that are in its corresponding block (this improves linkage quality).

Table 8.2: Example bins of similarity range

Bin label	Start range	End range
A	0.5	0.625
B	0.626	0.750
C	0.751	0.875
D	0.876	1.0

Table 8.3: Matching Bin Combinations (MBC)

Match ID	Attribute 1 (Given name)	Attribute 2 (Surname)
1	A,B	A,B
2	A,**B**	B,**C**
3	A,**B**	**C**,D
4	B,C	A,B
5	**B**,C	B,**C**
6	**B,C**	**C,D**
7	C,D	A,B
8	C,D	B,C
9	**C**,D	C,**D**

Once the similarities are calculated, the database owners can conduct the linkage by using a third party that classifies the records based on the reverse triangular inequality of these similarities, as was done by [154]. Since blocking is applied, this will reduce the runtime for linkage and it will be scalable to large databases. This approach provides a scalable three-party solution for approximate matching in PPRL. As with other three-party protocols, privacy is however the major drawback with this three-party approach as well. If one of the database owners colludes with the third party they can learn about the other database owner's private data.

Our aim is to develop a two-party protocol by using public reference lists and the reverse of triangular inequality property of distance metrics for matching and classification. If the calculated similarities can be exchanged between the database owners without revealing any sensitive information, then we can eliminate the need of a third party for the linkage. Since both database owners know the public reference list values, exchanging the calculated similarity values with each other can leak some information about the QIDs to the database owners. Our two-party solution for this problem is by binning the actual similarity values.

We split the similarity range (a possible range from 0.0 to 1.0) into a number of bins k ($k > 1$), and each database owner stores the similarities between their attribute values and the reference values as bin labels into which the calculated similarity values fall. Since we compare the attribute values only with the reference values that are in their corresponding block, the minimum similarity value will be larger than 0.0, and so we only need to bin similarities in an interval $[s_m, 1.0]$, with $s_m > 0.0$ selected by the user. Binning the similarity range from 0.5 to 1.0 into 4 bins, for example, is shown in Table 8.2. We will explain this example in detail further below.

We then calculate the Matching Bin Combinations (MBC) based on the binning distance d. The binning distance determines the maximum number of bin difference we allow for each attribute for the approximate matching of attribute values. For example, if the binning distance is $d = 1$ for each attribute and we use two attributes for the matching (and thus a total binning distance of $d = 2$), then the MBC would be the ones that are given in Table 8.3. Every Matching Bin Combination in MBC is given a unique Match ID (as shown in Table 8.3). Based on the MBC, each database

Table 8.4: Example calculation of bins and matches

	$RA1$ (*Alice*)		$RB1$ (*Bob*)	
	Given name	Surname	Given name	Surname
Attribute values	'millar'	'ameile'	'miller'	'amelia'
Phonetic (block) values	'm460'	'a540'	'm460'	'a540'
Reference values	'myler'	'amalia'	'myler'	'amalia'
Similarity values	0.7	0.8	0.8	0.9
Bin labels	B	C	C	D
Match IDs	{1,2,3,4,5,6}	{2,3,5,6,8,9}	{4,5,6,7,8,9}	{3,6,9}
	{2,3,5,6}		{6,9}	

owner calculates the set of Match IDs to which each of the records in their database corresponds to. Then these Match IDs are exchanged between the database owners. Computing the intersection set of Match IDs and then exchanging the records that are obtained for those common Match IDs between the database owners provide a two-party solution for our problem.

To illustrate our approach, assume we have two records ($RA1$ and $RB1$) in two different databases with their respective values for the attributes Surname and Given name as ('millar','ameile') and ('miller','amelia'), as shown in Table 8.4. Applying the Soundex [23] phonetic-based blocking to these values results in the two blocks 'm460' and 'a540' for Given name and Surname attributes, respectively. Assume that the reference list contains one value for each of these blocks, and they are 'myler' for 'm460' and 'amalia' for 'a540'.

Comparing the attribute values with the corresponding block reference values (using $sim(\cdot,\cdot)$), for example, gives us the similarity values of ($sim('millar','myler') = 0.7$, and $sim('ameile','amalia') = 0.8$) for $RA1$ and ($sim('miller','myler') = 0.8$, and $sim('amelia','amalia') = 0.9$) for $RB1$, which result in the bin combinations (B,C) and (C,D), respectively (see Table 8.2 for the bin ranges). According to the MBC in Table 8.3, the corresponding matches would be Match IDs $\{2,3,5,6\}$ for $RA1$ and Match IDs $\{6,9\}$ for $RB1$, because the bin combination of B for attribute Surname and C for Given name appears in Match IDs 2, 3, 5 and 6, whereas the combination of C and D appears in Match IDs 6 and 9 only (shown in bold font in Table 8.3).

The intersection of these two sets results in Match ID 6, which is considered to be the match combination for these two example records (i.e. the two records' similarity values calculated with the reference value for the Surname attribute are in the interval of $[B-C] = [0.626-0.875]$, and for the Given name attribute they are in the interval of $[C-D] = [0.751-1.0]$), and so the two records can be classified as a match (with a minimum total similarity of $(1.0-[0.875-0.626])+(1.0-[1.0-0.751]) = 0.751+0.751 = 1.502$, calculated according to the reverse triangular inequality property given in Equation 8.2). If the intersection list is empty, then the records do not match.

The MBC calculated here are supersets of all the subsets of bin combinations. For example, if we consider the bin combination of Match ID $6(B,C/C,D)$, the subsets of bin combinations for this Match ID 6 are shown in Table 8.5. As shown in this table,

Table 8.5: Subsets of bin combinations for the $(B, C/C, D)$ combination - Match ID 6 from Table 8.3

Database owner 1		Database owner 2		Total binning distance (d)
Given name	Surname	Given name	Surname	
B	C	B	C	(0+0) = 0
B	C	B	D	(0+1) = 1
B	C	C	C	(1+0) = 1
B	C	C	D	(1+1) = 2
B	D	B	C	(0+1) = 1
B	D	B	D	(0+0) = 0
B	D	C	C	(1+1) = 2
B	D	C	D	(1+0) = 1
C	C	B	C	(1+0) = 1
C	C	B	D	(1+1) = 2
C	C	C	C	(0+0) = 0
C	C	C	D	(0+1) = 1
C	D	B	C	(1+1) = 2
C	D	B	D	(1+0) = 1
C	D	C	C	(0+1) = 1
C	D	C	D	(0+0) = 0

if the combination $(B, C/C, D)$ is a match, then all the subsets of this combination can also be considered as matches, since they all have a binning distance of 2 or less. This improves the privacy aspect of our approach, because there can be many possible matching combinations (16 in this example) for one Match ID.

This parametric solution requires the number of bins k to be determined before the linkage. The selection of k is crucial for the performance of the protocol as the three main properties of PPRL, privacy, scalability and linkage quality, depend on this parameter. The larger the number of bins the smaller the range of each bin is, which results in higher accuracy of the protocol. But the smaller the number of bins the lower the computational complexity is, as the number of candidates of matching bin combinations is reduced, and the more secure the protocol is due to the higher range of bins. So the number of bins must be carefully chosen. We will experimentally investigate how these three properties are affected by the value for the parameter, the number of bins (k), in Section 8.5.

8.2.2 Protocol Specification

In this section we illustrate the steps (1 to 9) of our protocol (which we call 2P-Bin) in detail using an example consisting of two small databases, as shown in Figure 8.3, with Given names and Surnames used as the linkage attributes. Assume two database owners, *Alice* and *Bob*, with their respective databases $\mathbf{D}^A$ and $\mathbf{D}^B$, wish to identify which of their records $RA_i \in \mathbf{D}^A$ and $RB_i \in \mathbf{D}^B$ have an overall similarity $sim \geq s_t$, in order to classify them as matches. The notation used throughout this chapter is summarised in Table 8.1. Figures 8.3 to 8.13 (taken from [191]) illustrate the steps of our protocol.

D^A

RecID	Surname	Given name
RA1	millar	robert
RA2	myler	amelia
RA3	millar	gail
RA4	peter	robart
RA5	peterra	gail
RA6	smyth	amelie

D^B

RecID	Surname	Given name
RB1	millar	amelia
RB2	miller	roberto
RB3	petera	gayle
RB4	smitth	amilia
RB5	smeth	amelie
RB6	smeetth	rupert

Figure 8.3: Example databases held by *Alice* ($\mathbf{D}^A$) and *Bob* ($\mathbf{D}^B$) with Surname and Given name attributes, used to illustrate the protocol described in Section 8.2.2.

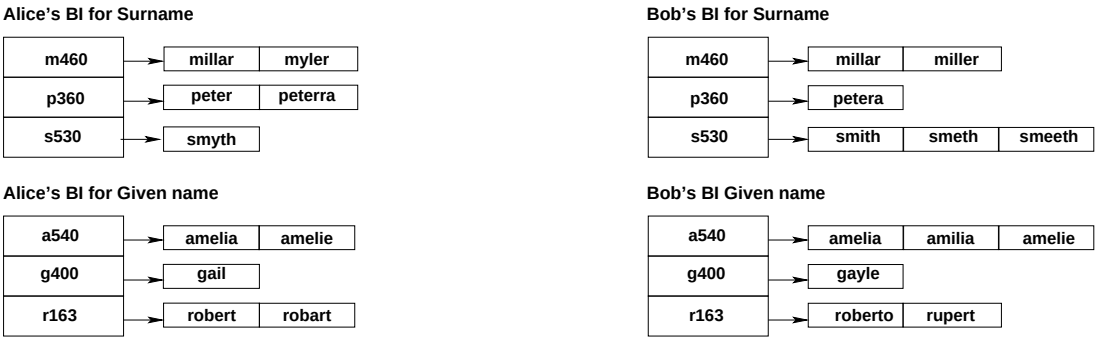


Figure 8.4: The Block Indexes (BIs) of *Alice* and *Bob* for the Surname and Given name attributes. The BIs are generated in step 2 of the protocol as the databases are loaded, and are used in step 5 to build the similarity index.

1. *Alice* and *Bob* agree upon (1) a list of attributes A to be used for the linkage (2) a blocking function (phonetic [103] is used in this example) $block_a(\cdot)$ for each attribute $a \in A$, used to generate blocking key values (BKV) b; (3) a similarity function $sim_a(v,r)$, used to calculate the numerical similarity for a pair of values v and r, where v is an attribute value and r is a reference value, such that for an exact match ($v = r$) $sim_a(v,r) = 1.0$ and for two totally different values $sim_a(v,r) = 0.0$; (4) a minimum similarity threshold s_m, which determines the start range of the first similarity bin; (5) the number of bins k to be used; (6) a binning distance d, used for finding the candidates of Matching Bin Combinations (MBC) for each attribute; (7) a hash-encoding function $enc(\cdot,h)$ and a corresponding hash key h, used to encode the Compound BKVs (CBKVs), reference lists, and finally the matching records before they are being exchanged between the database owners. This hash-encoding function can for example be the HMAC (Hashed Message Authentication Code) function [116], as described in Section 4.2.1.3. To simplify the illustration we do not apply any hash-encoding function in the example.

2. *Alice* and *Bob* each read their databases and independently build their local Block Index (BI) data structures for each linkage attribute, and a Block List Index (BLI) data structure by blocking their databases using the blocking function

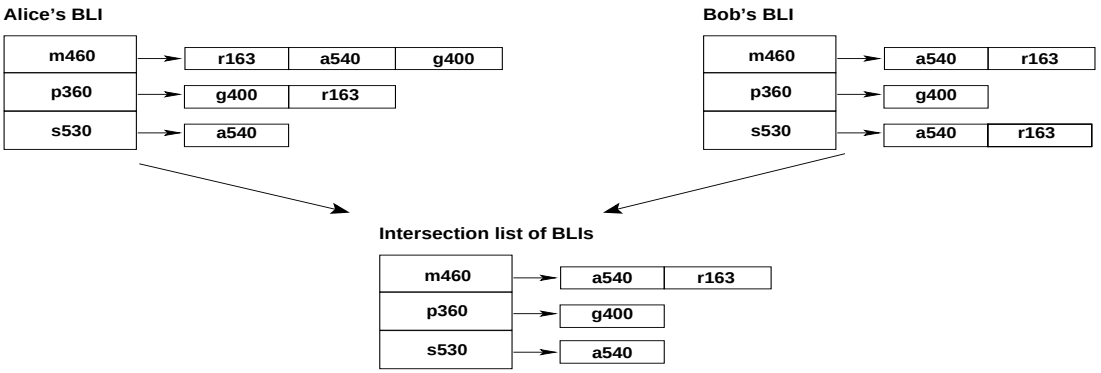


Figure 8.5: The Block List Index (BLI) of *Alice* and *Bob* and the intersection list of BLIs. Exchanging the BLIs in order to calculate the intersection list of the BLIs can reveal some information to a database owner about the other database owner's data. This is discussed in detail in Section 8.3.2. The BLI is generated in step 2 of the protocol.

Sorted compound blocks

Index number	Surname	Given name
0	m460	a540
1	m460	r163
2	p360	g400
3	s530	a540

Individual blocks

Surname	Given name
m460	a540
p360	r163
s530	g400

Figure 8.6: The compound blocks c in the sorted intersection list of BLIs and the individual blocks b for each linkage attribute. The intersection list of BLIs is sorted and the individual blocks are found in step 3 of the protocol. The compound blocks are sorted and given index numbers which will be needed in step 7 of the protocol.

$block_a(\cdot)$, as illustrated in Figures 8.4 and 8.5. The BI data structures are implemented as inverted indexes [206]. The index keys are the unique encodings / keys of a linkage attribute (the BKVs), and the corresponding lists contain the actual attribute values in a block (that have the corresponding encoding). The BLI data structure is implemented as a nested inverted index where the keys are the unique encodings of the first linkage attribute and the index lists are again inverted indexes with keys being the unique encodings of the second linkage attribute and the index lists are the lists of unique encodings of the third linkage attribute, for example if the number of linkage attributes is three. The nested inverted indexes for two linkage attributes are shown in Figure 8.5.

3. *Alice* and *Bob* exchange their BLI data structure with each other. This communication is encrypted, for example using public key encryption [171], such that only *Alice* and *Bob* can decrypt each other's values. Once the BLIs are exchanged, *Alice* and *Bob* can generate an intersection list of BLIs, as illustrated

Figure 8.7: The Reference List Index (RLI) for Surname blocks and Given name blocks. In the example, we use one reference value per block. RLI is generated in step 4 of the protocol.

in Figure 8.5. Exchanging the BLIs to find out the intersection list, which is the list of compound blocks c (individual blocks b for each linkage attribute are grouped together to generate the compound block c) that are common to both databases, might leak some information about each other's data. In order to overcome this, a secure set intersection protocol can be used that enables to find the intersection list of BLIs securely [1, 114]. This is discussed in detail in Section 8.3.2. *Alice* and *Bob* then sort the intersection list of BLIs and find the common individual blocks b for each linkage attribute separately, as illustrated in Figure 8.6.

4. The next step is to generate the Reference List Index (RLI) which contains lists of reference values, one list for each individual block b in the intersection list of BLIs (shown in Figure 8.5). The RLI can be generated by both parties together, for example one could generate reference lists for odd blocks and the other for even blocks, or one for the first attribute blocks and the other for second attribute blocks. This is shown in Figure 8.7. Exchanging the RLI between the database owners would not reveal any private information, as the RLI contains publicly known reference values (no sensitive private values). In our example, we assume the number of reference values generated for each block is 1. If more than one reference value is used, then the average of similarities with all the reference values is calculated.

5. *Alice* and *Bob* then build their Similarity Index (SI). For each unique individual block b in the intersection list of BLIs, they calculate the similarity of each unique attribute value in that block (which is stored in their BI as generated in step 2) with the list of reference values of that block (which is retrieved from the RLI). Figure 8.8 illustrates this for the running example. For example, *Alice*'s 'm460' block for Surname attribute contains two values ('millar' and'myler') and the reference value for this block is 'malar'. Therefore, the similarities between ('millar' and 'malar'), and ('myler' and 'malar') are calculated as 0.8 and 0.7, respectively, and stored in *Alice*'s SI for Surname.

6. In the next step the database owners build the bins with their similarity ranges and the Matching Bin Combinations (MBC), as illustrated in Figure 8.9. The similarity range between s_m and the complete similarity (i.e., 1.0) is split into

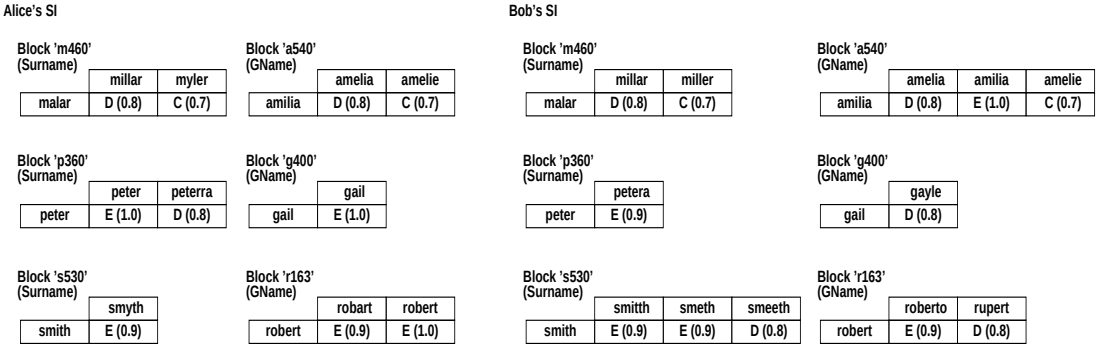


Figure 8.8: The Similarity Index (SI) which contains the similarities between attribute values and their corresponding reference values calculated using the edit distance [148] approximate string comparison function, rounded to one digit, along with their corresponding bins. The SI is generated in step 5 of the protocol.

Bin Intervals

Label	Range
A	0.5 – 0.59
B	0.6 – 0.69
C	0.7 – 0.79
D	0.8 – 0.89
E	0.9 – 1.0

Matching Bin Combinations

Match ID	Surname	Given name
1	A–B	A–B
2	A–B	B–C
3	A–B	C–D
4	A–B	D–E
5	B–C	A–B
6	B–C	B–C
7	B–C	C–D
8	B–C	D–E
9	C–D	A–B
10	C–D	B–C
11	C–D	C–D
12	C–D	D–E
13	D–E	A–B
14	D–E	B–C
15	D–E	C–D
16	D–E	D–E

Figure 8.9: The bins of similarity and the Matching Bin Combinations (MBC) used for the running example. The bins and their ranges are agreed upon by the database owners in step 1 of the protocol. In this example, the number of bins is $k = 5$ and the similarity range is from 0.5 to 1.0. The MBC are calculated based on the bins and the binning distance d. In this example, $d = 1$. The bin combinations (in MBC) are generated only for one compound block (the first compound block, i.e. the compound block with index number of 0 in Figure 8.6). Based on this, the Match IDs are calculated using Equation 8.3 for bin combinations in all the compound blocks.

k bins. Based on the bins and the binning distance d, the bin combinations in MBC are generated with their corresponding Match IDs.

7. *Alice* and *Bob* go through their database and build their local Matching Bins of Records (MBR) data structure, as shown in Figure 8.10. The MBR data structure contains unique combined blocking key values (CBKVs) of linkage attributes and for each unique CBKV, c, it contains (a) a list of bin labels for each of the attribute values (which are retrieved from SI, as generated in step 5), (b) a list of Match IDs that correspond to this combination of bin labels (which are

Alice's MBR

CBKV	Surname	Given name	Surname bin	Given name bin	Match IDs	Rec ID
(m460,r163)	millar	robert	D	E	12,16	RA1
(m460,a540)	myler	amelia	C	D	23,24,27,28	RA2
(m460,g400)	millar	gail	D	E	---	RA3
(p360,r163)	peter	robart	E	E	---	RA4
(p360,g400)	peterra	gail	D	E	44,48	RA5
(s530,a540)	smyth	amelie	E	C	63	RA6

Bob's MBR

CBKV	Surname	Given name	Surname bin	Given name bin	Match IDs	Rec ID
(m460,a540)	millar	amelia	D	D	27,31	RB1
(m460,r163)	miller	roberto	C	E	12,16	RB2
(p360,g400)	petera	gayle	E	D	47	RB3
(s530,a540)	smitth	amilia	D	E	60,64	RB4
(s530,a540)	smeth	amelie	D	C	58,59,62,63	RB5
(s530,r163)	smeeth	rupert	D	D	---	RB6

Figure 8.10: The Matching Bins of Records (MBR) of *Alice* and *Bob*. For each of the unique tuple of encoding values (CBKVs), it contains the combination of Surname and Given name attribute values with their corresponding bin labels, a list of Match IDs, and a list of record identifiers that contain the combination. The MBR is generated in step 7 of the protocol. The Match IDs are calculated only for the records that belong to the compound blocks that are in the intersection list of BLIs. The records RA3, RA4 and RB6 in this example belong to the compound blocks of ['*m460*','*g400*'], ['*p360*','*r163*'], and ['*s530*','*r163*'], respectively, which are not in the intersection list of BLIs in Figure 8.6. In other words, these compound blocks are not common in both databases and therefore they cannot be matches.

retrieved from MBC, as generated in step 6) by using Equation 8.3, and (c) a list of record IDs that contain this unique tuple of attribute values.

$$\begin{aligned} Match\ ID &= (compound\_block\_index\_number \times number\_of\_candidates\_in\_MBC) \\ &\quad + match\_ID\_in\_MBC \end{aligned} \tag{8.3}$$

It is important to note that the MBC data structure (shown in Figure 8.9) is calculated only for one compound block because all the compound blocks will have the same set of candidates of matching bin combinations. Based on this, the Match IDs can be calculated for bin combinations in all the compound blocks. The compound blocks in the intersection list of BLIs have unique index numbers (as shown in Figure 8.6). For example, consider the compound block of c=['p360','g400'] in Figure 8.6. The index number of this compound block is 2. The number of candidates in the MBC is 16 in our example (see Figure 8.9). A record with the bin combination of 'D' for Surname and 'E' for Given name attributes (in the first compound block) corresponds to Match IDs 12 and 16 in the MBC (Figure 8.9). Using Equation 8.3, the Match IDs for a record (RA5) with the same bin combination (of 'D' and 'E' for Surname and Given name attributes, respectively) in the compound block of c=['p360','g400'] (with index number of 2) would be calculated as $(2 \times 16 + 12)$ and $(2 \times 16 + 16)$, which are equal to 44 and 48 (the Match IDs for RA5 in *Alice*'s MBR in Figure 8.10).

8. Once the MBRs are generated, *Alice* and *Bob* retrieve the list of unique Match IDs from their MBR. They then exchange their list of Match IDs (MILs) with

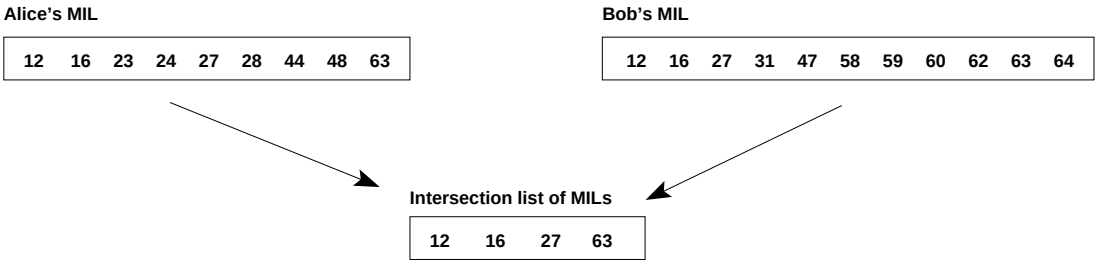


Figure 8.11: The Match ID List (MIL) of *Alice* and *Bob* that contains the list of Match IDs found in their records and the intersection list of MILs. The MIL is generated in step 8 of the protocol.

Alice's matches

Match ID	Rec ID	Surname	Given name
12	RA1	millar	robert
16	RA1	millar	robert
27	RA2	myler	amelia
63	RA6	smyth	amelie

Bob's matches

Match ID	Rec ID	Surname	Given name
12	RB2	miller	roberto
16	RB2	miller	roberto
27	RB1	millar	amelia
63	RB5	smeth	amelie

Figure 8.12: The matches of *Alice* and *Bob* for the corresponding match IDs in the intersection list of MILs, which are generated in step 9 of the protocol.

Accumulator

Match IDs	Alice	Bob	Surname similarity	Given name similarity	Total minimum similarity
12, 16 (D,E/D,E)	RA1	RB2	1.0 – [1.0–0.8]	1.0 – [1.0–0.8]	1.6
27 (C,D/C,D)	RA2	RB1	1.0 – [0.89–0.7]	1.0 – [0.89–0.7]	1.6
63 (D,E/C,D)	RA6	RB5	1.0 – [1.0–0.8]	1.0 – [0.89–0.7]	1.6

Figure 8.13: The accumulator generated by *Alice* and *Bob* which contains the matching record pairs of *Alie* and *Bob*, and their similarity range. The accumulator is generated in step 9 of the protocol.

each other and find the intersection list of the MILs, which contains the Match IDs that are common in both databases. This step is illustrated in Figure 8.11. For example, the match IDs 44 and 48 (continuing the above example) are not in common in both databases and therefore the record RA5 is not a match with any of the records in *Bob*'s database.

9. In the final step, as illustrated in Figures 8.12 and 8.13, both *Alice* and *Bob* identify the records of the matches that are corresponding to the Match IDs in the intersection list of MILs. An accumulator is built for storing these matching records, as shown in Figure 8.13.

8.3 Analysis of the Protocol

In this section we analyze our 2P-Bin protocol in terms of complexity, privacy, and linkage quality.

8.3.1 Complexity

We assume both databases contain n ($n = n^A = n^B$) records, $m = |A|$ attributes are used for linking the records, and each linkage attribute contains $u_a = u$ unique values ($1 \leq a \leq m$). We also assume that each attribute generates n_B blocks by applying the $block_a(\cdot)$ functions to block their databases. It is obvious that for large databases it commonly holds that $m \ll n_B \leq u \ll n$. In step 1, the agreement of the parameters and functions between *Alice* and *Bob* has a constant communication complexity. Reading the databases in step 2 and building the local BI data structures and the BLI data structure are $O(n)$ computation complexity, if m, n_B and u are very small compared to n, because building the BI and BLI data structures are $O(m \times u)$ and $O(m \times n_B)$, respectively.

The exchange of the BLIs in step 3 requires the communication of $m \times n_B$ values for each party, and with m, the number of linkage attributes, being comparatively a very small constant, this results in an $O(n_B)$ communication complexity. Assuming each BLI contains n_B compound blocks ($m \times n_B$ individual blocks), calculating the intersection of the two BLIs is $O(n_B \; log \; n_B)$ computation complexity.

We assume the number of reference values used for each individual block in the intersection list of the BLIs is on average n_R. In step 4, the total number of reference values to be generated and exchanged is $m \times n_B \times n_R$. With n_R and m being very small compared to n_B, this step requires $O(n_B)$ computation and communication complexities. In step 5, assuming each list (or block) in the BI that was generated in step 1 contains on average u/n_B attribute values, each of the $m \times n_B$ individual blocks requires $(u/n_B) \times n_R$ similarity calculations, and thus a total of $m \times u \times n_R$. Again with m and n_R being very small, the computation complexity of step 5 is $O(u)$.

Candidates of Matching Bin Combinations are calculated for one compound block based on the number of bins k, the similarity range (which includes the minimum similarity value s_m and the maximum similarity value 1.0), and the binning distance $d, (0 \leq d \leq k)$ for each attribute. For each of the candidates a unique Match ID is given. This can be used to calculate the Match IDs for any bin combinations in any compound block using Equation 8.3. The number of candidates is given by $(k-d)^m$ for one compound block and thus the computation complexity is $O((k-d)^m)$.

In step 7, building the MBR by reading the n records requires a total of $O(n)$ computation complexity. In the next step *Alice* and *Bob* exchange the lists of unique Match IDs that are corresponding to the bin combinations found in their records. This is $O((k-d)^m \times n_B)$, because a maximum of $(k-d)^m$ candidates of bin combinations are calculated for one compound block and with the total number of compound blocks being n_B, this step results in $O((k-d)^m \times n_B)$ communication complexity. Finding the intersection of these two lists requires $O((k-d)^m n_B \; log \; (k-d)^m n_B)$. Finally,

the generation of the accumulator to store the matches using the Match IDs in the intersection list requires a computation complexity of $O((k-d)^m \times n_B)$, because a maximum of $(k-d)^m \times n_B$ Match IDs can be found in the intersection list.

Overall, the communication and computation complexities of our protocol are linear in the size of the databases $O(n)$ and the number of blocks $O(n_B)$, but they are of exponential complexity in the number of attributes m and bins k, $O(k^m)$. The complexity of our protocol therefore depends on the value of k and the number of linkage attributes m.

8.3.2 Privacy

The protocol assumes that both *Alice* and *Bob* follow the honest but curious (HBC) adversarial model [78], in that the parties are curious and they try to find out as much as possible about the other party's data while following the protocol. The protocol is secure in this adversarial model if and only if both parties have no new knowledge at the end of the protocol above what they would have learned from the output of the matched record pairs. We analyze the privacy of our protocol by discussing what the two parties can learn from the data they communicate with each other during the protocol.

There are mainly two steps where we have to consider the privacy aspect in our protocol. One is the exchange of the BLIs (that contain compound blocks) in step 3 which might leak some information regarding the compound block values in each party's database to the other party. Using a secure set intersection (SSI) protocol to find out the intersection set of the compound blocks in both databases (without revealing any additional information about the blocks that are not in common to either party) will solve this problem. There are two major types of SSI protocols that are commutative encryption [1] and homomorphic encryption [114].The encryptions of both types of SSI protocols have a linear communication complexity. Since the exchange of the BLIs is $O(n_B)$ (as was discussed in Section 8.3.1), using a SSI protocol for this step would be feasible.

The second privacy issue in our protocol is at the step of exchanging the Match IDs (step 8) to find the intersection list that contains the Match IDs of Matching Bin Combinations that are common to both databases. The privacy of this step depends on the number of bins k. If k is large then the range of each bin is low and thus the number of unique attribute values that fall in each bin will become smaller. This results in higher probability of suspicion (P_s) values (i.e. lower privacy). So the smaller the value for the number of bins the higher the privacy of our protocol.

8.3.3 Linkage Quality

Evaluating the quality of our protocol is crucial since we use the bins of similarity values instead of the actual similarity values for the approximate matching of attribute values. In this section, we analyze the quality of our protocol in terms of efficiency measured by recall (i.e. how many true matches are retrieved by the clas-

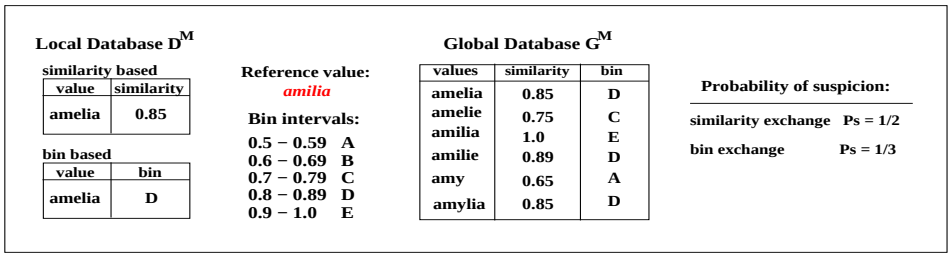


Figure 8.14: An attack method for reference values-based private matching and classification solutions (taken from [190]). The similarity of value 'amelia' (0.85) matches with two global values in $\mathbf{G}^M$ while the bin of similarity (D) matches with three global values and thus the P_s is reduced to 1/3 from 1/2.

sification model) and effectiveness measured by precision (i.e. from the record pairs that are classified as matches how many are true matches).

The quality of linkage depends on the number of bins k. If k is large, then the range of each bin is small which results in more specific ranges of similarity values, and thus the number of false matches and false non-matches will be smaller, resulting in higher precision. However, if we increase k and thus decrease the range of each bin above a certain value, then the number of false non-matches will begin to increase, because the number of missed matches, classified incorrectly as non-matched pairs (false non-matches), increases. This reduces the recall of the approach with increasing k. Therefore, the larger the number of bins used the higher the precision but lower the recall of our protocol.

Bins of similarities used in our protocol only provide similarity ranges (but not the exact similarities), and therefore ranking of which pairs are more similar than others is not possible with this approach.

8.4 Linkage Attack

A frequency linkage attack method for reference values-based private matching and classification solutions is explained in Figure 8.14 for one example database value ('amelia'). An adversary (*Alice*, *Bob*, and / or *Carol* - we only consider insider attacks in this thesis, as was described in Section 5.2) having access to a global dataset $\mathbf{G}$ can compute the number of matching values n_g in $\mathbf{G}^M$ that have the same similarity or bin of similarity with the same set of reference values to calculate the probability of suspicion P_s. DR measures can then be calculated using the P_s values for each masked value in $\mathbf{D}^M$. As illustrated in Figure 8.14, the exchange of bins of similarity reduces the probability of suspicion and thus increases the privacy guarantees compared to revealing the actual similarity values to a third party, as proposed in the three-party solution [154] (assuming the third party might collude and / or it might have information about the reference values used). In addition, the number of bins used in this 2P-Bin solution determines the privacy of this approach. If the number of bins k is large, then the similarity range of each bin becomes smaller, and

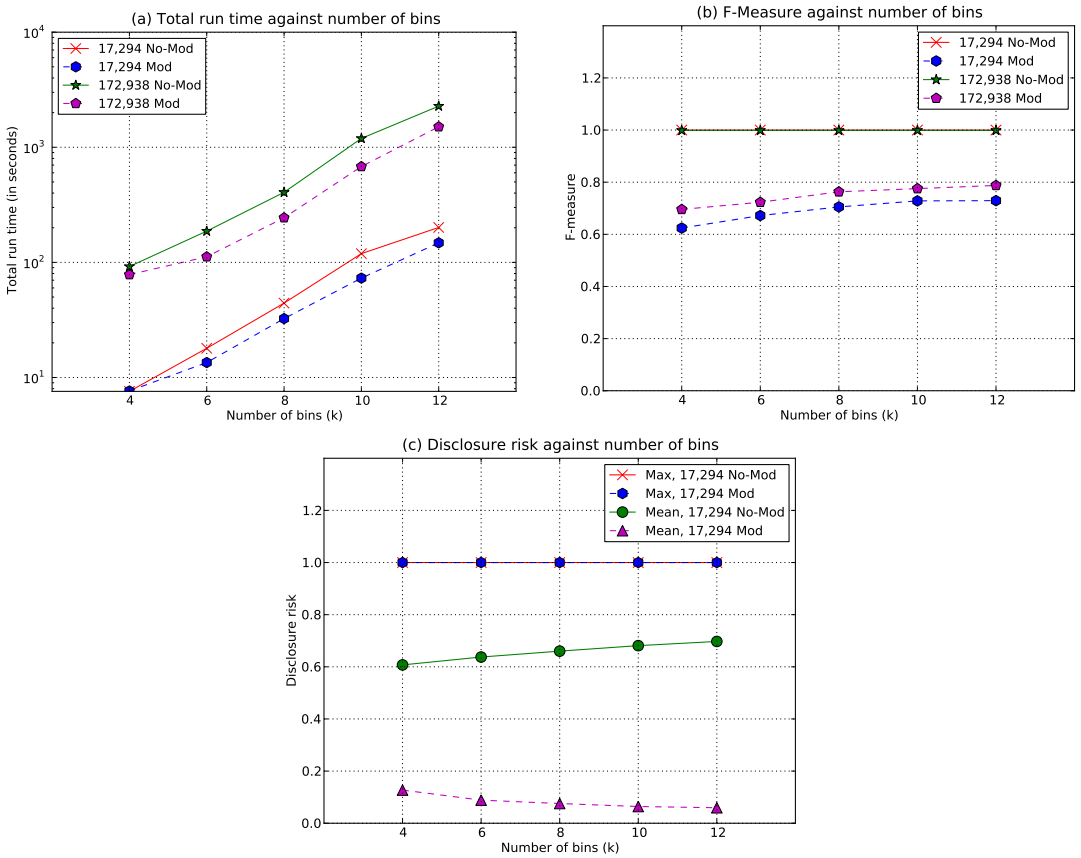


Figure 8.15: (a) Total linkage time (b) F-measure and (c) DR measures of the 2P-Bin approach on the OZ-17,294 and OZ-172,938 datasets, with different values for the number of bins parameter (k).

this results in a smaller number of global values n_g in $\mathbf{G}^M$ that match with a specific bin value. Therefore, the larger the number of bins used the lower the privacy of the solution but higher the quality of linkage. Also, the privacy of this solution depends on the number of attributes used to link records. If more number of attributes are used, then the number of combinations with the same bin values in $\mathbf{G}^M$ will become smaller and thus the probability of suspicion will be increased with more attributes.

8.5 Experimental Evaluation

In this section we present the results of the empirical study of our 2P-Bin approach conducted on the datasets described in Section 5.4 using the evaluation framework proposed in Chapter 5. The default parameters were set to $k = 6$ (this gives best results in terms of all three properties of PPRL, as shown in Figure 8.15), and $s_t = 0.8$. The similarity range was set as $s_m = 0.5$ to 1.0. Different values for k in the range of $[4, 6, 8, 10, 12]$ were also used to evaluate the performance of the 2P-Bin solution against k. All four attributes in the datasets were used as linkage attributes.

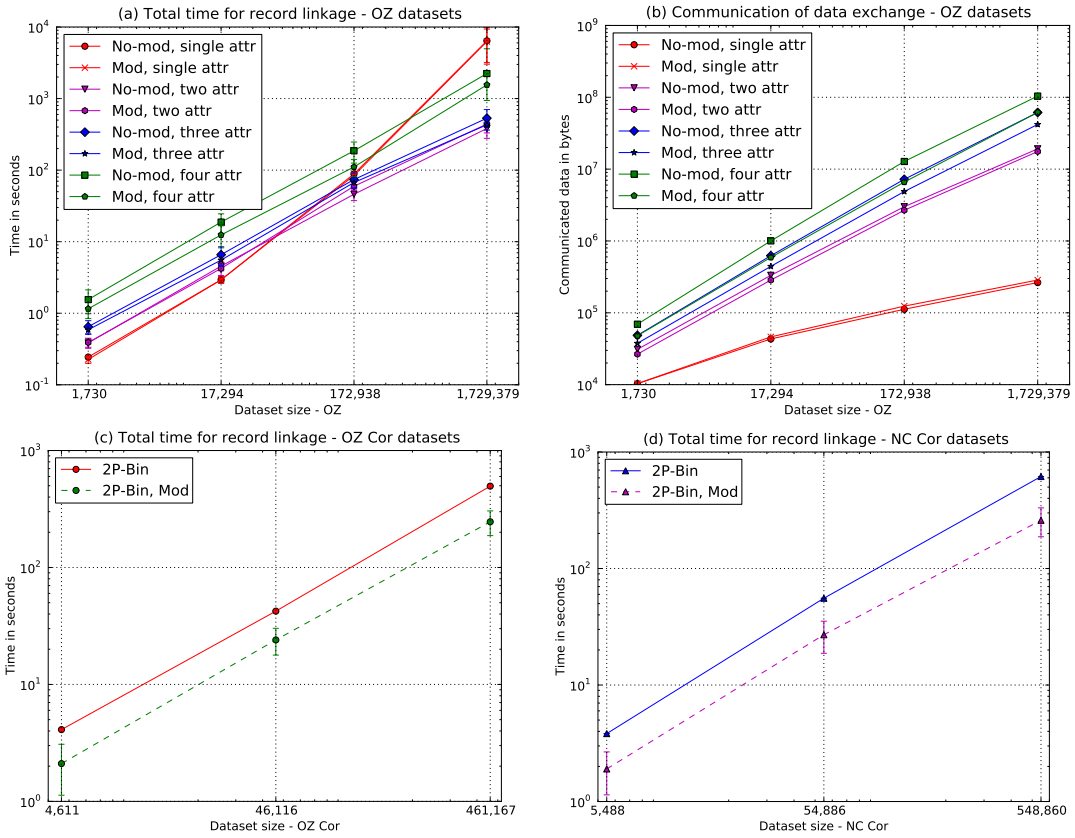


Figure 8.16: (a) Total linkage time, and (b) total communication size of the 2P-Bin approach on the OZ datasets, (c) total linkage time on the OZ Cor datasets, and (d) total linkage time on the NC Cor datasets averaged over the results of all variations of each dataset.

Figure 8.15 presents the results of the scalability, quality, and privacy of the approach for different number of bins k. The linkage time is calculated for different number of bin values to evaluate the scalability of the protocol and how it is influenced by the value for k. As shown in Figure 8.15(a), the linkage time increases with k, because the complexity of our protocol depends on the value for k (as was discussed in Section 8.3.1). As expected, the quality of linkage (calculated by F-measure) increases with k. As we discussed in Section 8.3.3, although the number of false matches decreases with k, the number of true matches missed (false non-matches) increases with k due to smaller bin ranges. This leads to a consistent F-measure with increasing k after certain point. We achieved a high F-measure of 1.0 on the No-mod datasets. Privacy evaluated by the DR measures for the linkage attack presented in Section 8.4 for different k is given in Figure 8.15(c). The disclosure risk increases with k which reduces the privacy of the protocol with increasing k.

Figure 8.16 shows the total linkage time and communication size required for private matching and classification of the 2P-Bin approach on different datasets. As can be seen from the figure, the 2P-Bin approach is efficient and is also scalable to large datasets. As expected, the computation complexity (shown in Figures 8.16(a), 8.16(c),

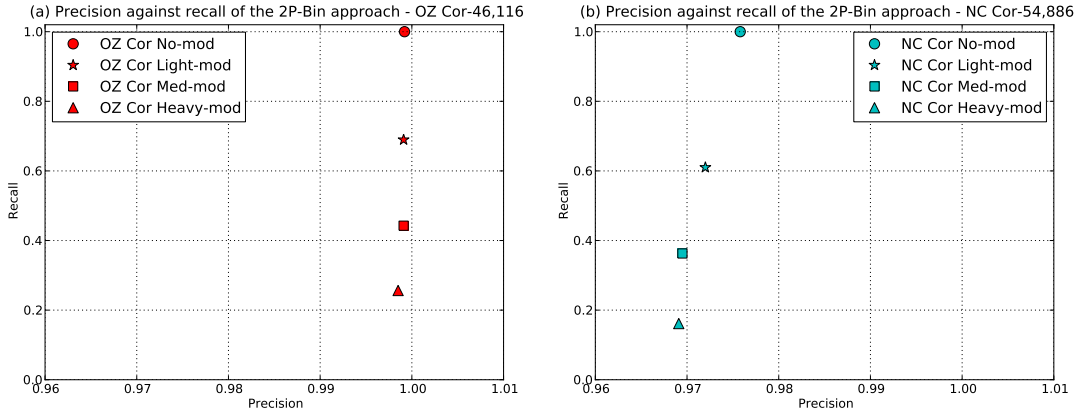


Figure 8.17: A comparison of precision against recall of the 2P-Bin solution on the (a) OZ Cor-46,116 and (b) NC Cor-54,886 datasets with No-mod, Light-mod, Med-mod, and Heavy-mod variations.

and 8.16(d)) of our approach is linear in the size of the datasets, and it increases with the number of attributes m used for the linkage (see Figure 8.16(a)). Most of the steps in our protocol depend on the number of linkage attributes m and the number of bins k used. However, the linkage performed with only one attribute takes longer time than with two, three and even four attributes, especially on larger datasets (as can be seen from Figure 8.16(a)). All the steps performed after the step of calculating the intersection list of the BLIs (step 3) are dependent on this intersection list. With only one linkage attribute, there are many common values that exist in both databases and thus the intersection list of the BLIs will be larger than when several attributes are used. As a result, the calculation of similarities of these attribute values, the generation of the Matching Bins of Records, and building the accumulator take longer time with one linkage attribute only than performing the linkage with several attributes.

As can be seen from Figure 8.16(b), the communication complexity of our protocol is linear or sub-linear in the size of the datasets. It increases with the number of attributes m used for linkage. With a smaller number of attributes used, the communication complexity tends to be more sub-linear, while with all four attributes used it becomes linear in the size of the datasets.

The computation and communication complexities of our approach on the modified datasets are lower than on the non-modified datasets, as shown in Figure 8.16. The reason is with modified datasets the number of similar attribute values that fall into the same block will be smaller, which results in a smaller number of similarity calculations compared with non-modified datasets.

A comparison of precision and recall of the 2P-Bin protocol is presented in Figure 8.17 on the OZ Cor-46,116 and NC Cor-54,886 datasets with different levels of data modifications (corruptions). Different levels of corruptions applied to the datasets (as was described in Section 5.4) allow us to evaluate the performance of approximate matching of our protocol in the presence of data errors. As shown in the figure, our 2P-Bin approach achieves high precision and recall when no modification

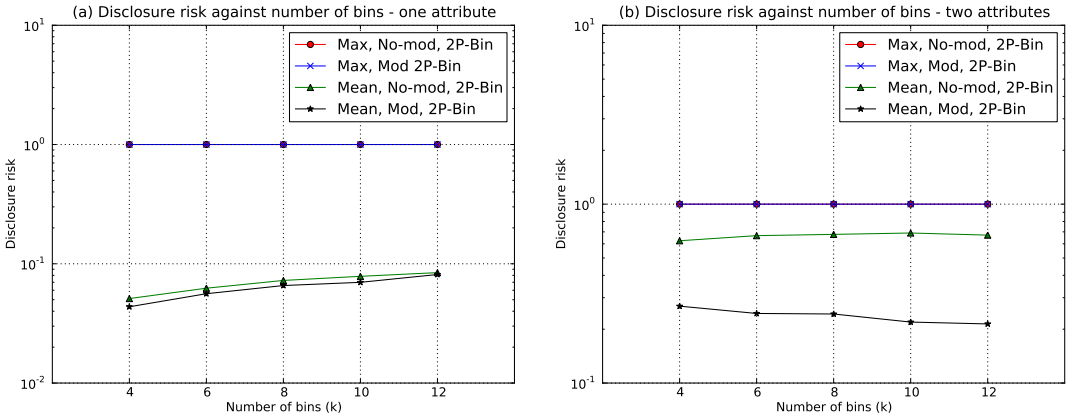


Figure 8.18: Disclosure risk of the 2P-Bin solution against the number of bins (k) used on the OZ-17,294 No-mod and Mod datasets with (a) one attribute and (b) two attributes used.

is applied to the datasets, and then the values decrease as the level of modifications increases, as one would expect. Recall drops quite drastically with the level of modifications, because the number of matches missed by the classification due to errors and other variations introduced (false non-matches) increases more than the number of false matches.

Next we evaluate the privacy of our approach using the disclosure risk measures presented in Section 5.3.1, calculated for the frequency linkage attack (proposed in Section 8.4) under the worst case assumption of $\mathbf{G} \equiv \mathbf{D}$. Figure 8.18 shows how disclosure risk increases with the number of bins (k) and the number of attributes used to link records (linkage attributes) in the 2P-Bin solution. Since we used the original dataset $\mathbf{D}$ as the global dataset $\mathbf{G}$, the number of global values that match a certain masked value is very small (resulting in high disclosure risk values). However, this worst case scenario provides a baseline for empirical privacy evaluation of solutions.

Disclosure risk values in the modified datasets are lower than the values in the non-modified datasets, because the number of global matches becomes smaller with modified (by data errors and variations) values. Interestingly, in the modified dataset the mean disclosure risk with two attributes decreases with k. This is because with modified datasets, the number of global matches n_g in $\mathbf{G}^M$ with the same bin values as the bin values in $\mathbf{D}^M$ for both attributes becomes zero with larger number of bins, and thus all the N global values in $\mathbf{G}^M$ can be considered as possible matches, which decreases the disclosure risk. Small variations in the attribute values would make a frequency linkage attack more difficult.

Linkage quality (measured by precision) against privacy (measured by DR_{Mean}) for different number of bins is shown in Figure 8.19. As can be seen from Figures 8.18 and 8.19, the 2P-Bin provides more privacy at the cost of quality loss on the modified datasets compared to non-modified datasets.

Finally, we studied how a private blocking solution combined with our 2P-Bin private matching and classification solution influences the scalability, quality, and privacy of the process. We evaluated the 2P-Bin private matching and classification

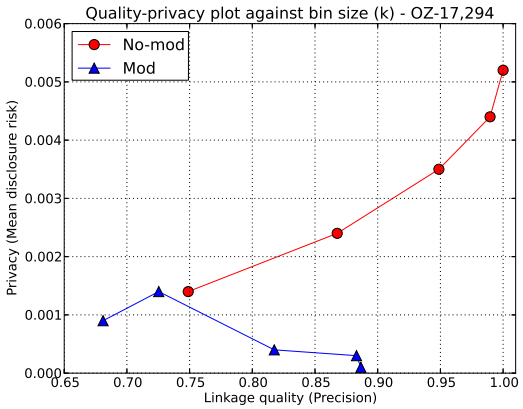


Figure 8.19: Privacy and linkage quality plot of the 2P-Bin solution for different number of bins ($k = [4, 6, 8, 10, 12]$) used on the OZ-17,294 No-mod and Mod datasets.

Table 8.6: Blocking combined with the 2P-Bin private matching and classification solution on the OZ-17,294 Mod dataset.

	No blocking	Phonetic	SNC-2P
Time (seconds)	622.7531	**10.5247**	11.6744
Precision	0.0001	**1.0000**	0.9443
Recall	**0.9993**	0.5059	0.8447
F-measure	0.0002	0.6719	**0.8917**
DR_{Mean}	**0.0006**	0.5888	0.2637
DR_{Mark}	**0.0002**	0.5885	0.3066

solution with no blocking, Soundex [23]-based phonetic blocking (a standard blocking approach that has been used in non-PPRL, as described in Chapter 2), and our SNC-based private blocking proposed in Chapter 7. In Table 8.6, we present the total time required for blocking and linkage, linkage quality results, and the DR measures in the worst case setting ($\mathbf{G} \equiv \mathbf{D}$) of our 2P-Bin solution with these three blocking scenarios. As the results show, when no blocking is applied the DR values are very low. However, it requires significantly longer linkage time compared to when a blocking technique is applied. Phonetic-based blocking requires shorter time than our SNC-based blocking, though privacy and linkage quality results are comparatively better with the SNC-based approach. Phonetic-based blocking provides lower privacy guarantees.

8.6 Summary

In this chapter, we have presented a novel two-party protocol for scalable and approximate privacy-preserving record linkage (PPRL) by using reference values and binning the similarity ranges for secure calculation of the similarities between attribute values. Our protocol is linear in the size of the databases to be linked which allows scalability to large databases. This has been validated in our experimental

evaluation where we performed the linkage on different datasets of up to a size of nearly two million records. However, our protocol is a parametric solution which depends on the number of bins k.

As shown in the experimental evaluation k plays a major role in determining the three main properties of PPRL, which are scalability, linkage quality, and privacy. A specific future research avenue is to tackle the problem of finding the optimal value for k. As similar to the previous chapters, the selection of reference values is a crucial step to achieve high linkage quality while providing sufficient privacy protection, which requires additional work in this direction. In our current implementation, we used the Levenshtein edit distance based string comparison function [148] to measure the similarity between two strings. Another extension to our current work is to compare the performances of the protocol when different approximate string comparison functions are used.

Chapter 9

Two-Party Bloom Filter-based Private Matching and Classification

Similar to the previous chapter, in this chapter we propose a novel two-party private matching and classification algorithm for privacy-preserving record linkage (PPRL) that uses one of the efficient privacy techniques, namely Bloom filter-based encoding. Our protocol conducts iterative classification of record pairs into matches and non-matches, as selected bits of the Bloom filters are exchanged across the database owners. We describe the protocol in Section 9.2 and analyze in terms of complexity, linkage quality, and privacy in Section 9.3. We then conduct an empirical study with respect to these analyses in Section 9.5 based on a linkage attack presented in Section 9.4. Finally, we summarize our findings and discuss future work in Section 9.6.

9.1 Introduction

Several approaches have been proposed to deal with PPRL over the past two decades [58, 102, 184, 194]. As we reviewed in Chapter 3, most of these approaches either use computationally expensive secure multi-party computation (SMC)-based privacy techniques [41, 78, 135] or they require a trusted third party (which might not always be available in a real application) to perform the linkage using efficient data perturbation-based privacy techniques. Developing efficient and practical algorithms for private matching and classification in PPRL applications is one of the important research directions identified in Chapter 4.

Among different perturbation-based privacy techniques that have been applied in PPRL solutions, the Bloom filter-based encoding [17] is an efficient technique that can provide adequate privacy guarantees if effectively used. A Bloom filter is an array of bits of length l, where all the bits are initially set to 0. k independent hash functions, $h_1, h_2, \ldots, h_k$, each with range $1, \ldots l$, are used to map the elements of a set into the Bloom filter by setting the corresponding bit positions to 1. Bloom filters have previously been used in several three-party and multi-party PPRL solutions.

Schnell et al. [174] were the first to propose a method for approximate matching in PPRL using Bloom filters. In their work, the attribute values of each record in the databases to be linked are concatenated into one string, and the q-grams (sub-

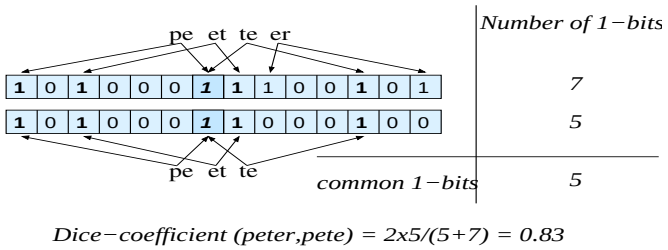


Figure 9.1: Mapping of strings into Bloom filters and calculating their Dice coefficient similarity (taken from [193]).

strings of length q) of these strings are mapped into Bloom filters using k independent hash functions. This method of encoding is known as cryptographic longterm key (CLK) [175]. The Bloom filters are then sent to a third party and the Dice coefficient [29] is used to calculate the similarity of two Bloom filters:

$$sim(b^A, b^B) = \frac{2c}{x^A + x^B} \tag{9.1}$$

where c is the number of common bit positions that are set to 1 in both Bloom filters b^A and b^B (common 1-bits), x^A is the number of bit positions that are set to 1 in b^A, and x^B is the number of bit positions that are set to 1 in b_B. The Dice coefficient is used since it is insensitive to many matching zeros in long Bloom filters [174]. For example, mapping the bigrams ($q = 2$) of the two string values 'peter' and 'pete' into $l = 14$ bits long Bloom filters using $k = 2$ hash functions and calculating the Dice coefficient similarity of the two Bloom filters are illustrated in Figure 9.1.

The approach is efficient because of the use of Bloom filters and it supports approximate matching of values as well, rendering it applicable to real-world conditions. However, as with other three-party protocols, collusion between the parties is a major privacy drawback of this approach [174]. Recent research in PPRL has analysed the weaknesses of Bloom filters using constraint satisfaction cryptanalysis [122], and novel solutions based on random sampling of bits from field-level Bloom filters have been proposed to improve the privacy of Bloom filter-based PPRL [56, 175].

Our aim is to develop a two-party protocol for PPRL using Bloom filters. We propose a protocol that eliminates the need of a third party by adopting an iterative method for revealing selected bits in the Bloom filters between the database owners and classifying record pairs into matches and non-matches in an iterative way such that the pairs that are unlikely to correspond to matches are removed before revealing more bits for those pairs. An iterative strategy was introduced in a previous two-party exact matching solution [14] for PPRL where characters of hash-encoded values are iteratively revealed until they are identical for a certain length. The use of iterative method would prevent revealing more information for pairs that are highly likely to be non-matches. We use the cryptographic longterm key (CLK) [175] based Bloom filter encoding for our two-party solution in this chapter and we will compare different encoding methods (which we will discuss in Section 9.3.3) in Chapter 10.

Table 9.1: Notation used in this chapter.

$\mathbf{D}^A$, $\mathbf{D}^B$	Databases held by database owners *Alice* and *Bob*, respectively
$\mathbf{S}^A$, $\mathbf{S}^B$	Lists containing tuples of linkage attributes' values for each record in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
b^A, b^B	A Bloom filter, one for each record in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
$\mathbf{O}^A$, $\mathbf{O}^B$	Lists of record IDs and number of 1-bits for each record in $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively
$\mathbf{C}$, $\mathbf{C_i}$	List of candidate record pairs, list of candidate record pairs at iteration i
s_t	Minimum similarity threshold value to classify a record pair as a match
s_l	Minimum acceptable similarity threshold value to add noise
s_r	Minimum similarity threshold value to reveal bits in an iteration
l	Length of Bloom filters
$h_1 \dots h_k$, k	Hash functions used to map a set of elements into a Bloom filter, number of hash functions
q, i	Number of characters that make a q-gram, iteration i, $i > 0$
r, r_i	Number of bit positions revealed, number of bit positions revealed in iteration i
t_i	Total number of bit positions revealed so far up to iteration i, $t_i = \sum_i r_i$
x, x^A, x^B	Number of 1-bits, number of 1-bits in b^A and b^B, respectively
x_i, x_i^A, x_i^B	Total number of 1-bits revealed so far up to iteration i, total number of 1-bits revealed in b^A and b^B so far up to iteration i, respectively
r_{min}, r_{max}, z_{max}	Minimum number of bits that can be revealed in an iteration, maximum number of total bits to be revealed, maximum number of noise bits that can be added or removed
c_{min}, c_i	Minimum number of common 1-bits required in both Bloom filters b^A and b^B to be classified as a match, total number of common 1-bits revealed from both Bloom filters b^A and b^B so far up to iteration i
d	Difference between x^A and x^B
d_{max}	Maximum difference between x^A and x^B to be classified as a match
$sim(\cdot, \cdot)$	Function used to calculate the similarity of two Bloom filters b^A and b^B (Dice coefficient)

9.2 Proposed Solution

Similar as in the previous chapters, we assume two database owners, *Alice* and *Bob*, with their respective databases D^A and D^B, participate in the protocol. We divide the steps of our two-party Bloom filter-based protocol (2P-BF) into three main phases, which are the preparation phase, the length filtering phase, and the iterative classification phase. The notation we use is summarized in Table 9.1. Figures 9.2 to 9.7 (taken from [188]) illustrate the steps of the protocol.

9.2.1 Preparation Phase

In the initial preparation phase the database owners prepare their data to be used in the protocol. The steps of this first phase are:

1. *Alice* and *Bob* agree upon a bit array length l; k hashing functions $h_1 \dots h_k$; the length (in characters) of grams q; the similarity function $sim(b^A, b^B)$ to measure the similarity of two Bloom filters b^A and b^B; a minimum similarity threshold value s_t, above which a pair of records is classified as a match; the maximum number of bit positions they are willing to reveal to each other r_{max} ($r_{max} \leq l$); and a set of attributes A (linkage attributes) that are used to link the records.

2. *Alice* and *Bob* each stores the tuples of the linkage attributes' values $(a_1, \cdots, a_m)$ in a list, $\mathbf{S}^A$ and $\mathbf{S}^B$, respectively, for each of the records in their databases.

3. For every tuple s in $\mathbf{S}^A$, *Alice* performs the following steps:

   (a) *Alice* converts each attribute string a_i in s ($1 \leq i \leq m$) into a set of q-grams.

Alice's Bloom Filters

RecID	Bloom Filters	Num 1s (x^A)	O^A
RA1	1 1 1 1 0 1 0 1 0 0	6	RA1, 6
RA2	0 1 0 0 0 0 1 0 0 1	3	RA2, 3
RA3	0 1 1 1 0 1 1 0 0 1	6	RA3, 6
RA4	1 1 0 0 1 0 1 1 0 0	5	RA4, 5

Bob's Bloom Filters

RecID	Bloom Filters	Num 1s (x^B)	O^B
RB1	1 1 1 1 0 1 0 1 1 0	7	RB1, 7
RB2	1 1 1 1 0 1 1 0 0 1	7	RB2, 7
RB3	1 1 0 1 1 0 1 0 0 0	5	RB3, 5

Figure 9.2: Example Bloom filters held by *Alice* and *Bob* for the records in their databases $\mathbf{D}^A$ and $\mathbf{D}^B$, respectively, and the number of 1-bits in each of their Bloom filers along with the record identifiers stored in $\mathbf{O}^A$ and $\mathbf{O}^B$, respectively.

(b) *Alice* maps these q-gram sets into a Bloom filter b^A (of that record) of length l using the hash functions $h_1 \ldots h_k$. All the attribute values ($a_1, \cdots, a_m$ in s) of a record are mapped into one single Bloom filter.

4. *Alice* also counts for each Bloom filter the number of bit positions that are set to 1 (1-bits), x^A, and stores this number along with the identifier of the record into its list $\mathbf{O}^A$, as illustrated in Figure 9.2 for the example Bloom filters.

5. For every tuple of attribute values s in $\mathbf{S}^B$, *Bob* performs steps 3 and 4.

9.2.2 Length Filtering Phase

The second phase of our protocol aims to remove non-matching record pairs using a length filtering method on the Bloom filters. The output of this phase is a set of candidate record pairs ($\mathbf{C}$) generated with their corresponding value for the minimum number of common 1-bits they require (c_{min}) to be potentially classified as a match. We use the Dice-coefficient (Equation 9.1) as the similarity function $sim(\cdot, \cdot)$ to compare two Bloom filters, as it is insensitive to many zeros in Bloom filters [174]. However, any q-gram-based similarity function can be used [29]. Algorithm 9.1 shows the main steps involved in this phase.

1. *Alice* and *Bob* exchange the number of 1-bits in each of their Bloom filters along with their record identifiers or randomly generated unique ID numbers (lists $\mathbf{O}^A$ and $\mathbf{O}^B$, respectively). They then generate all the record pairs ($|\mathbf{D}^A| \times |\mathbf{D}^B|$ if no blocking function is applied, see Section 9.2.4 for how this can be improved) along with the number of 1-bits (x^A and x^B), as illustrated in Figure 9.3.

2. In order to consider a record pair as a possible match, the difference between the number of 1-bits in their Bloom filters $d = |x^A - x^B|$ should be less than or equal to the maximum bit difference d_{max}, which can be calculated as below. Assume $x^A \leq x^B$ and all the bit positions set to 1 in b^A are also set to 1 in b^B ($c = x^A$). This worst case assumption gives the lower bound of the similarity coefficient (s_t) and the upper bound of bit difference (d_{max}). The value for d_{max}

Record Pairs

A	B	x^A	x^B	Length Filter
RA1	RB1	6	7	(|6−7| <= 6/2) Yes
RA1	RB2	6	7	(|6−7| <= 6/2) Yes
RA1	RB3	6	5	(|6−5| <= 6/2) Yes
RA2	RB1	3	7	(|3−7| <= 3/2) No
RA2	RB2	3	7	(|3−7| <= 3/2) No
RA2	RB3	3	5	(|3−5| <= 3/2) No
RA3	RB1	6	7	(|6−7| <= 6/2) Yes
RA3	RB2	6	7	(|6−7| <= 6/2) Yes
RA3	RB3	6	5	(|6−5| <= 5/2) Yes
RA4	RB1	5	7	(|5−7| <= 5/2) Yes
RA4	RB2	5	7	(|5−7| <= 5/2) Yes
RA4	RB3	5	5	(|5−5| <= 5/2) Yes

Candidate Record Pairs

A	B	x^A	x^B	c_{min}
RA1	RB1	6	7	6
RA1	RB2	6	7	6
RA1	RB3	6	5	5
RA3	RB1	6	7	6
RA3	RB2	6	7	6
RA3	RB3	6	5	5
RA4	RB1	5	7	5
RA4	RB2	5	7	5
RA4	RB3	5	5	4

Figure 9.3: Record pairs that are likely to be non-matches are pruned (length filtering) according to the number of 1-bits, x^a and x^b, using Equation 9.2, as shown in the left table. Candidate record pairs generated after the length filtering phase are shown in the right table, with the minimum number of common 1-bits required to be classified as a match, c_{min}, according to the values of x^A and x^B, calculated using Equation 9.3. s_t is set to 0.8. The minimum value of all c_{min}, $min(c_{min})$, is 4 which will be used as the value for r_1 in the first iteration ($i = 1$).

can be calculated given the minimum similarity coefficient threshold s_t and the number of 1-bits in the Bloom filters, x^A and x^B, as shown in Equation 9.2.

$$\begin{aligned} sim(b^A, b^B) &= \frac{2c}{x^A + x^B} \geq s_t \\ \frac{2\min(x^A, x^B)}{\min(x^A, x^B) + (\min(x^A, x^B) + d)} &\geq s_t \\ \frac{2x^A}{x^A + x^A + d} &\geq s_t \\ d &\leq \frac{2x^A(1 - s_t)}{s_t} \\ d_{max} &= \frac{2x^A(1 - s_t)}{s_t}. \end{aligned} \tag{9.2}$$

A record pair must exhibit at most d_{max} difference between the 1-bits in their Bloom filters in order to be considered as a possible match (according to the similarity threshold value s_t). All the pairs that have a larger 1-bit difference than d_{max} can be removed without proceeding further, since they cannot be matches. For example, if s_t is set to 0.8, then the difference between 1-bits in two Bloom filters must be at maximum half the value of the smaller value for the 1-bits in the two Bloom filters ($0.5 \times min(x^A, x^B)$) in order to be classified as a match, following $sim(b^A, b^B) \geq 0.8 \Rightarrow \frac{2c}{x_1^A + x_1^B} \geq \frac{8}{10} \Rightarrow \frac{2x_1^A}{x_1^A + (x_1^A + d)} \geq \frac{8}{10} \Rightarrow d \leq 0.5x_1^A$. *Alice* and *Bob* store only the record pairs that have $|x^A - x^B| \leq d_{max}$, as illustrated in Figure 9.3.

Algorithm 9.1 : Length filtering (phase 2)

Input: $\mathbf{O}^A$: List of record IDs and number of 1-bits (r^A, x^A) from *Alice*
$\mathbf{O}^B$: List of record IDs and number of 1-bits (r^B, x^B) from *Bob*
s_t: Minimum similarity threshold
Output: $\mathbf{C}$: List of candidate record pairs with their minimum number of common 1-bits required (c_{min})

```
C = [ ]
for (r_i^A, x_i^A) ∈ O^A do
    for (r_i^B, x_i^B) ∈ O^B do
        x_min = min(x_i^A, x_i^B)
        d = |x_i^A − x_i^B|
        d_max = 2x_min(1−s_t) / s_t
        if d ≤ d_max then
            c_min = ⌊ s_t(x_i^A + x_i^B) / 2 ⌋
            C+= ([r_i^A, x_i^A], [r_i^B, x_i^B], c_min)
        end if
    end for
end for
```

3. *Alice* and *Bob* now calculate the minimum number of common 1-bits required for a record pair to be classified as a match, c_{min}, for each pair of the remaining candidate records, as illustrated in Figure 9.3. This is calculated for each pair using the values for x^A, x^B and s_t as shown in Equation 9.3, where $\lfloor \cdot \rfloor$ denotes the rounding to the next lower integer value. The resulting candidate record pairs with their values for x^A, x^B, and c_{min} are stored in the Candidates Index data structure, $\mathbf{C}$ (as shown in the right table in Figure 9.3), which will be used as an input to the next phase of the protocol, the iterative classification phase.

$$\begin{aligned} sim(b^A, b^B) &= \frac{2c}{x^A + x^B} \geq s_t \\ &\frac{2c_{min}}{x^A + x^B} = s_t \\ c_{min} &= \lfloor \frac{s_t(x^A + x^B)}{2} \rfloor \end{aligned} \tag{9.3}$$

9.2.3 Iterative Classification Phase

The main task of a record linkage process is the classification of record pairs [29]. The iterative classification phase is where we classify record pairs into matches, non-matches, and possible matches. This classification needs to be done in such a way that no information about the attribute values that were mapped into Bloom filters is being revealed to the two database owners, with the exception of some information regarding the matches.

We assume that *Alice* and *Bob* are prepared to reveal r_{max} bit positions to each other in an iterative way without compromising the sensitive values in their Bloom filters, where r_{max} is the maximum number of bits in Bloom filters that the database owners agree to reveal ($r_{max} \leq l$). The number of bits to be revealed in each iteration, r_i, is a crucial parameter to be set as it provides a trade-off between privacy and computational efficiency of the protocol. There are two possible extreme cases.

Candidate Record Pairs − Iteration 1

A	B	c_{min}	Alice's BF	Bob's BF	$c_{min} - c_1$	$x^A - x^A_1$	$x^B - x^B_1$	Class
RA1(6)	RB1(7)	6	1 1 1 1	1 1 1 1	2	RA1(2)	RB1(3)	Pos Match
RA1(6)	RB2(7)	6	1 1 1 1	1 1 1 1	2	RA1(2)	RB2(3)	Pos Match
RA1(6)	RB3(5)	5	1 1 1 1	1 1 0 1	2	RA1(2)	RB3(2)	Pos Match
RA3(6)	RB1(7)	6	0 1 1 1	1 1 1 1	3	RA3(3)	RB1(3)	Pos Match
RA3(6)	RB2(7)	6	0 1 1 1	1 1 1 1	3	RA3(3)	RB2(3)	Pos Match
RA3(6)	RB3(5)	5	0 1 1 1	1 1 0 1	3	RA3(3)	RB3(2)	*Non Match*
RA4(5)	RB1(7)	5	1 1 0 0	1 1 1 1	3	RA4(3)	RB1(3)	Pos Match
RA4(5)	RB2(7)	5	1 1 0 0	1 1 1 1	3	RA4(3)	RB2(3)	Pos Match
RA4(5)	RB3(5)	4	1 1 0 0	1 1 0 1	2	RA4(3)	RB3(2)	Pos Match

Figure 9.4: Bloom Filters of *Alice* and *Bob* with $t_1 = 4$ ($r_1 = min(c_{min}) = 4$) bits revealed after the first iteration. The calculated values for c_1 are used to calculate the value for r_2 for the next iteration, $r_2 = min(c_{min} - c_1) = 2$.

1. Revealing all the r_{max} bits in one iteration, which is very fast but is not secure since all the r_{max} bit positions are revealed for all the Bloom filter pairs including non-matches as well. This might allow *Alice* and *Bob* to re-identify certain values of non-matches from the revealed bit patterns based on a linkage attack, as will be explained in Section 9.4.

2. Revealing the r_{max} bits in r_{max} iterations where only 1 bit position is revealed in each iteration. This would be the best case for preserving privacy as it removes the non-matches in an iterative way before revealing the rest of the bit positions. This approach is however not scalable to large databases, especially with long Bloom filters, as each iteration requires communication between the database owners.

Hence, a method to reveal the optimal number of bits, r_i, in each iteration is required. We propose a method to calculate this optimal number by finding the smallest value of the minimum number of additional common 1-bits required to classify a pair as a match in each iteration among all the record pairs. The record pair that requires the smallest number of additional common 1-bits among all the other pairs has a privacy risk if more bit positions are revealed than the minimum number of common 1-bits it requires.

Assume c_i is the total number of common 1-bits revealed so far up to iteration i. The value for $min(c_{min} - c_{i-1})$ ($i > 0$) is calculated to be used as the value for r_i in the i^{th} iteration, with c_{min} as calculated in Equation 9.3. For example, in the first iteration ($i = 1$), $min(c_{min})$ ($c_0 = 0$) will be used as the value for the number of bit positions to be revealed, r_1. After r_1 bit positions are revealed in the first iteration, the value for $(c_{min} - c_1)$ will be calculated for each of the remaining record pairs to calculate the value for $r_2 = min(c_{min} - c_1)$ in the second iteration, and then $min(c_{min} - c_2)$ will be used as the value for r_3 in the third iteration, and so on.

Candidate Record Pairs − Iteration 2

A	B	$c_{min} - c_1$	Alice's BF	Bob's BF	$c_{min} - c_2$	$x^A - x_2^A$	$x^B - x_2^B$	Class
RA1(2)	RB1(3)	2	1 1 1 1 0 1 ⊠⊠⊠⊠	1 1 1 1 0 1 ⊠⊠⊠⊠	1	RA1(1)	RB1(2)	Pos Match
RA1(2)	RB2(3)	2	1 1 1 1 0 1 ⊠⊠⊠⊠	1 1 1 1 0 1 ⊠⊠⊠⊠	1	RA1(1)	RB2(2)	Pos Match
RA1(2)	RB3(2)	2	1 1 1 1 0 1 ⊠⊠⊠⊠	1 1 0 1 1 0 ⊠⊠⊠⊠	2	RA1(1)	RB2(1)	*Non Match*
RA3(3)	RB1(3)	3	0 1 1 1 0 1 ⊠⊠⊠⊠	1 1 1 1 0 1 ⊠⊠⊠⊠	2	RA3(2)	RB1(2)	Pos Match
RA3(3)	RB2(3)	3	0 1 1 1 0 1 ⊠⊠⊠⊠	1 1 1 1 0 1 ⊠⊠⊠⊠	2	RA3(2)	RB2(2)	Pos Match
RA4(3)	RB1(3)	3	1 1 0 0 1 0 ⊠⊠⊠⊠	1 1 1 1 0 1 ⊠⊠⊠⊠	3	RA4(2)	RB1(2)	*Non Match*
RA4(3)	RB2(3)	3	1 1 0 0 1 0 ⊠⊠⊠⊠	1 1 1 1 0 1 ⊠⊠⊠⊠	3	RA4(2)	RB2(2)	*Non Match*
RA4(3)	RB3(2)	2	1 1 0 0 1 0 ⊠⊠⊠⊠	1 1 0 1 1 0 ⊠⊠⊠⊠	1	RA4(2)	RB3(1)	Pos Match

Figure 9.5: Bloom Filters of *Alice* and *Bob* with $t_2 = 6$ ($r_2 = 2$) bits revealed after the second iteration. The calculated values for c_2 are used to calculate the value for r_3 for the next iteration, $r_3 = min(c_{min} - c_2) = 1$.

Candidate Record Pairs − Iteration 3

A	B	$c_{min} - c_2$	Alice's BF	Bob's BF	$c_{min} - c_3$	$x^A - x_3^A$	$x^B - x_3^B$	Class
RA1(1)	RB1(2)	1	1 1 1 1 0 1 0 ⊠⊠⊠	1 1 1 1 0 1 0 ⊠⊠⊠	1	RA1(1)	RB1(2)	Pos Match
RA1(1)	RB2(2)	1	1 1 1 1 0 1 0 ⊠⊠⊠	1 1 1 1 0 1 1 ⊠⊠⊠	1	RA1(1)	RB2(1)	Pos Match
RA3(2)	RB1(2)	2	0 1 1 1 0 1 1 ⊠⊠⊠	1 1 1 1 0 1 0 ⊠⊠⊠	2	RA3(1)	RB1(2)	Pos Match
RA3(2)	RB2(2)	2	0 1 1 1 0 1 1 ⊠⊠⊠	1 1 1 1 0 1 1 ⊠⊠⊠	1	RA3(1)	RB2(1)	Pos Match
RA4(2)	RB3(1)	1	1 1 0 0 1 0 1 ⊠⊠⊠	1 1 0 1 1 0 1 ⊠⊠⊠	0	RA4(1)	RB3(0)	*Match*

Figure 9.6: Bloom Filters of *Alice* and *Bob* with $t_3 = 7$ ($r_3 = 1$) bits revealed after the third iteration. The calculated values for c_3 are used to calculate the value for r_4 for the next iteration, $r_4 = min(c_{min} - c_3) = 1$.

Candidate Record Pairs − Iteration 4

A	B	$c_{min} - c_3$	Alice's BF	Bob's BF	$c_{min} - c_4$	$x^A - x_4^A$	$x^B - x_4^B$	Class
RA1(1)	RB1(2)	1	1 1 1 1 0 1 0 1 ⊠⊠	1 1 1 1 0 1 0 1 ⊠⊠	0	RA1(0)	RB1(1)	*Match*
RA1(1)	RB2(1)	1	1 1 1 1 0 1 0 1 ⊠⊠	1 1 1 1 0 1 1 0 ⊠⊠	1	RA1(0)	RB2(1)	*Non Match*
RA3(1)	RB1(2)	2	0 1 1 1 0 1 1 0 ⊠⊠	1 1 1 1 0 1 0 1 ⊠⊠	2	RA3(1)	RB1(1)	*Non Match*
RA3(1)	RB2(1)	1	0 1 1 1 0 1 1 0 ⊠⊠	1 1 1 1 0 1 1 0 ⊠⊠	1	RA3(1)	RB2(1)	Pos Match

Figure 9.7: Bloom Filters of *Alice* and *Bob* with $t_4 = 8$ ($r_4 = 1$) bits revealed after the fourth iteration. The pairs that are still classified as possible matches (the pair of records $RA3$ and $RB2$ in this example) will need to be re-processed with different hash functions. $r_{max} = 8$ in our example.

Algorithm 9.2 : Iterative classification (phase 3)

Input: **C**: Candidate record pairs from phase 2
r_{max}: Maximum number of bits to be revealed in Bloom filters
Output: **M**: Set of record pairs classified as matches
N: Set of record pairs classified as non-matches
P: Set of record pairs classified as possible matches

M $= [\,]$; **N** $= [\,]$; **P** $=$ **C**
while P $\neq [\,]$ **do**
  $i = 1$; $t = 0$
  while $t \leq r_{max}$ **do**
    $r_i = min(c_{min} - c_{i-1})$
    $t = t + r_i$
    for $(b^A, b^B) \in$ **P do**
      x^A = number of 1-bits in b^A
      x^B = number of 1-bits in b^B
      c_{min} = number of common 1-bits in b^A and b^B
      reveal_bits(r) {A function to reveal bits from Bloom filters}
      x_i^A =total number of 1-bits revealed in b^A
      x_i^B =total number of 1-bits revealed in b^B
      c_i =total number of common 1-bits revealed in b^A and b^B
      if $c_i \geq c_{min}$ **then**
        M+= (b^A, b^B)
        P-= (b^A, b^B)
      else if $c_i < c_{min}$ **and** $(c_{min} - c_i) > (l - t)$ **then**
        N+= (b^A, b^B)
        P-= (b^A, b^B)
      else if $c_i < c_{min}$ **and** $(c_{min} - c_i) \leq (l - t)$ **then**
        if $((x^A - x_i^A) < (c_{min} - c_i))$ **or**
          $((x^B - x_i^B) < (c_{min} - c_i))$ **then**
          N+= (b^A, b^B)
          P-= (b^A, b^B)
        end if
      end if
    end for
    i+= 1
  end while
  for $(b^A, b^B) \in$ **P do**
    Do_rehash(**P**) {Restart the protocol from phase 1}
  end for
end while

The iterative classification phase is done as follows (Algorithm 9.2 provides an overview of these steps):

1. Among all the $(c_{min} - c_{i-1})$ values for all the unclassified pairs of records, the minimum value, $min(c_{min} - c_{i-1})$, is taken as the lower bound of the number of bits to be revealed in the next iteration. *Alice* and *Bob* both will exchange $r_i = min(c_{min} - c_{i-1})$ same bit positions from each of their Bloom filters. For example, if $r_1 = min(c_{min} - c_0) = min(c_{min}) = 4$, then the first 4 bit positions are exchanged in the first iteration, as shown in Figure 9.4. The total number of bit positions revealed so far up to iteration i is $t_i = \sum_i r_i$.

   From the exchange of t_i bit positions, three possible cases can occur with each record pair.

   - Case 1: Record pairs which have c_{min} or more than c_{min} out of t_i bit positions set to 1 in both Bloom filters (b^A and b^B) ($c_i \geq c_{min}$). These

pairs are classified as matches, because the similarity of these pairs is $sim(b^A, b^B) \geq s_t$ as explained in Equation 9.3.

- Case 2: Record pairs which have some or none of the t_i bit positions set to 1 in both Bloom filters b^A and b^B ($c_i < c_{min}$) and the number of additional common 1-bits required $(c_{min} - c_i)$ is greater than the number of remaining unrevealed bit positions ($c_i < c_{min}$ and $(c_{min} - c_i) > (l - t_i)$). These pairs are classified as non-matches, because the remaining number of unrevealed bits is not sufficient to be $c_i \geq c_{min}$.
- Case 3: Record pairs which have some or none of the t_i bit positions set to 1 in both Bloom filters b^A and b^B ($c_i < c_{min}$) and the number of additional common 1-bits required $(c_{min} - c_i)$ is less than or equal to the number of remaining unrevealed bit positions ($c_i < c_{min}$ and $(c_{min} - c_i) \leq (l - t_i)$). These record pairs are classified as possible matches, as there can be more common 1-bits in the unrevealed bits to be $c_i \geq c_{min}$.

2. After having t_i bit positions revealed in iteration i, all the pairs that are classified as matches and non-matches (cases 1 and 2) can be removed from the set of candidate record pairs **C**. Only the pairs that are classified as possible matches (case 3) will be taken to the next iteration.

   Based on the revealed bit positions, *Alice* and *Bob* calculate the new values for c_i, x_i^A, and x_i^B. Moreover, the values for x_i^A and x_i^B can also be used to prune more non-matches from the pairs of records that were classified as possible matches. Record pairs which have $(c_{min} - c_i) < (x^A - x_i^A)$ or $(c_{min} - c_i) < (x^B - x_i^B)$ can be classified as non-matches and pruned. For example, if 2 more 1-bits are left unrevealed in b^B ($x^B - x_i^B = 2$) after revealing 4 bit positions in the first iteration, and $(c_{min} - c_i)$ is 3 which means at least 3 more common 1-bits are required for the record pair to be classified as a match from only 2 1-bits in b^B (which is impossible), then this record pair can be removed at this iteration without participating in the next iteration and revealing more bits for this non-matching pair. Record pair RA3 and RB3 in Figure 9.4 is such a case.

3. For the pairs that are classified as possible matches (case 3), *Alice* and *Bob* repeat the steps until r_{max} bit positions are exchanged in an iterative method (r_{max} is set to 8 in our running example).

4. The record pairs that are still classified as possible matches in the last step, after r_{max} bit positions have been revealed, need to be encoded (re-hashed) into new Bloom filters by different k hash functions (lines 30-32 in Algorithm 9.2).

9.2.4 Improving Efficiency

In the length filtering phase, we remove record pairs that have a difference between the number of 1-bits larger than a certain value, depending on the minimum similarity threshold value s_t before starting the iterations, as explained in Section 9.2.2

(following Equation 9.2). This reduces the number of candidate record pairs to be processed in the iterative classification phase.

Blocking techniques [30] can be applied before performing the linkage (for example, phonetic-based blocking [103] or our SNC-based private blocking approaches presented in Chapters 6 and 7) such that similar records are grouped together. This further reduces the number of candidate record pairs, because only the pairs that are in the same block will be considered as candidate record pairs. *Alice* and *Bob* each individually applies a private blocking function to their databases (using another set of quasi-identifier attributes or part of the linkage attributes as the blocking keys) to privately identify the list of common blocks in both databases or to generate the resulting candidate record pairs from both databases.

The iterative pruning of candidate record pairs using Bloom filters allows removing pairs that have higher probability of being non-matches before exchanging more bit positions. The aim of our iterative method is to prune the record pairs that are classified as non-matches or matches and thereby reduce the number of unclassified pairs (possible matches) in each iteration as much as possible. We proposed to reveal $min(c_{min} - c_{i-1})$ bits in each iteration. Experiments conducted on real-world datasets (see Section 9.5) show that although many bits are being revealed in the first few iterations, only a small number of bits are being revealed in the later iterations which requires many iterations to run and thus makes the process not scalable to large datasets.

To overcome this problem, we propose a method for revealing more bits when the number of bits to be revealed becomes very small, without compromising privacy. Assume t_i bits have been revealed so far up to iteration i, among which c_i common 1-bits have been found in a record pair which needs $c_{min} - c_i$ more common 1-bits in both Bloom filters in order to be classified as a match. If $c_{min} - c_i$ is very small and hence we can classify the pair as a match even if no more common 1-bits are found in the later iterations, then this pair will not be at a privacy risk if more bits are revealed in the next iteration (because it has already been considered as a match). The question now arises what is the maximum value for $c_{non} = c_{min} - c_i$ that can be ignored to classify pairs as matches without accuracy loss.

$$\begin{aligned} s_t - s_r &= \frac{2(c_{non})}{x^A + x^B} \\ r_{min} &= min(c_{non}) \\ r_i &= min(r_i, r_{min}) \end{aligned} \tag{9.4}$$

We introduce another similarity threshold value, s_r, to calculate the value for the minimum number of bits that can be revealed for each pair in an iteration, r_{min}, as shown in Equation 9.4. This basically expands the calculation of value r_i in line 5 of Algorithm 9.2 as below. Among the values for c_{non} for all the pairs, the smallest value is taken to be used as the value for the minimum number of bits that can be revealed in all the pairs of Bloom filters in an iteration, $r_{min} = min(c_{non})$.

If r_i becomes less than r_{min} in an iteration, especially in later iterations, then r_{min} bits will be revealed. It is important to note that the similarity threshold to reveal, s_r, is only used to calculate the value for r_{min} while the similarity threshold s_t is used to classify the pairs. This approach improves the efficiency (and thereby scalability) of the protocol significantly without compromising the privacy of the non-matched record pairs. This is empirically evaluated in Section 9.5.

9.3 Analysis of the Protocol

In this section we analyze our 2P-BF approach in terms of complexity, privacy, and linkage quality.

9.3.1 Complexity

We assume both databases contain n ($n = n^A = n^B$) records, the average number of q-grams in each record is n_q, the number of hash functions used to map q-grams of a record into a Bloom filter is k, and the length of Bloom filters is l. In phase 1 (the preparation phase) of our protocol, the agreement of the parameters and functions between *Alice* and *Bob* has a constant communication complexity. Hash-mapping the n_q q-grams in each of their n records into Bloom filters using k hash functions has a computation cost of $O(n * n_q * k)$ hash operations for each.

Alice and *Bob* then exchange in the second phase (i.e., the length filtering phase) their $\mathbf{O}^A$ and $\mathbf{O}^B$ lists, respectively, that contain n records with their record identifiers and the number of 1-bits in the records' Bloom filters. This has a communication complexity of $O(n)$, and computing the number of 1-bits differences between each pair of records from *Alice* and *Bob* for length filtering (as explained in Section 9.2.2) is of $O(n^2)$ computation complexity.

The length filtering phase reduces the number of candidate pairs by pruning potential non-matches based on their lengths (number of 1-bits). Further applying a private blocking protocol (such as phonetic based [103] or our SNC-based private blocking presented in Chapters 6 and 7) can reduce more number of candidate pairs by grouping similar records into the same block.

In the third iterative classification phase, *Alice* and *Bob* iteratively exchange bits from (at most) each of their n Bloom filters of l length (if no blocking or length filtering is applied). The computation cost of this phase is $O((n * l)^2)$ bit comparisons, while the communication cost is $O(n * l)$. The communication complexity of this protocol is therefore linear in the size of the databases.

9.3.2 Privacy

As in all three previous chapters, we assume that both *Alice* and *Bob* follow the honest but curious (HBC) adversarial model [78], in that the parties are curious and they try to find out as much as possible about the other party's data while following the protocol. In this section, in order to analyze the privacy of our solution, we discuss

what the two parties can learn from the exchanged data between them during the iterative protocol. In the length filtering phase (Section 9.2.2), *Alice* and *Bob* exchange the number of 1-bits in their Bloom filters, $\mathbf{O}^A$ and $\mathbf{O}^B$, respectively. This might leak some information regarding the presence of uncommon (infrequently occurring) shorter or longer tuples of linkage attribute values in their databases that are mapped into the Bloom filters (not the actual attribute values of tuples). This can be overcome by noise addition or simulation techniques, as will be described further below.

In the iterative classification phase (Section 9.2.3), they iteratively reveal bits from their Bloom filters to each other. The amount of privacy provided by any Bloom filter based PPRL protocol depends on the number of hash functions used (k) and the length of the Bloom filter (l) [122, 174]. The values for k and l have to be carefully chosen as these values provide a trade-off between the quality of the classification and privacy. The higher the value for k/l, the higher the privacy (as empirically validated by Kuzu et al. [122]) and the lower the quality of linkage, because the number of q-grams mapped to one single bit increases, which leads to lower linkage quality but makes it harder for an attacker to infer the possible combinations.

The privacy of our iterative protocol depends mainly on the number of bits revealed and how they are revealed. We propose to reveal $r_i = min(c_{min})$ bits in each iteration i without compromising privacy and complexity. Assume the minimum number of bits required to be revealed in order to re-identify the revealed bit pattern based on a linkage attack using an external dataset (as will be described in Section 9.4) is t_a. The privacy characteristics provided by our protocol (which will be empirically evaluated in Section 9.5) are:

1. Non-matching record pairs are removed in the earlier iterations when only a small number of bits t_i have been revealed ($t_i < t_a$), which therefore cannot be used to re-identify records using a linkage attack (Figures 9.16(a) and 9.19).
2. More bits are revealed for pairs that are more likely to be matches (Figure 9.19).
3. When a sufficient number of bits t_a are revealed for a linkage attack (in iteration i), the remaining unclassified pairs have a minimum similarity, as calculated in Equation 9.5, that is close enough ($sim_{min}(\mathbf{C}_i) \approx s_t$) to be considered as matches (Figure 9.16).

$$sim_{min}(\mathbf{C}_i) \quad = \quad min\left(\forall_{p \in \mathbf{C}_i} \frac{2 \times (c_{min} - c_i)}{x^A + x^B}\right) \tag{9.5}$$

where c_{min}, c_i, x^A, and x^B are calculated for every pair p in the set of candidate pairs $\mathbf{C}_i$ in iteration i.

Pruning candidate record pairs that have a higher probability of being classified as non-matches (based on c_{min} calculated in Equation 9.3) at early iterations improves the privacy of the protocol, since the non-matches are removed without revealing more bits in the next iterations. Only the pairs that have a higher probability to

be classified as matches, exhibit a higher probability of a linkage attack when the number of revealed bits increases. We will empirically evaluate this in Section 9.5. In addition, hash-mapping several attribute values from each record into one compound Bloom filter (CLK encoding [175]) makes it harder for an attacker to infer individual attribute values that correspond to a revealed bit pattern.

The security parameter r_{max}, which is the maximum number of bits to be revealed in the Bloom filters, is agreed upon by the two database owners. This parameter determines the privacy of the protocol. For any given Bloom filter length l, a larger value of r_{max} results in lower privacy but more record pairs are being classified as matches and non-matches, while a smaller value of r_{max} will lead to a smaller number of record pairs being classified but with a higher level of privacy. The database owners can individually simulate a linkage attack (as will be described in Section 9.4) on their own databases to calculate the probability of suspicion with the number of bits revealed in order to agree upon an appropriate value for r_{max}. This is empirically explained in Section 9.5.

Depending on the data and the distribution of 1-bit patterns, another privacy issue to be considered with our protocol is that revealing some bits (that have comparatively high sensitive information due to a small number of q-grams that are mapped to those bits) are susceptible to a linkage attack. We propose two methods for overcoming the problem of revealing the rare or sensitive bits in Bloom filters that can be attacked with higher probability.

1. **Adding noise**: Noise can be added to Bloom filters by converting 1s into 0s and 0s into 1s individually by the database owners in the preparation phase in order to perturb their datasets. Noise bits can either be added randomly or they can be selectively added depending on the sensitivity of the bits. In the selective noise addition method, bits that have high frequency (occur in many Bloom filters) and that have more q-grams mapped to them can be added (by setting 0s to 1s), and bits with low frequency and that have a smaller number of q-grams mapped to them can be removed (by setting 1s to 0s) from the Bloom filters. This is similar to the approach by Durham [56], where bit frequency ranges are examined to eliminate sensitive bits from the field-level Bloom filters that can be mapped back to less than a certain number of fields, when composing the record-level Bloom filters. Privacy is improved by removing less frequent bits, while the loss of quality that occurs due to the noise addition is reduced by adding high frequent bits.

   The question is how many noise bits need to be added or removed in order to increase the privacy without compromising linkage quality and complexity. When adding noise three cases can occur. The first is when the bits added or removed (flipped 0s to 1s and 1s to 0s, respectively) by the two database owners lead to the same number of additional matching 1-bits at the same positions (common 1-bits), which results in almost the same similarity value. The second case is where only some of the flipped bits are matching with the already existing bits in the other database owner's Bloom filters, and thus the

number of additional common 1-bits introduced by the noise bits is lower than the total number of noise bits added by the database owners. The third case occurs when the added noise bits do not match with any existing 1-bits and thus no additional common 1-bits are introduced by the noise bits.

In both the second and third cases, the new similarity value decreases because of the noise bits. However, the third case is the worst case and needs to be considered in determining the similarity threshold value. The database owners must agree on a minimum acceptable lower bound of the similarity threshold, s_l, with $s_l < s_t$. If the values for x^A_{min}, x^B_{min}, s_t, and s_l are known, then the maximum number of noise bits that can be added by the database owners, z_{max}, can be estimated under the worst case (the third case described above, i.e. no additional common 1-bits are introduced due to adding z_{max} noise bits) using Equation 9.6.

$$\begin{aligned} s_t &= \frac{2 \times c_{min}}{x^A_{min} + x^B_{min}} \\ s_l &= \frac{2 \times (c_{min} + 0)}{(x^A_{min} + z_{max}) + (x^B_{min} + z_{max})} \\ s_l &= \frac{s_t \times (x^A_{min} + x^B_{min})}{(x^A_{min} + z_{max}) + (x^B_{min} + z_{max})} \\ z_{max} &= \left\lfloor \frac{(s_t - s_l) \times (x^A_{min} + x^B_{min})}{2 \times s_l} \right\rfloor \end{aligned} \tag{9.6}$$

Figure 9.8 shows the maximum number of noise bits (z_{max}) that can be individually added to each Bloom filter by the database owners to perturb the bit distribution in Bloom filters against the minimum similarity threshold value that is acceptable (s_l) without much quality loss in the classification results. The maximum number of noise bits linearly increases when the minimum similarity threshold value decreases.

2. **Simulation attack**: The database owners can each individually simulate the protocol and attack their own database before exchanging the values in order to identify if there exist any sensitive bits that map only to a small number of q-grams. Based on that, they can either change the values for k, l, and q, or they can agree on an appropriate value for the security parameter r_{max}. The bit distribution in Bloom filters (based on the number of q-grams mapped) in a real Australian telephone database (OZ dataset, as described in Section 5.4) with 17,294 records, for example, shows that an average of 22 q-grams ($q = 2$) and a minimum of 14 q-grams are mapped to one single bit when $k = 30$ and $l = 1000$, while when $k = 30$ and $l = 200$ the bit distribution has a minimum of 88 and an average of 101 q-grams mapped to the bits (Figure 9.9). The number of q-grams mapped to one bit decreases with l while it increases with k.

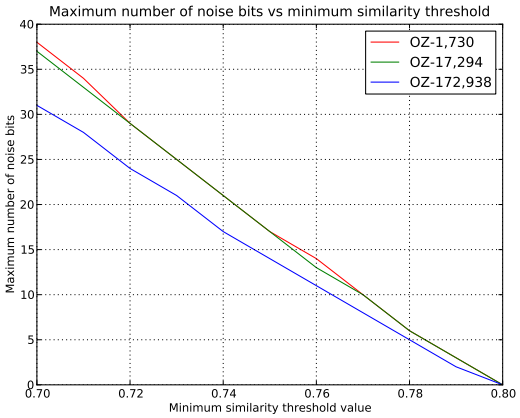


Figure 9.8: Maximum number of noise bits z_{max} that can be added against the minimum lower bound s_l of the similarity threshold value, with $s_t = 0.8$.

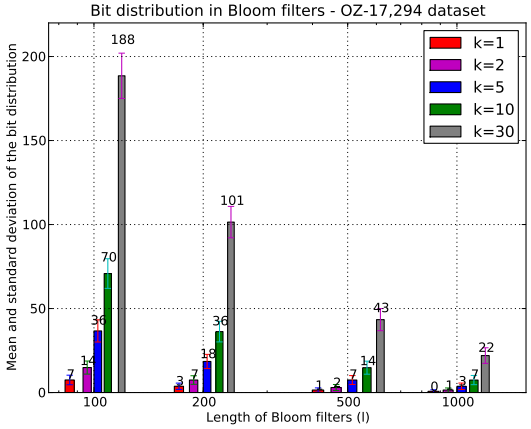


Figure 9.9: Bit distribution (number of q-grams mapped to the bits) in Bloom filters in the OZ-17,294 dataset for different values of l and k. The numbers shown are mean values.

9.3.3 Linkage Quality

Similar as with the three-party Bloom filter-based solutions [56, 174, 175], our 2P-BF approach's linkage quality is dependent on the Bloom filter parameterization and the encoding method (as will be discussed further below). In order for a string s_1's Bloom filter b_1 to be considered as a match with another string s_2's Bloom filter b_2, the bit positions that are set to 1 in b_1 when hash-mapping the sub-strings (q-grams) of s_1 into b_1 using k hash functions, must have been set to 1 in b_2 as well. However, there can be false positives due to collisions between bits (i.e., two different q-grams of s_1 and s_2 are hash-mapped into the same bit and classified as a match). Equation 9.7 shows the probability f that a classification of two Bloom filters into a match (s_1 matches with s_2) becomes a false positive. From the equation, we can see that the probability of false positive depends on the length of Bloom filters (l), the number of hash functions k, and the number of elements n_q (q-grams) in the string [144]. This probability of false positives f provides privacy in privacy-preserving solutions at the cost of linkage quality loss. Therefore, the parameter values (l and k) have to be chosen carefully to balance this trade-off between privacy and linkage quality.

$$f = (1 - e^{-kn_q/l})^k \tag{9.7}$$

Several Bloom filter encoding methods have been proposed in the literature [56, 174, 175]. As discussed in Section 9.1, hash-mapping the q-grams of all the linkage attribute values of a record into one Bloom filter is known as cryptographic longterm key (CLK) [174, 175] encoding.

Durham [56] studied this approach in detail by using record-level Bloom filter encoding (RBF) to improve the quality of linkage based on the weights of the link-

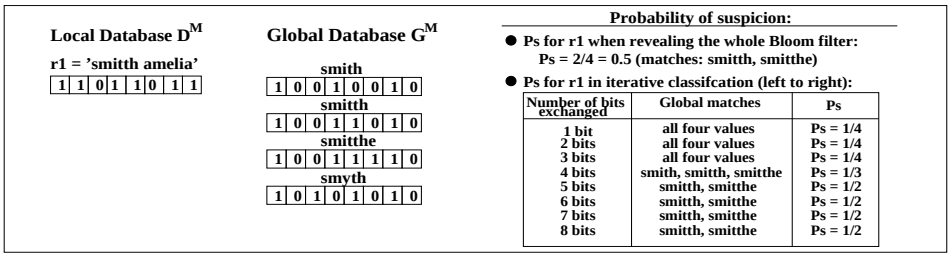


Figure 9.10: An attack method for Bloom filter-based private matching and classification solutions (taken from [190]). As the membership theory states [122], all the bit positions that are set to 0 in the Bloom filter of record r_1 must also be set to 0 in the Bloom filters in $\mathbf{G}^M$ that are possible matches to r_1. Hence, in the shown example two of four global values' Bloom filters ('smitth' and 'smitthe') in $\mathbf{G}^M$ match with the Bloom filter of r_1 and therefore $P_s = 1/2$. In the iterative classification approach, P_s increases with more bits revealed.

age attributes. In her approach, first the attribute values (q-grams) are hash-mapped into different Bloom filters (field-level Bloom filters). Then bits are selected from each of the attributes' (field-level) Bloom filters according to their weights calculated based on Fellegi and Sunter's agreement and disagreement weights [65] (more bits are selected from attributes with higher weights) and frequencies (bits with certain frequencies are not included to improve privacy) in order to compose the RBF. Random shuffling of bits is also used by the database owners in order to hide the order of bits in the Bloom filters (RBF) from the third party (however, it is is not applicable in two-party solutions).

In a hybrid encoding we can combine both CLK and RBF (which we call CLKRBF) to select different numbers of hash functions k for different attributes according to their weights and map them into the same Bloom filter of length l. Having different numbers of hash functions for different attributes based on weights provides more linkage quality as with RBF [56], and mapping them into the same Bloom filter improves privacy due to collisions between bits as with CLK [175]. In Chapter 10 we will empirically evaluate and compare these encoding methods in our 2P-BF solution with respect to the linkage quality and privacy.

9.4 Linkage Attack

In order to check if a q-gram $q_j (q_j \in s, 1 \leq j \leq n_q)$ is a member of a Bloom filter b, all the k integer values (positions in the Bloom filter) returned by the hash functions $h_i(q_j), 1 \leq i \leq k$ should be set to 1 in b [122]. If at least one of the bits (returned integers) is set to 0 in b, then q_j cannot be a member of b.

A simple example of the linkage attack method and the calculation of DR measures for Bloom filter-based private matching and classification is presented in Figure 9.10. The main idea of a cryptanalysis attack [122] is that if a bit position is set to 0 in a Bloom filter, then all the possible matches (members or sub-strings of the string which is mapped to this Bloom filter) must not independently set the specific bit position to 1, as proven in [122]. In our 2P-BF, the probability of suspicion (P_s)

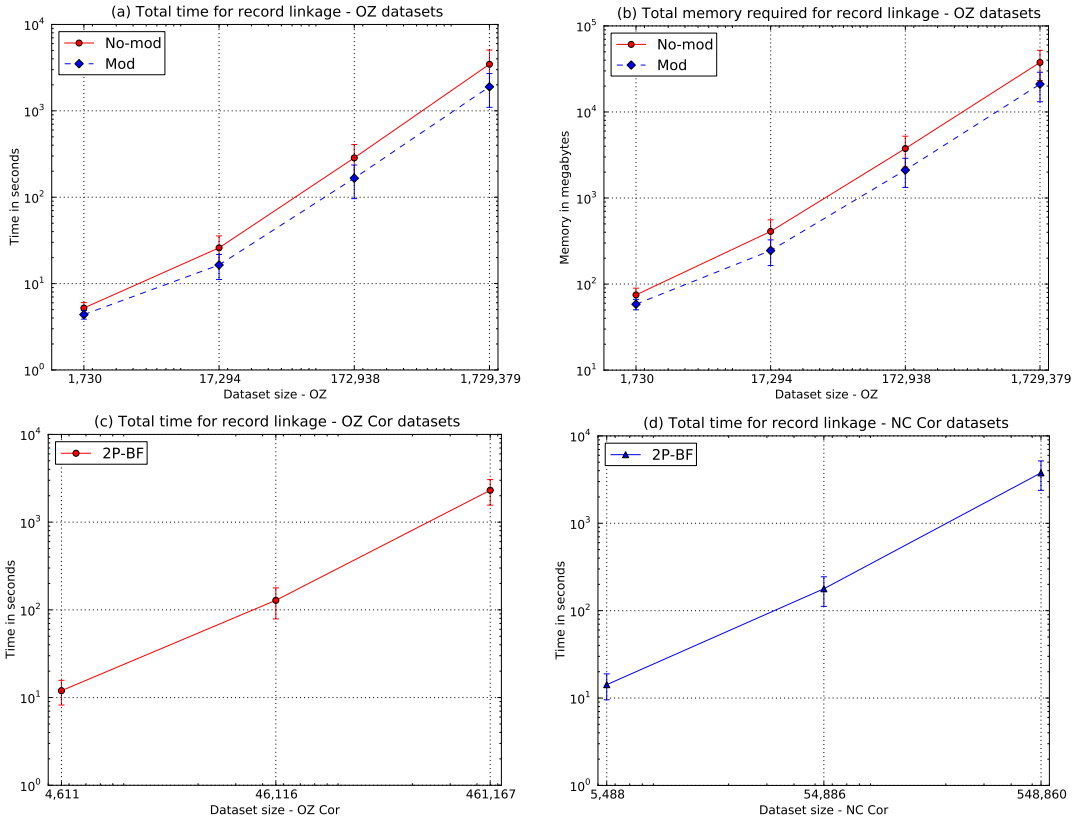


Figure 9.11: (a) Total linkage time, and (b) total memory size required for the 2P-BF approach on the OZ datasets, (c) total linkage time on the OZ Cor datasets, and (d) total linkage time on the NC Cor datasets averaged over the results of all variations of each dataset.

increases with the number of bits revealed, as shown in Figure 9.10. We did not consider error bounds in our attack methods that allow for approximate matching errors. Developing attack methods for randomized masking [67] with error bounds in Bloom filter-based PPRL solutions is left out for future work.

9.5 Experimental Evaluation

In this section we present the results of the empirical study of our 2P-BF approach conducted on the datasets described in Section 5.4 using the evaluation framework proposed in Chapter 5. Following previous work [174], we set the values for the Bloom filter parameters as $l = 1000$, $k = 30$, and $q = 2$. The minimum similarity threshold to classify was set to $s_t = 0.8$ and the threshold to reveal bits was set to $s_r = 0.77$. All four attributes in the datasets were used as linkage attributes.

Figure 9.11 shows the total linkage time and memory size required for private matching and classification of the 2P-BF approach on different datasets. The result figures exhibit an almost linear complexity trend in the size of the databases which makes the protocol scalable to large databases.

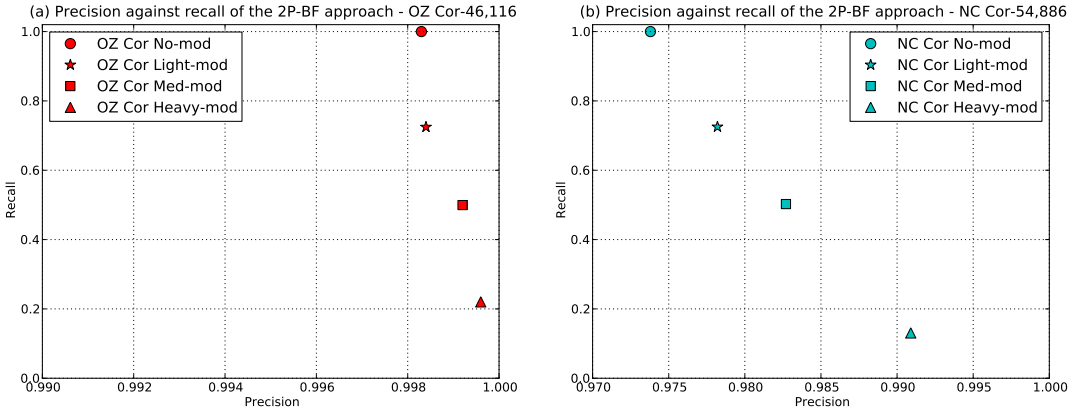


Figure 9.12: A comparison of precision against recall of the 2P-BF solution on the (a) OZ Cor-46,116 and (b) NC Cor-54,886 datasets with No-mod, Light-mod, Med-mod, and Heavy-mod variations.

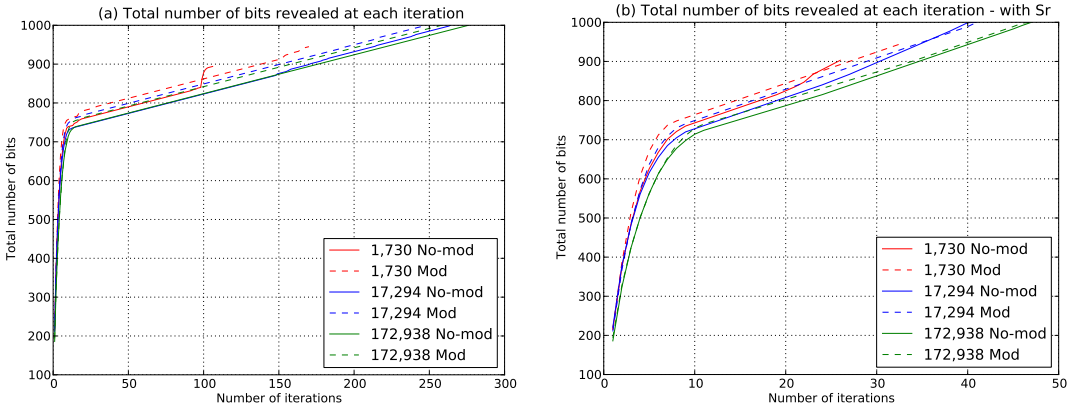


Figure 9.13: Total number of bits revealed at each iteration (a) without the similarity threshold to reveal, s_r, and (b) after introducing the threshold s_r (as described in Section 9.2.4) on the OZ datasets. The values for the similarity thresholds were set as $s_t = 0.8$ and $s_r = 0.77$.

A comparison of precision and recall of the 2P-BF protocol is presented in Figure 9.12 on the OZ Cor-46,116 and NC Cor-54,886 datasets with different levels of data modifications (corruptions). Different levels of modifications applied to the datasets (as was described in Section 5.4) allow us to evaluate the performance of approximate matching of our protocol in the presence of data errors. As shown in the figure, the 2P-BF approach achieves high precision and recall when no modification is applied to the datasets, and then recall drastically decreases while precision slightly increases as the level of modifications increases. This is because the number of false non-matches increases while the number of matches decreases due to the modifications applied.

As discussed in Section 9.2.4, the number of bits revealed in the later iterations is very small and therefore it takes more iterations to classify all the record pairs into

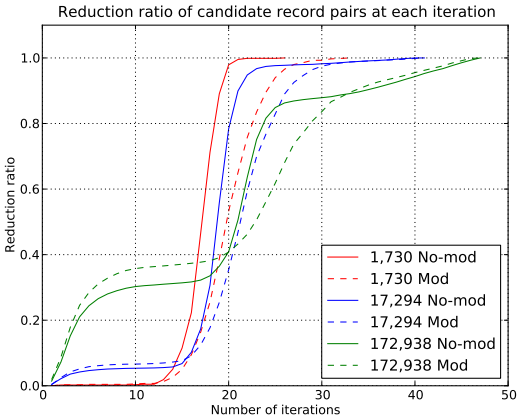


Figure 9.14: Reduction ratio of classified pairs at each iteration on the OZ datasets.

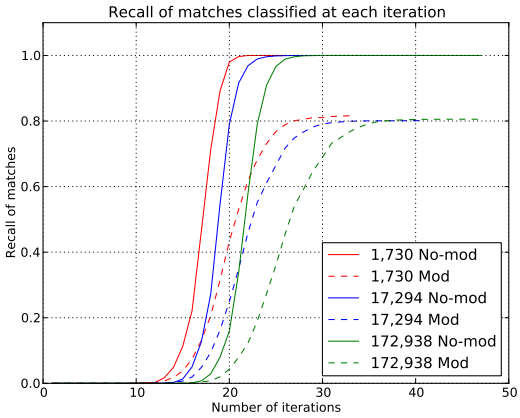


Figure 9.15: Recall of matches at each iteration on the OZ datasets.

matches and non-matches (see Figure 9.13(a)). With the proposed method of using a second similarity threshold, $s_r = 0.77$, this has been significantly improved, as shown in Figure 9.13(b). The total number of iterations required to classify all the record pairs is reduced 6-fold (from 300 to 50 iterations on the largest dataset - OZ-172,938) with the proposed approach using a second threshold $s_r = 0.77$.

The reduction ratio (RR) of record pairs with unknown match status after classifying record pairs as 'matches' and 'non-matches' at each iteration is shown in Figure 9.14. As can be seen from the figure, our approach shows a high increment rate in the reduction ratio after the first few iterations. The recall ratio (as shown in Figure 9.15) is almost 1.0 for the datasets with no modifications ('No-mod'). It is high (nearly 0.8) with modified datasets as well (a total of 8 edits per record that results in almost 50% modifications in the corresponding q-grams), which explains the aspect of fault-tolerance to data errors by performing approximate matching.

The privacy characteristics of this protocol (as discussed in Section 9.3.2) are empirically evaluated in Figure 9.16(a) based on the attack method proposed in Section 9.4 using the Australian telephone database (OZ) as the global dataset. This study empirically validates that the probability of a linkage attack increases with the number of bits revealed, and the maximum probability of an attack (maximum disclosure risk) becomes greater than 0.05 (i.e. $> 1/20$) only after 800 bits have been revealed. However, when 800 bits are revealed, most of the non-matching record pairs have already been removed (as can be seen from Figure 9.19), and the minimum similarity value of the remaining record pairs is nearly 0.7 (illustrated in Figure 9.16(b)), which assures that the privacy of non-matches with similarity less than 0.7 is not compromised with the iterative pruning approach.

We also tested noise addition techniques into the Bloom filters to perturb the bit distributions (as discussed in Section 9.3.2) that improve privacy against linkage attacks at the cost of quality loss. The minimum accuracy loss threshold was set to $s_l = 0.78$, so that a maximum of $z_{max} = 6$ noise bits can be included (see Figure 9.8).

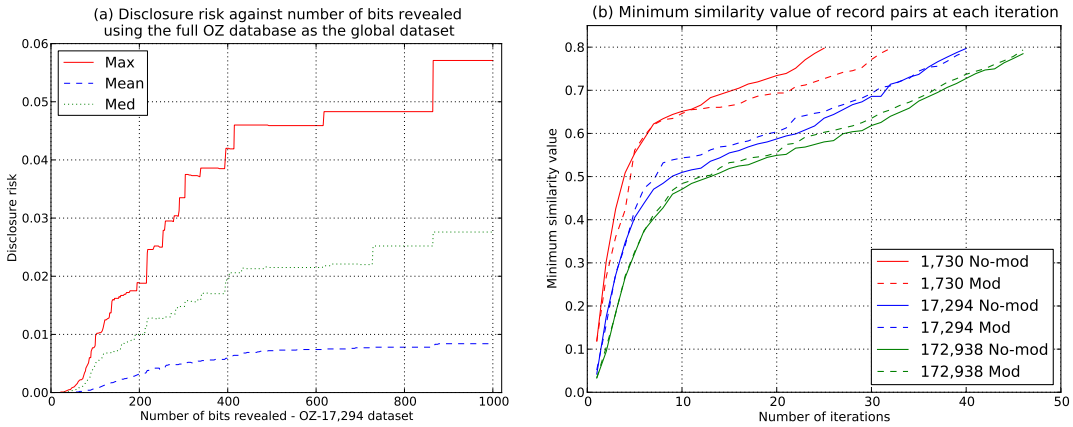


Figure 9.16: (a) Disclosure risk values against the number of bits revealed on the OZ-17,294 dataset and (b) the minimum similarity value of unclassified record pairs at each iteration on the OZ datasets.

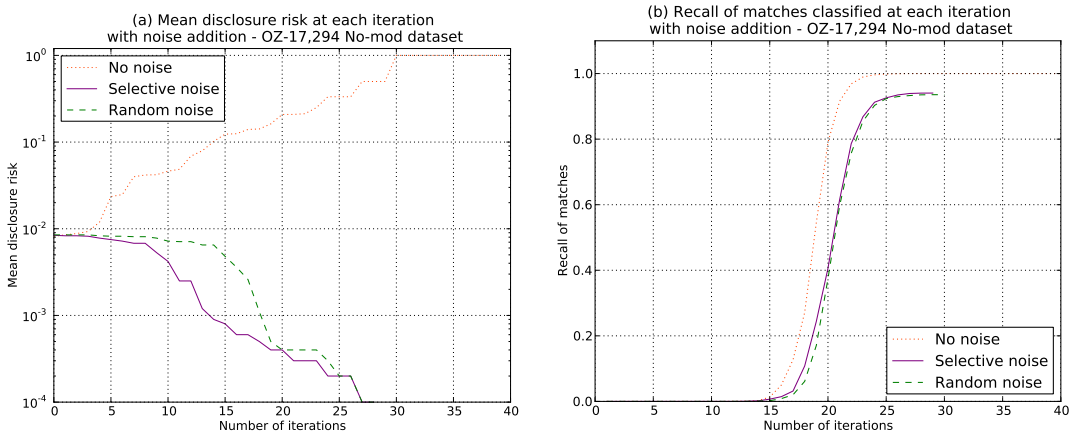


Figure 9.17: (a) Disclosure risk and (b) recall of matches at each iteration for noise addition techniques on the OZ-17,294 No-mod dataset.

Noise bits are added or removed (flipped) either randomly ('random', i.e. bits are flipped with the probability of z_{max}/l) or selectively according to the sensitivity of the bits ('selective', i.e. bits with a smaller number of q-grams mapped to them are highly sensitive compared to bits with a larger number of q-grams mapped and thus they are less likely to be flipped). As shown in Figure 9.17, the noise addition techniques reduce the disclosure risk significantly at the cost of some loss in recall. When no noise is added disclosure risk increases with bits revealed, since more information is revealed with more bits and thus a smaller number of global values match with longer bit patterns. However, when noise bits are added into or removed from the Bloom filters the number of global values that match with the perturbed bit patterns becomes zero with more bits, and thus the disclosure risk decreases as more bits are revealed. Disclosure risk is reduced more with the selective noise addition technique than with the random noise addition at the cost of similar loss in recall.

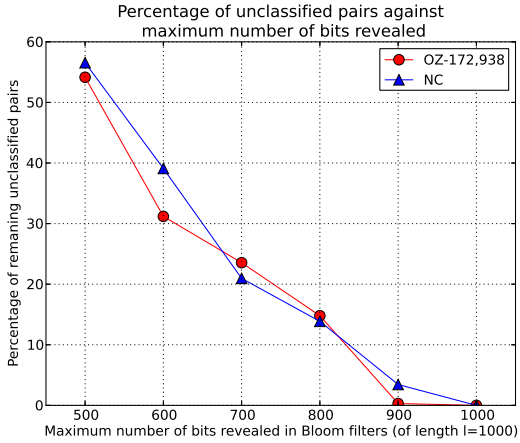


Figure 9.18: Percentage of remaining unclassified record pairs in the class of possible matches against different values of r_{max} on the OZ-172,938 and NC datasets.

Table 9.2: Blocking combined with the 2P-BF private matching and classification solution on the OZ-1,730 Mod dataset.

	No blocking	Phonetic	SNC-2P
Time (seconds)	173.92	**6.6233**	15.1179
Precision	0.9208	**1.0000**	0.9972
Recall	**1.0000**	0.7680	0.9504
F-measure	0.9588	0.8688	**0.9732**
DR_{Mean}	**0.0010**	0.9909	0.0217
DR_{Mark}	**0.0000**	0.9908	0.0046

Figure 9.18 shows the percentage of remaining record pairs that are unclassified after r_{max} bits have been revealed from the Bloom filters for different values of r_{max}. Around 50% of record pairs are classified into matches and non-matches when 50% ($r_{max} = 500$, $l = 1000$) of the bits have been revealed, and when $r_{max} = 800$ only 15% of pairs remain unclassified (in the class of possible matches which need to be re-hashed with different Bloom filter parameters in order to re-conduct the iterative classification process, as was explained in Section 9.2.3).

As can be seen from Figure 9.19, many non-matches are being classified in the first few iterations and then matches are being classified more towards the middle and later iterations. The overall RR of candidate record pairs is thus high (as shown in Figure 9.19(c)), while the recall ratio of matches being classified is also high (as was shown in Figure 9.15).

Finally, similar as in the previous chapter, we studied how a private blocking solution combined with our 2P-BF private matching and classification solution determines the three properties of scalability, quality, and privacy. We evaluated the 2P-BF private matching and classification solution with no blocking, Soundex [23]-based phonetic blocking (a standard blocking approach that has been used in non-PPRL, as described in Chapter 2), and our SNC-based private blocking proposed in Chapter 7. In Table 9.2, we present the total time required for blocking and linkage,

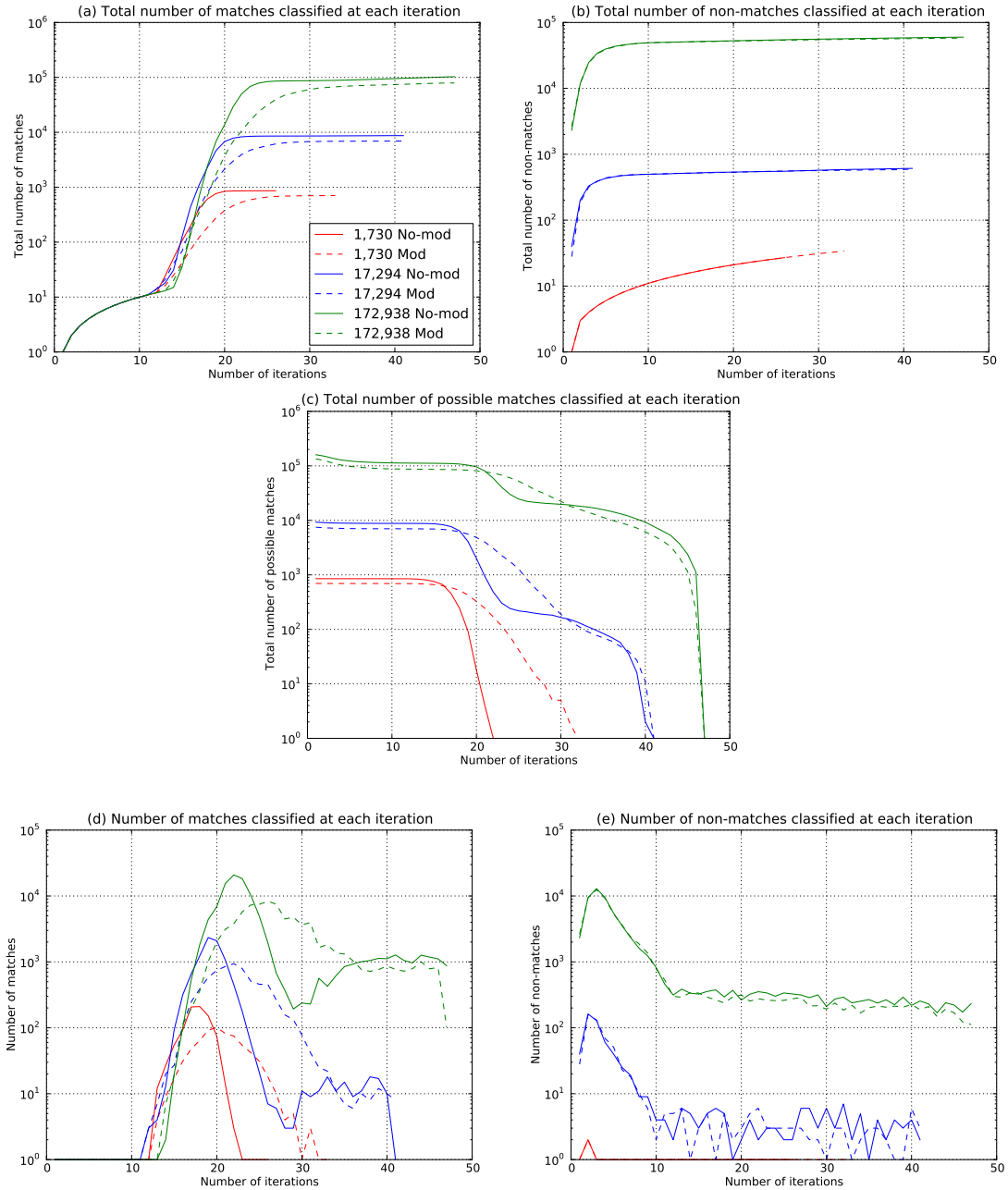


Figure 9.19: (a) Total number of record pairs classified as matches, (b) non-matches, (c) possible matches, and (d) number of record pairs classified as matches, and (e) non-matches at each iteration on the OZ datasets.

linkage quality results, and the DR measures in the worst case setting ($\mathbf{G} \equiv \mathbf{D}$) of our 2P-BF solution with these three blocking scenarios. As the results show, when no blocking is applied the DR values are very low. However, it requires significantly longer linkage time compared to when a blocking technique is applied. Phonetic-based blocking requires shorter time than our SNC-based blocking, though privacy

and linkage quality results are comparatively better with the SNC based approach. Phonetic-based blocking provides lower privacy guarantees.

9.6 Summary

In this chapter we have proposed a practical two-party private matching and classification solution for privacy-preserving record linkage (PPRL) by addressing the three main challenges, which are scalability to large databases, high linkage quality results, and sufficient privacy characteristics. With the appropriate determination of values for the parameters, the experimental studies conducted on real-world datasets show that our proposed two-party protocol can perform efficient linkage with high linkage quality while providing adequate privacy characteristics.

Learning other advanced Bloom filter encoding methods to improve the linkage quality without compromising the privacy in this two-party protocol is one interesting direction for future work. Another avenue of future work is to develop efficient linkage attack methods with approximation of error bounds for privacy evaluation of Bloom filter-based private matching and classification solutions.

As we identified in Chapter 4, conducting a comprehensive empirical study of different PPRL solutions is an important avenue of research in PPRL. In the next chapter, we will empirically compare and evaluate our proposed solutions with some of the state-of-the-art solutions using our evaluation framework proposed in Chapter 5.

Chapter 10

Comparative Evaluation

We have addressed some of the shortcomings identified in Chapter 4 by proposing novel and practical solutions for privacy-preserving record linkage (PPRL) in Chapters 6 to 9. In this chapter, we comparatively evaluate the proposed solutions with some of the state-of-the-art solutions using the evaluation framework proposed in Chapter 5. We first present the comparative empirical evaluation results of several private blocking solutions in Section 10.2, and then we present the results of private matching and classification solutions in Section 10.3. Finally, we discuss our findings in Section 10.4 and summarize our evaluation in Section 10.5.

10.1 Introduction

Developing novel algorithms for viable real-world PPRL applications that address the three key challenges (or properties) of PPRL, which are scalability, linkage quality, and privacy, is an important research problem in PPRL. We have proposed several novel algorithms in Chapters 6 to 9 for PPRL addressing the three challenges.

The general pipeline of the PPRL process of two data sources is outlined in Figure 10.1. We described the steps of this process and their challenges in a privacy-preserving setting in Chapter 2. The scalability challenge of PPRL can be addressed by using two-step algorithms, where in the first step (Step 1 in Figure 10.1) a blocking or indexing technique is applied to reduce the number of candidate record pairs that need to be compared [30]. These candidate record pairs are then compared and classified into matches and non-matches in the second step (Step 2 in Figure 10.1) using approximate and effective private matching and classification techniques (addressing the linkage quality challenge). Privacy is addressed by applying a masking function (as described in Chapter 5) to the linkage attributes and / or blocking keys in such a way that besides the record pairs classified as matches no other sensitive information is revealed to any internal or external parties, using privacy techniques as detailed in Chapter 4.

We proposed efficient private blocking solutions (three-party and two-party) based on the sorted neighborhood clustering in Chapters 6 and 7. We then proposed two different two-party solutions for private matching and classification in PPRL in Chapters 8 and 9. Efficient perturbation-based privacy techniques such as k-anonymous

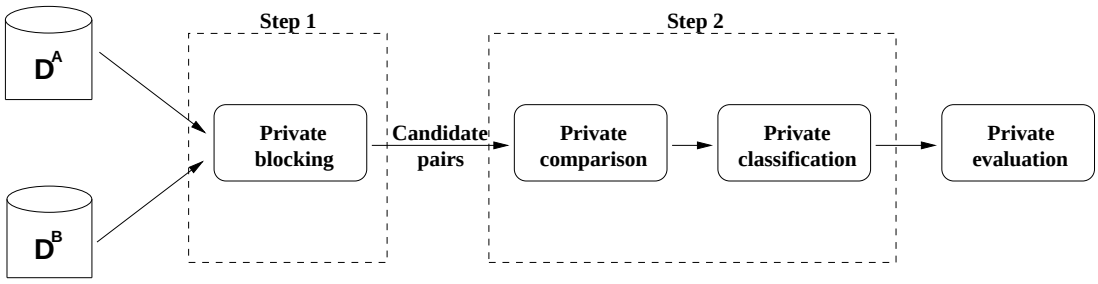


Figure 10.1: A general outline of the privacy-preserving record linkage pipeline of two databases (taken from [190]).

mapping [72, 182], reference values [154], Bloom filters [17], and random noise [108] are used in these proposed solutions. In addition, the use of efficient perturbation-based privacy techniques reduces the computation and communication complexities of the algorithms and thus improves the scalability.

In the following, we compare and evaluate our proposed solutions with some of the state-of-the-art solutions using our evaluation framework (which was proposed in Chapter 5) in terms of all three properties of PPRL. Figures 10.2 to 10.13 and Tables 10.3 and 10.4 present the results of our empirical study (taken from [190]). We prototyped all the solutions in Python version 2.7.3 (as detailed in Section 5.6), and all the experiments were conducted on the datasets described in Section 5.4.

10.2 Private Blocking Techniques

We comparatively evaluated the scalability, quality, and privacy of the following six private blocking approaches: our sorted neighborhood clustering (SNC)-based three-party private blocking solutions proposed in Chapter 6 (labelled as **SNC-3PSim** for similarity-based merging and **SNC-3PSize** for size-based merging); our SNC-based two-party private blocking solution proposed in Chapter 7 (labelled as **SNC-2P**); Karakasidis et al.'s [104] three-party private blocking based on k-nearest neighbor clustering and reference values (labelled as **k-NN**); Durham's [56] Hamming-based locality sensitive hashing three-party private blocking (labelled as **HLSH**); and Kuzu et al.'s [124] two-party private blocking based on hierarchical clustering and differential privacy (labelled as **HCLUST**). We reviewed the details of **k-NN**, **HLSH**, and **HCLUST** in Chapter 3 (named as Kar12 on Page 31, Dur12 on Page 30, and Kuz13 on Page 31, respectively).

We used parameter settings for the **k-NN**, **HLSH**, and **HCLUST** methods in similar ranges as used by the authors of these private blocking techniques.

- For **k-NN**, k was set to 3 and the minimum similarity threshold was $s_t = 0.6$.
- In the **HLSH** method, the number of iterations was set to $\mu = 40$, the number of hash functions was $k = 30$, the length of Bloom filters was $l = 1,000$ bits,

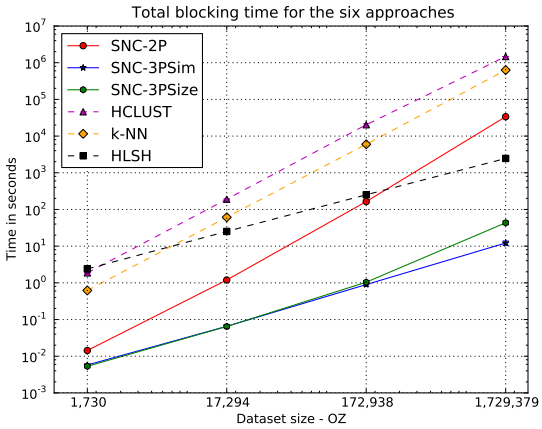


Figure 10.2: A comparison of scalability (evaluated by total blocking time) of the six private blocking approaches on the OZ datasets.

and the number of bits to be sampled from the Bloom filters at each iteration was $\phi = 45$.

- In the **HCLUST** method, the number of clusters was set as one tenth of the number of records in the dataset, the differential privacy parameter $\epsilon = 0.3$, and the fake records tolerance parameter w_n was set as the number of records in the datasets to be linked.

- The default parameters for the **SNC**-based approaches were set as minimum block size $k = 100$, minimum similarity threshold $s_t = 0.8$, and window size $w = 2$ (as discussed in Sections 6.5 and 7.5).

Figure 10.2 shows the scalability of private blocking approaches to different sizes of the OZ datasets measured by total blocking time (averaged over the results of all parties over all variations of each dataset). As can be seen from the figure, the **SNC**-based approaches (**SNC-2P**, **SNC-3PSim** and **SNC-3PSize**) require shorter time than the other approaches and are scalable to large databases. **k-NN** and **HCLUST** take significantly longer blocking time than **HLSH** and the **SNC**-based approaches.

The efficiency of blocking (scalability) measured by reduction ratio (RR) and the effectiveness of blocking (quality) measured by pairs completeness (PC) of the six private blocking approaches are compared on the OZ-172,938 Mod and NC datasets in Figure 10.3. **SNC-2P** achieves the highest PC at the cost of some reduction in RR, while the other approaches comparatively have lower PC with RR being almost 1.0. **HLSH** performs better by achieving high values for both RR and PC. These scalability and quality values for the private blocking approaches are mapped into a RR and PC plot, as shown in Figure 10.4, to compare the solutions in the trade-off of scalability (efficiency) and quality (effectiveness) of blocking. As the figure clearly shows, the **HLSH** approach achieves high values for both scalability and quality, following the **SNC-2P** approach. The **k-NN** blocking approach achieves comparatively lower values for both properties.

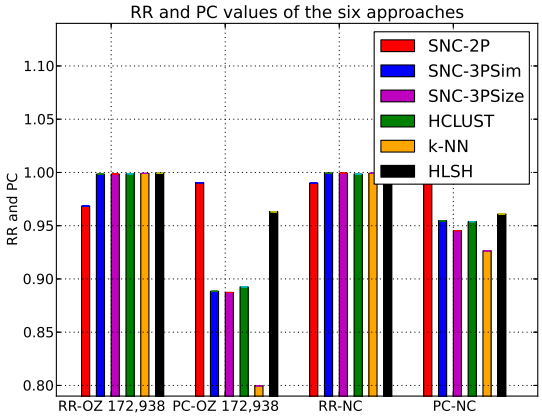


Figure 10.3: A comparison of reduction ratio (RR) and pairs completeness (PC) of the six private blocking approaches on the OZ-172,938 Mod and NC datasets.

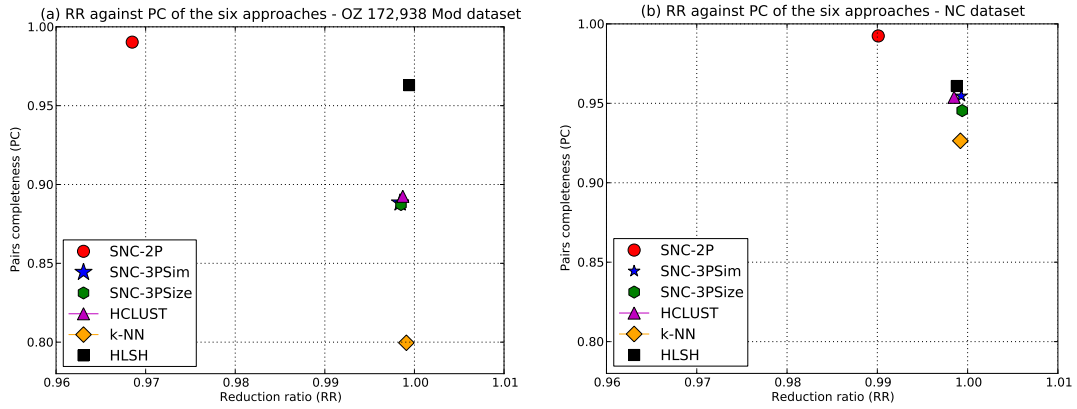


Figure 10.4: Reduction ration (RR) against pairs completeness (PC) of the six private blocking approaches on the (a) OZ-172,938 Mod and (b) NC datasets. The best solutions are the ones closest to the upper right corner.

Finally, the privacy protection of the solutions are evaluated using the disclosure risk measures presented in Section 5.3.1. Due to time and memory constraints, we used the original dataset as the global dataset ($\mathbf{G} \equiv \mathbf{D}$) for privacy evaluation under the worst case assumption. The size of blocks generated by the six private blocking approaches are compared on the OZ-172,938 Mod and NC datasets in a box-and-whisker plot in Figure 10.5. The **SNC**-based approaches and **HCLUST** have lower variances between the block sizes which make a frequency attack using block sizes more difficult. The **HLSH** approach generates overlapping blocks of smaller sizes and the variance between block sizes is comparatively very high. It is important to note that if the third party (in three-party solutions) does not have any information regarding the parameters used and / or if it does not collude with any of the database owners, then trying to mount a frequency attack even with variant block sizes is non-trivial, because the third party does not know the parameter values (that can be used to mount a frequency attack using a global dataset) and therefore learning the actual

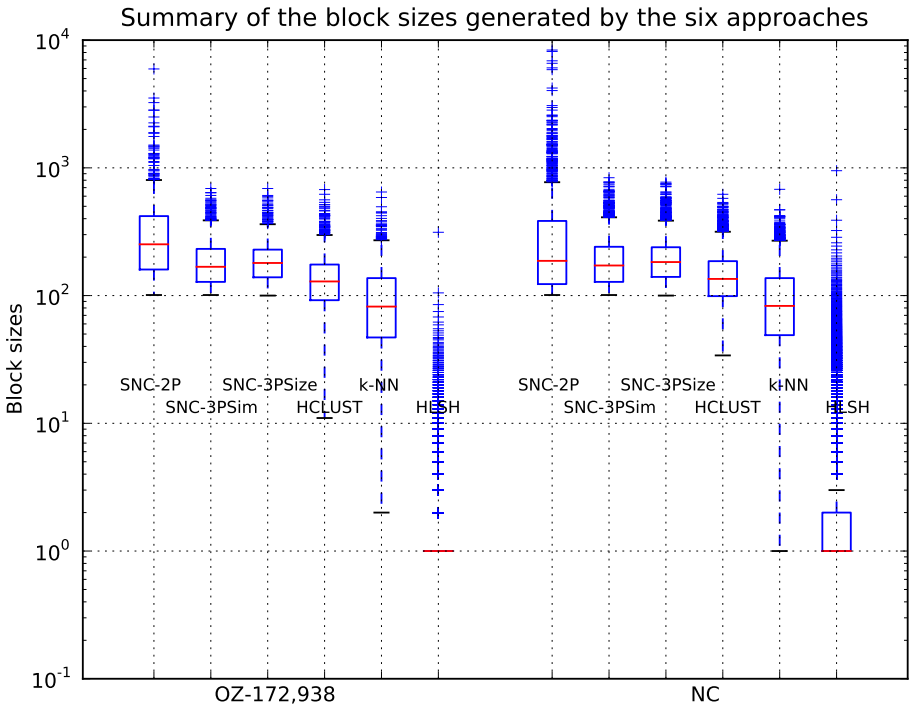


Figure 10.5: A comparison of block sizes generated by the six private blocking approaches on the OZ-172,938 Mod and NC datasets.

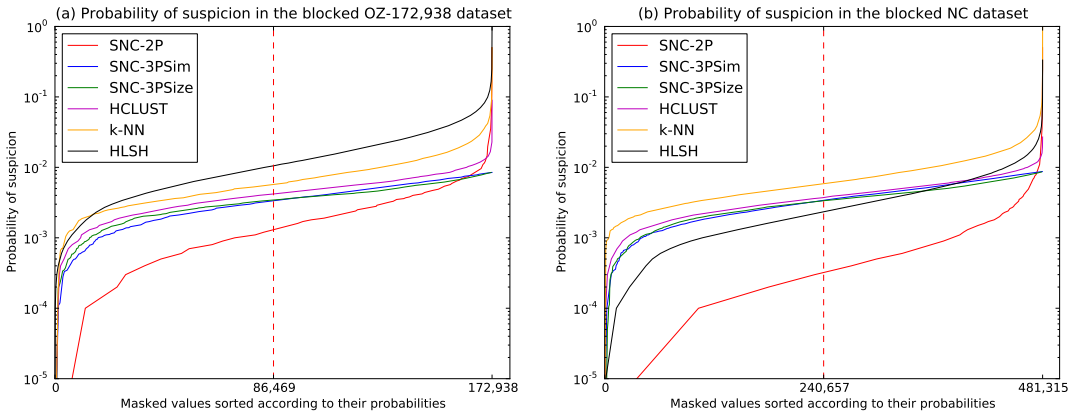


Figure 10.6: A comparison of the distributions of probability of suspicion (P_s) values of the blocked datasets generated by the six private blocking approaches on the (a) OZ-172,938 Mod and (b) NC datasets.

blocking key values (BKVs) in the masked datasets is difficult.

Figure 10.6 shows the distributions of probability of suspicion (P_s) values (similar to the examples illustrated in Figures 5.4, 5.5, and 5.6) in the OZ-172,938 Mod and NC datasets blocked by the six private blocking approaches. The records in the datasets are sorted according to their p_s values. The median line in the blocked datasets (which is used to calculate the median P_s and consequently the median disclosure

Table 10.1: Disclosure risk (DR) measures of the six private blocking approaches. Best values in each row are shown in bold font.

Dataset	DR	SNC-2P	SNC-3PSim	SNC-3PSize	HCLUST	*k*-NN	HLSH
OZ-172,938	Max	0.4999	**0.0085**	**0.0086**	0.0896	0.4999	1.0000
	Mean	**0.0008**	0.0037	0.0036	0.0036	0.0085	0.0067
	Med	**0.0015**	0.0059	0.0031	0.0043	0.0096	0.0092
	RIG	**0.4690**	0.5603	0.5613	0.5365	0.6049	0.9387
NC	Max	0.4999	**0.0087**	**0.0087**	0.0278	1.0000	0.4999
	Mean	**0.0007**	0.0037	0.0036	0.0033	0.0085	0.0015
	Med	**0.0017**	0.0062	0.0068	0.0038	0.0050	**0.0017**
	RIG	**0.5118**	0.6028	0.6031	0.5784	0.6483	0.8870

risk DR_{Med}) is marked by a vertical dotted line in the figures. **SNC-2P** generates the lowest probability of suspicion curve on both datasets. However, its maximum P_s goes higher compared to **SNC-3PSim**, **SNC-3PSize**, and **HCLUST** approaches.

A comparison of disclosure risk (DR) measures (DR_{Max}, DR_{Mean}, and DR_{Med} calculated from the probability of suspicion values P_s, as shown in Figure 10.6 and explained in Section 5.3.1.1, and relative information gain RIG calculated as explained in Section 5.3.1.1) of the six private blocking approaches on the OZ-172,938 Mod and NC datasets is given in Table 10.1. **SNC-2P** has the lowest values for DR_{Mean}, DR_{Med}, and RIG measures. However, DR_{Max} is relatively higher than **SNC-3PSim**, **SNC-3PSize** and **HCLUST** approaches. The disclosure risk values for the **HLSH** and **k-NN** approaches are higher compared to the other approaches, with DR_{Max} being 1.0 or 0.5 where there exists a block with a single or two BKVs.

The trade-off between privacy (measured by DR_{Max}, DR_{Mean}, DR_{Med}, and RIG) and quality (measured by PC) of private blocking solutions is illustrated in Figure 10.7 for all six private blocking approaches on the OZ-172,938 Mod and NC datasets. **SNC-2P** provides the highest PC with reasonably lower DR. Next follow the **SNC-3PSim**, **SNC-3PSize**, and **HCLUST** approaches, which perform better compared to the **k-NN** and **HLSH** ones by achieving higher PC with lower values for DR measures.

10.3 Private Matching and Classification Techniques

In this section, we empirically evaluate the following private matching and classification solutions: our two-party private matching and classification solution based on reference values and binning proposed in Chapter 8 (labelled as **2P-Bin**); our two-party Bloom filter-based private matching and classification solution proposed in Chapter 9 with Schnell's cryptographic longterm (CLK) encoding [175] (labelled as **2P-BF CLK**); Durham's record level Bloom filter (RBF) encoding [56] (labelled as **2P-BF RBF**); and our hybrid encoding of CLK and RBF proposed in Chapter 9 (labelled as **2P-BF CLKRBF**). For the **2P-Bin** [191] solution, the number of bins was used in the range of $k = [4, 6, 8, 10, 12]$ and the minimum similarity threshold was set to $s_t = 0.8$. As in previous work [174, 175], the default parameters for the **2P-BF** [188]

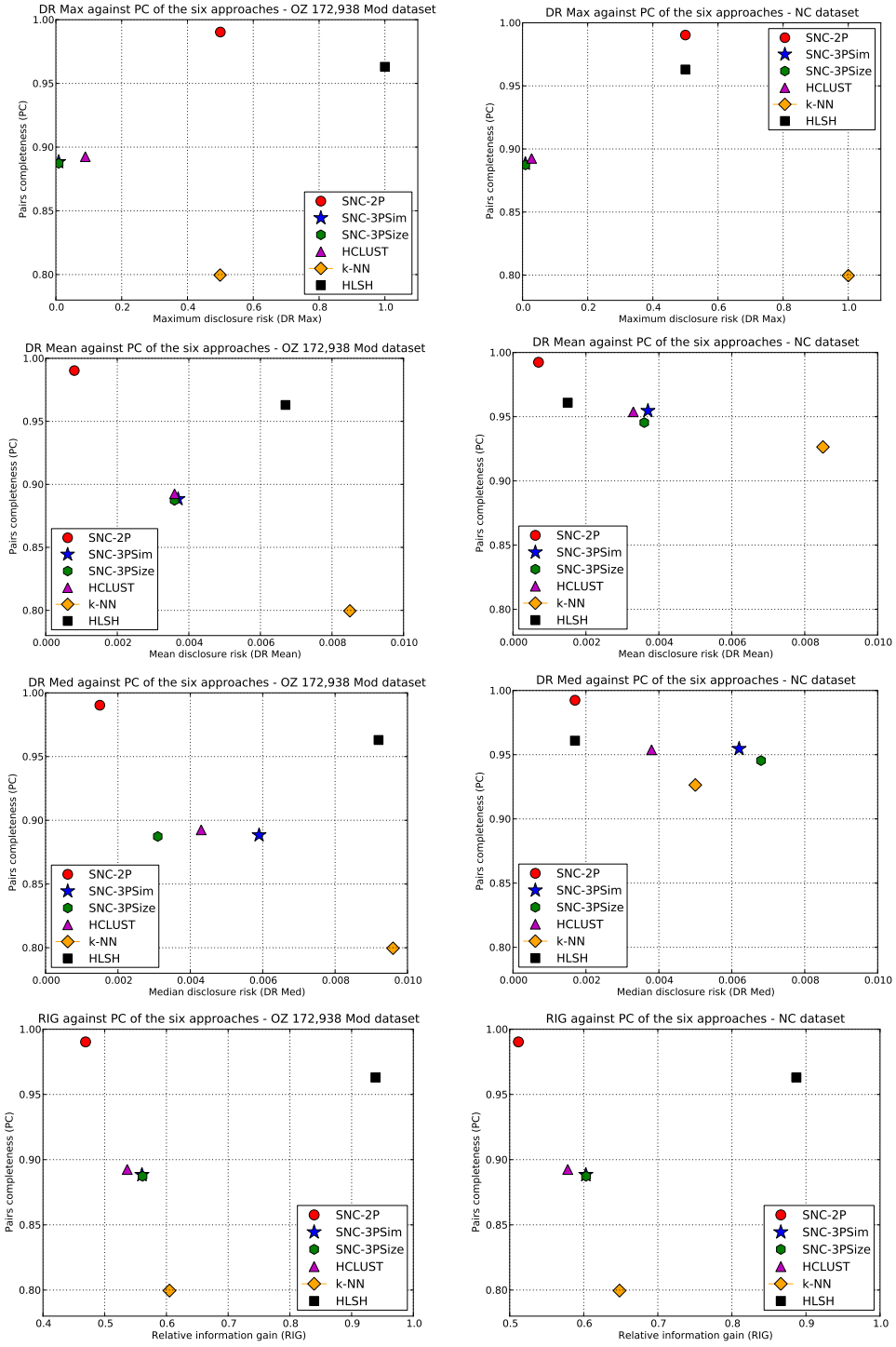


Figure 10.7: A comparison of disclosure risk measures (DR_{Max}, DR_{Mean}, DR_{Med}, RIG) against pairs completeness (PC) of the six private blocking approaches on the OZ-172,938 Mod (left column) and NC (right column) datasets. The best solutions are the ones closest to the upper left corner.

Table 10.2: Bloom filter parameterization for **CLK**, **RBF**, and **CLKRBF** methods.

	First name	Last name	City	Postcode
CLK hash functions (k)	30	30	30	30
CLK length (l)	1,000	1,000	1,000	1,000
RBF hash functions (k)	30	30	30	30
Agreement weight	2.5834	2.8908	1.2415	2.0852
Disagreement weight	−1.3757	−1.1752	−0.7708	−0.3543
Range (weight)	3.9591 (32%)	4.0660 (33%)	2.0123 (16%)	2.4395 (19%)
Average q-grams (g)	5.0762	5.3255	7.7592	3.9861
Dynamic BF length [59]	223	233	334	173
RBF length [59] (l)	668	689	334	397
Weight	32%	33%	16%	19%
CLKRBF hash functions (k)	29	30	15	17
CLKRBF length (l)	1,000	1,000	1,000	1,000

based solutions were set as the number of hash functions $k = 30$, the length of Bloom filters $l = 1,000$, $q = 2$, and the minimum similarity threshold $s_t = 0.8$. Weights, l for each attribute in the **RBF** method, and k for each attribute in the **CLKRBF** method on the NC dataset are given in Table 10.2. Soundex-based phonetic blocking [23] was employed in all these solutions to reduce the number of candidate record pairs.

We first compared the three Bloom filter encoding methods of **CLK**, **RBF**, and **CLKRBF** in our **2P-BF** solution. As Figure 10.8 illustrates, the **RBF** encoding requires more iterations to converge but achieves a higher recall of matches compared to the **CLK** method that completes the task in a smaller number of iterations. The hybrid **CLKRBF** method achieves a higher recall in a smaller number of iterations. The minimum similarity value of record pairs that remain unclassified shows that the **CLK** and **CLKRBF** encoding methods have a minimum similarity of 0.5 (i.e., non-matches with less than 0.5 similarity are removed) when half of the iterations are completed, while the **RBF** encoding requires three quarter of iterations to classify pairs so that the remaining pairs have a minimum of 0.5 similarity value.

We also investigated the distribution of bits (number of q-grams mapped to the bits) in Bloom filters, and frequencies of bits in the NC dataset by the three encoding methods (**CLK**, **RBF**, and **CLKRBF**) for the **2P-BF** solution in Figure 10.9. A minimum of 24 bigrams are mapped to every bit in the Bloom filters by the **CLKRBF** encoding method, while with the **CLK** and **RBF** methods a minimum of 10 and 15 bigrams are mapped, respectively (Figure 10.9(a)). Every bit with the **CLKRBF** method appears in at least 100,000 records in the NC dataset, which is significantly larger than the minimum number of records a bit appears with the **CLK** and **RBF** methods (Figure 10.9(b)). These results reveal that the **CLKRBF** encoding provides higher privacy than the other two encoding methods, because higher value for the number of bigrams mapped to a bit and higher frequency of a bit make a frequency linkage attack more difficult.

From the results shown in Figures 10.8 and 10.9, it can be seen that the **CLKRBF** encoding method outperforms the other two encoding methods in the **2P-BF** solution by achieving higher linkage quality, and better privacy in terms of bit distribution

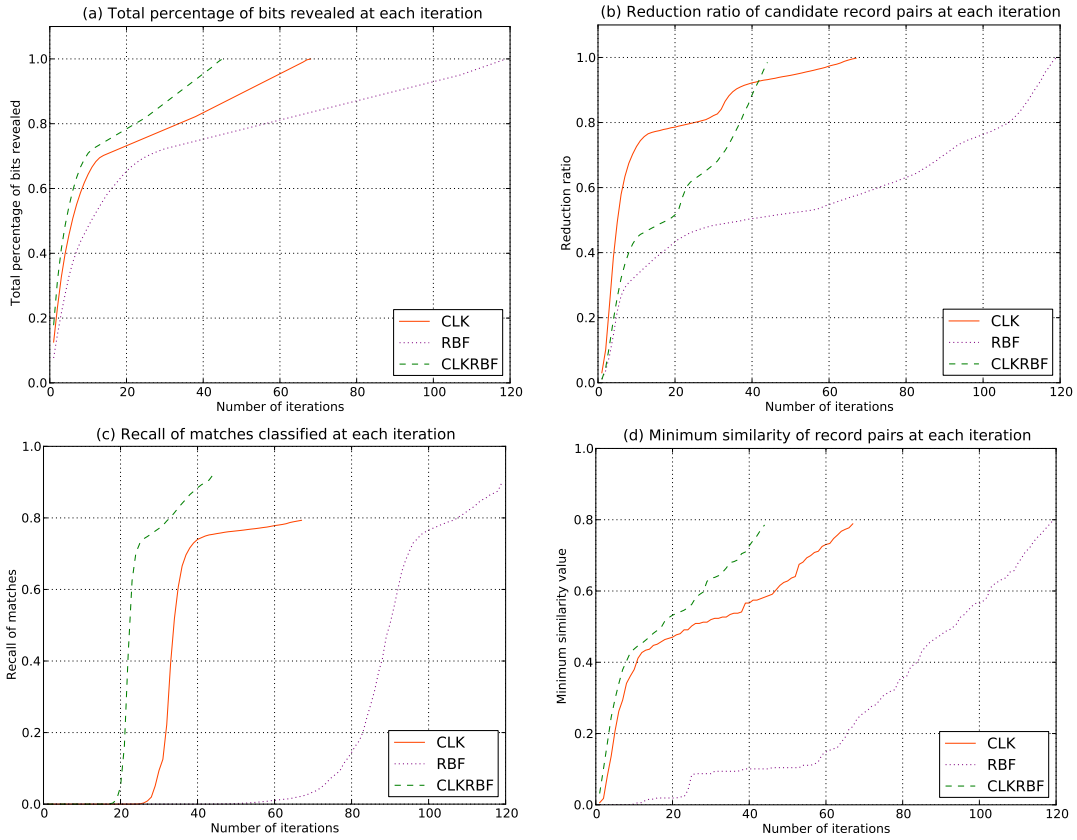


Figure 10.8: (a) The percentage of bits revealed, (b) reduction ratio (RR) of compared record pairs, (c) recall of matches, and (d) minimum similarity value of unclassified record pairs at each iteration for **CLK**, **RBF**, and **CLKRBF** encodings in the **2P-BF** solution on the NC dataset.

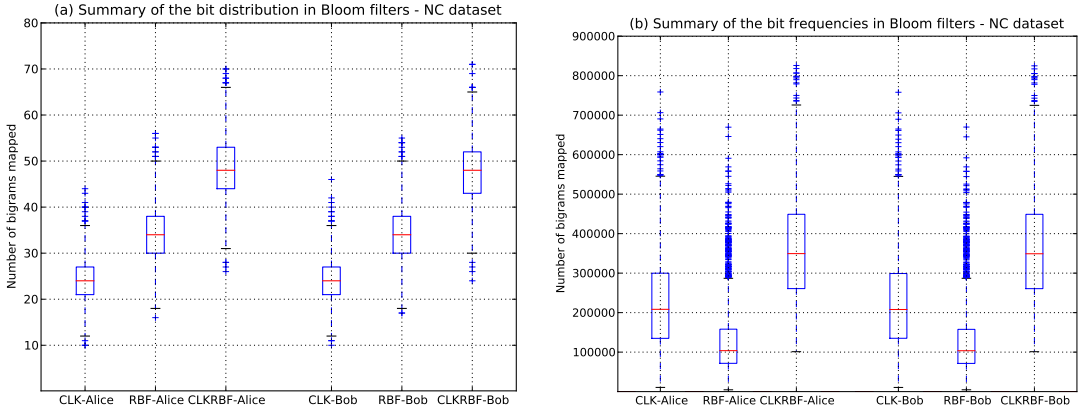


Figure 10.9: Summary of (a) number of bigrams mapped to bits in the Bloom filters and (b) frequencies of bits in the NC dataset for the **2P-BF** solution with different encoding methods.

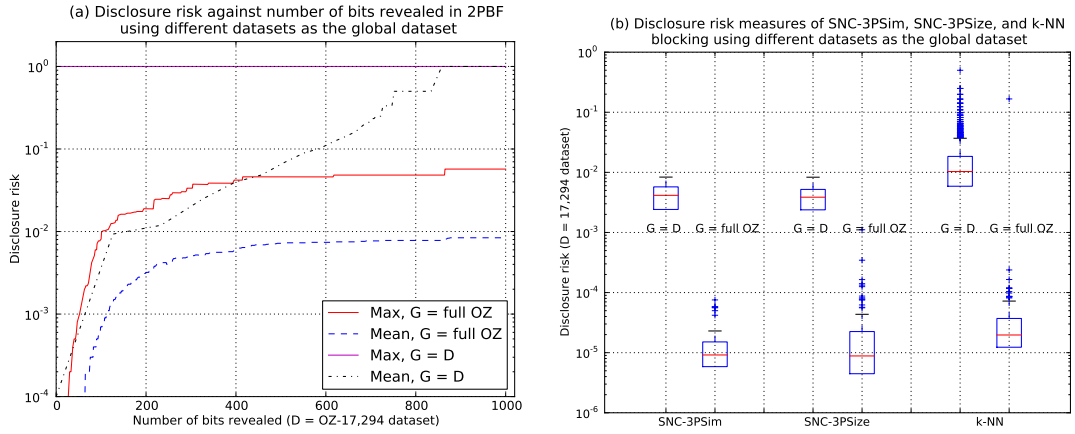


Figure 10.10: A comparison of (a) disclosure risk values of the **2P-BF** solution against number of bits revealed, and (b) disclosure risk values of private blocking solutions, on the **D** = OZ-17,294 dataset using **G** $\equiv$ **D** and **G** = full OZ database.

and pruning of non-matches.

The experiments described above assumed the worst case setting of global dataset. Since we used the original dataset as the global dataset (**G** $\equiv$ **D**) in this worst case, the number of global values n_g in $\mathbf{G}^M$ that match a certain masked value in $\mathbf{D}^M$ is very small, which results in high disclosure risk values. Ideally, a global dataset would not necessarily be equivalent to the original dataset and would have many combinations of different attribute values resulting in lower disclosure risk values, as was discussed in Section 5.5. Testing the privacy of the **2P-BF** technique and several private blocking techniques such as **SNC-3PSim**, **SNC-3PSize**, and **k-NN** on the OZ-17,294 dataset using a global dataset that is the full Australian telephone database (containing around 6.9 million records) provides much lower (around 2.5 magnitudes) disclosure risk results compared to the results in the worst case setting of **G** $\equiv$ **D**, as shown in Figure 10.10.

Figure 10.11 shows the scalability to different sizes of datasets (calculated by total linkage time) of the four private matching and classification techniques (**2P-Bin**, **2P-BF CLK**, **2P-BF RBF**, **2P-BF CLKRBF**) on the OZ datasets. Bin size was used as $k = 6$ for the binning-based approach (**2P-Bin**) and all four attributes in the OZ datasets were used as linkage attributes for all four techniques. The **2P-Bin** approach requires shorter linkage time and is efficient than the **2P-BF** based approaches (**2P-BF CLK**, **2P-BF RBF**, **2P-BF CLKRBF**). However, the disclosure risk is higher and the linkage quality is lower for the **2P-Bin** approach compared to the **2P-BF** based approaches, as will be compared in Figure 10.12. All three variations of the **2P-BF** based approaches require similar linkage time. The **CLKRBF** encoding method is faster than the **CLK** and **RBF** encoding methods as it requires a smaller number of iterations to converge compared to the other two encoding methods (which we discussed in Figure 10.8).

A comparison of disclosure risk measures (DR_{Max}, DR_{Mark}, DR_{Mean}, DR_{Med}) against linkage quality (calculated by F-measure) of the four private matching and classification techniques on the OZ-17,294 Mod dataset is given in Figure 10.12. The

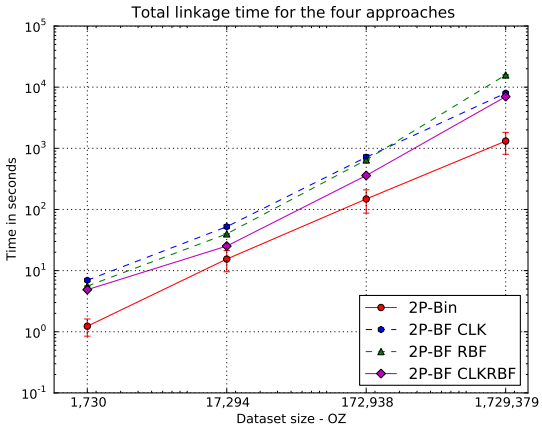


Figure 10.11: A comparison of scalability (measured by linkage time) of the four private matching and classification techniques on the OZ datasets.

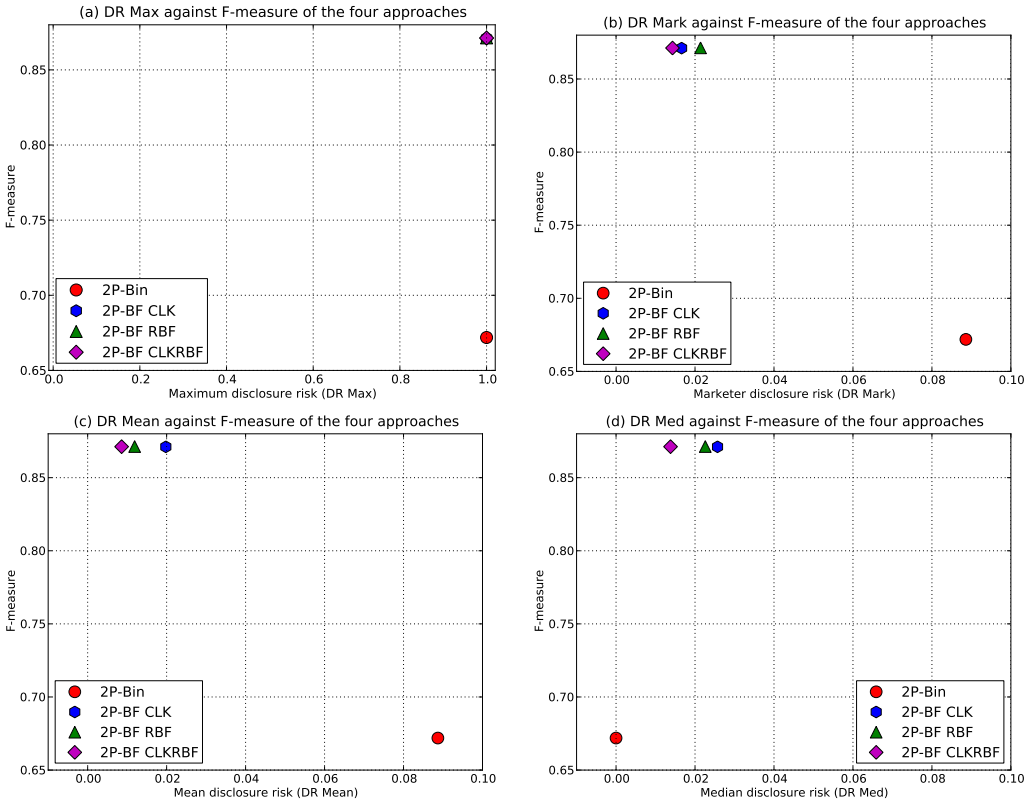


Figure 10.12: A comparison of disclosure risk measures ((a) DR_{Max}, (b) DR_{Mark}, (c) DR_{Mean}, and (d) DR_{Med}) against linkage quality (measured by F-measure) of the four private matching and classification approaches on the OZ-17,294 Mod dataset. The best solutions are the ones closest to the upper left corner.

2P-Bin solution leads to higher disclosure risk values (i.e. lower privacy) and lower linkage quality than the **2P-BF** based approaches. In **2P-BF** approaches, as can be seen from Figures 10.11 and 10.12, the **CLKRBF** encoding method performs better by achieving high F-measure, providing low disclosure risk, and requiring shorter linkage time than the other two encoding methods.

10.4 Discussion

The empirical evaluation of several private blocking and private matching and classification solutions using the proposed evaluation framework provides a comprehensive view of the performances of these solutions with regard to the three main properties of privacy, quality, and scalability.

The empirical results of private blocking solutions on the NC dataset and private matching and classification solutions on the OZ-17,294 Mod dataset are summarized in Tables 10.3 and 10.4, respectively, in terms of the three properties, scalability, quality, and privacy. We calculated overall scores (using Equation 5.9 on page 62 with $\alpha = 0.33$ and $\beta = 0.33$) based on different combinations of measures to compare the viability of PPRL solutions with respect to all three properties (as was discussed in Section 5.3.4). Different scores are calculated with different combinations of measures for the three properties, as presented in Tables 10.3 and 10.4.

For private blocking solutions we calculated the following four scores: score 1 is an average of RR, PC, and DR_{Max}, score 2 is an average of RR, PC, and DR_{Mean}, score 3 is an average of RR, PC, and RIG, and score 4 is an average of time, PC, and DR_{Mean}. The scores calculated for the private matching and classification solutions are: score 1 is an average of time, F-measure, and DR_{Max}, score 2 is an average of time, F-measure, and DR_{Mark}, and score 3 is an average of time, F-measure, and DR_{Mean}. We used equal weights for all measures in the calculation of scores. Scores with different weights would provide the ranking of solutions in the preferred context depending upon the application and / or user requirements. However, scoring is a cumbersome task that requires domain and application knowledge to determine the appropriate weight for each aspect.

The comparison results of the six private blocking solutions presented in Table 10.3 show that **SNC-2P** outperforms the other solutions in terms of all the measures except DR_{Max} (and thus except score 1). The **HLSH** is faster and achieves higher RR and PC compared to the other four approaches, however the DR and RIG measures are higher (i.e. lower privacy). **SNC-3PSim** and **SNC-3PSize** are faster as well with lower values for DR and RIG and achieve moderately higher RR and PC values. The **k-NN** and **HCLUST** require longer runtime though the other aspects provide moderate results.

Among the four private matching and classification solutions compared in Table 10.4, the **2P-BF** based approaches provide higher linkage quality results than the binning based approach (**2P-Bin**), while the DR measures are also lower (which means privacy is higher compared to the **2P-Bin** approach). However, the **2P-Bin**

Table 10.3: Comparison of the six private blocking approaches on the NC dataset. Best values in each row are shown in bold font. Four different scores are calculated as averages of the measures for the three properties of scalability, quality, and privacy. Measures marked with (+) have a positive impact on the overall score (i.e., high values are better) and measures with (-) have a negative impact (i.e., low values are better).

	SNC-2P	**SNC-3PSim**	**SNC-3PSize**	**HCLUST**	**k-NN**	**HLSH**
Time (-)	1044.02	**2.6439**	4.5502	95225.82	47075.76	1098.73
(normalized, see Section 5.3.3)	0.0109	**0.0000**	0.0001	1.0000	0.4943	0.0115
RR (+)	0.9901	0.9993	**0.9994**	0.9985	0.9992	0.9988
PC (+)	**0.9924**	0.9546	0.9454	0.9538	0.9264	0.9609
DR_{Max} (-)	0.4999	**0.0087**	**0.0087**	0.0278	1.0000	0.4999
DR_{Mean} (-)	**0.0007**	0.0037	0.0036	0.0033	0.0085	0.0015
RIG (-)	**0.5118**	0.6028	0.6031	0.5784	0.6483	0.8870
Score 1: RR, PC, DR_{Max}	0.8275	**0.9817**	0.9787	0.9748	0.6419	0.8199
Score 2: RR, PC, DR_{Mean}	**0.9939**	0.9834	0.9804	0.9830	0.9724	0.9861
Score 3: RR, PC, RIG	**0.8236**	0.7837	0.7806	0.7913	0.7591	0.6909
Score 4: Time, PC, DR_{Mean}	**0.9936**	0.9836	0.9806	0.6502	0.8079	0.9826

Table 10.4: Comparison of the four private matching and classification approaches on the OZ-17,294 Mod dataset. Best values in each row are shown in bold font. Three different scores are calculated as averages of the measures for the three properties of scalability, quality, and privacy. Measures marked with (+) have a positive impact on the overall score and measures with (-) have a negative impact.

	2P-Bin	**2P-BF CLK**	**2P-BF RBF**	**2P-BF CLKRBF**
Time (-)	11.2641	48.6865	39.8932	25.1866
(normalized, see Section 5.3.3)	**0.0000**	1.0000	0.7650	0.3720
Precision (+)	**1.0000**	0.9995	0.9997	0.9997
Recall (+)	0.5059	0.7719	**0.7721**	0.7720
F-measure / F (+)	0.6719	0.8711	**0.8713**	0.8712
DR Max (-)	1.0000	1.0000	1.0000	1.0000
DR Mark (-)	0.2886	0.0166	0.0214	**0.0143**
DR Mean (-)	0.2887	0.0198	0.0119	**0.0086**
Score 1: Time, F, DR_{Max}	**0.5573**	0.2904	0.3687	0.4997
Score 2: Time, F, DR_{Mark}	0.7944	0.6181	0.6950	**0.8283**
Score 3: Time, F, DR_{Mean}	0.7944	0.6171	0.6981	**0.8302**

solution is efficient and requires much shorter linkage time compared to others. The **2P-BF** with the **CLKRBF** encoding outperforms all others in terms of overall scores.

Figure 10.13 maps score 2 of the six private blocking solutions on the NC dataset, and score 3 of the four private matching and classification solutions on the OZ-17,294 Mod dataset into three-dimensional (3D) plots. Such a graphical representation of evaluation results allows us to analyze where a solution is placed in terms of the three properties of privacy, quality, and scalability and to compare different solutions. These 3D plots are better suited for interactive exploration or visualization than static visualization in a printed form.

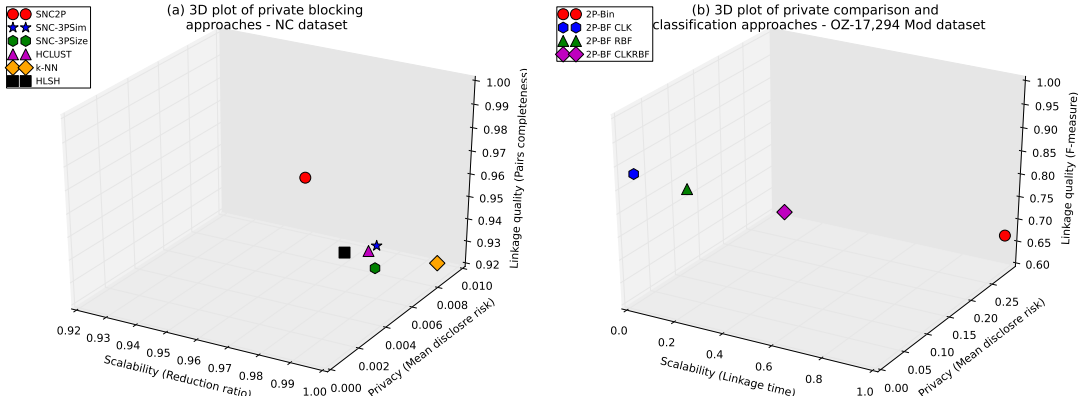


Figure 10.13: Three dimensional plots showing the comparison of (a) score 2: RR, PC, and DR_{Mean} of the six private blocking approaches, and (b) score 3: time, F-measure, and DR_{Mean} of the four private matching and classification approaches (right). The best solutions are the ones closest to the front upper right corner.

10.5 Summary

In this chapter, we have conducted a comprehensive evaluation and comparison of our proposed PPRL algorithms with several existing state-of-the-art techniques on large real-world databases using our evaluation framework proposed in Chapter 5. The experimental results on the datasets used show that our proposed algorithms perform better compared to several other existing solutions in terms of all three properties of PPRL: scalability, linkage quality, and privacy. Such large scale empirical study using our proposed evaluation framework allows extensive evaluation, analysis, and comparison of different PPRL solutions with respect to the three properties of PPRL.

Further work is required on large scale empirical evaluation [36] on other real datasets or realistic synthetic datasets generated using our GeCo tool [35, 183] in order to justify and generalize these empirical results. Investigating efficient and interactive linkage attacks for privacy evaluation, and approximation with error bounds in the linkage attacks would be another direction for future research.

Chapter 11

Conclusions and Future Work

In this thesis, we have addressed several shortcomings in the area of privacy-preserving record linkage (PPRL). These shortcomings have been identified by characterizing existing approaches using our taxonomy of PPRL techniques proposed in Chapter 4. In this chapter we summarize our contributions (in Section 11.2) and discuss directions for future research (in Section 11.3). Finally we conclude our work in Section 11.4.

11.1 Introduction

Based on our taxonomy proposed in Chapter 4 we characterized existing techniques that have been developed for PPRL in the last two decades (existing techniques are reviewed in Chapter 3). The taxonomy contains five main topics, namely privacy aspects, linkage techniques, theoretical analysis, evaluation, and practical aspects, each of which contains three dimensions (resulting in a total of fifteen dimensions). The characterization of existing techniques along these fifteen dimensions of our taxonomy (in Table 4.1 on page 42) allowed us to identify gaps in existing approaches and research directions. We formulated research questions along the five main topics of our taxonomy based on this study and addressed some of the identified gaps in our thesis. The addressed research questions among the identified gaps and the remaining questions for future work are presented in Figure 11.1. The research questions are numbered in the form of $x.y$, where x ($1 \leq x \leq 5$) denotes the main topic and y denotes the research question under each topic. We will next summarize the addressed gaps in Section 11.2 and discuss the gaps left for future research in Section 11.3.

11.2 Summary of our Contributions

The contributions of our work to address the shortcomings identified in the existing PPRL literature (as represented in Figure 11.1 by darker boxes with solid lines) are listed below:

1.1. Efficient two-party PPRL: We have addressed this question by developing two-party algorithms for PPRL based on efficient perturbation-based privacy techniques including k-anonymous mapping [72, 182], reference values [154], and

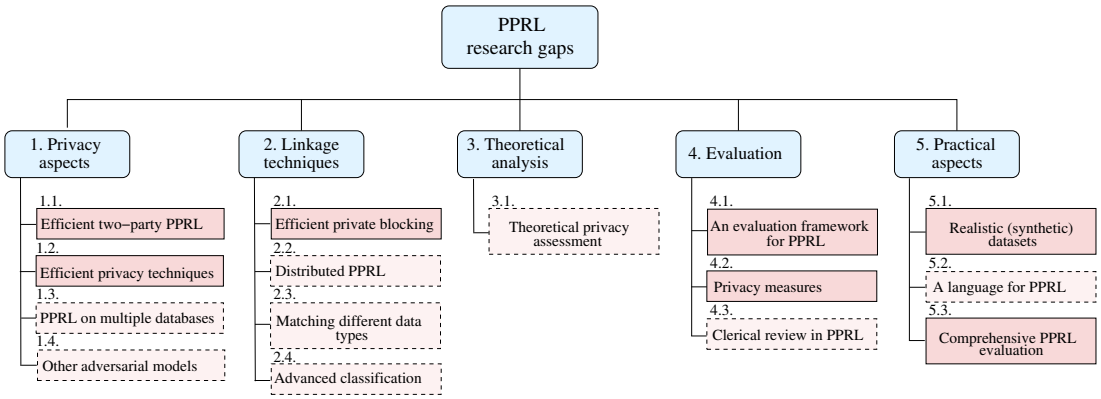


Figure 11.1: A summary of gaps identified in the existing PPRL research in Chapter 4 that have been addressed in this thesis, and future research questions. The research gaps represented by darker boxes with solid lines have been addressed in this thesis, while the lighter boxes with dotted lines are left for future work.

Bloom filters [17]. In Chapter 7, we proposed an efficient two-party private blocking technique based on *k*-anonymous sorted neighborhood clustering (adapted from the efficient three-party private blocking developed in Chapter 6); and in Chapters 8 and 9 we proposed efficient private matching and classification techniques based on reference values and Bloom filters, respectively.

1.2. Efficient privacy techniques: This question has been addressed in our solutions (proposed in Chapters 6 to 9) by replacing the expensive (in terms of computation and communication complexities) SMC-based privacy techniques with efficient perturbation-based privacy techniques such as *k*-anonymous mapping [72, 182], reference values [154], binning [132], random noise [108], and Bloom filters [17]. Empirical evaluation studies conducted on real-world datasets have shown that our approaches based on these perturbation techniques provide sufficient privacy guarantees while achieving high linkage quality.

2.1. Efficient private blocking: We have investigated the efficient blocking technique of sorted neighborhood approach [52, 84] in a privacy-preserving context for efficient private blocking (three-party and two-party) in Chapters 6 and 7. The comparative empirical evaluation conducted on real-world datasets in Chapter 10 validates the efficiency of our approach compared to several other existing private blocking solutions, including the *k*-nearest neighbor clustering-based approach [104], the Hamming-based locality sensitive hashing approach [56], and the hierarchical clustering-based approach [124].

4.1. An evaluation framework for PPRL: We have presented a comprehensive evaluation framework for PPRL (in Chapter 5) that contains numerical and normalized measures for the three key challenges of PPRL, which are scalability, linkage quality, and privacy. We used this framework to empirically evaluate

several private blocking and private matching and classification solutions for PPRL using real-world and synthetic datasets in Chapter 10. The empirical results have validated that our framework allows extensive evaluation, analysis, and comparison of different PPRL solutions on the same scale with respect to all three challenges of PPRL.

4.2. Privacy measures: We have proposed a novel set of disclosure risk measures (in Chapter 5) according to an evaluation model for privacy evaluation of PPRL solutions. The proposed disclosure risk measures provide a statistical summary of disclosure risk of revealing a masked dataset to an internal adversary, based on a linkage attack using an external global dataset with known values. The experiments conducted in Chapters 6 to 10 using our disclosure risk measures for privacy evaluation have shown that the proposed measures provide a practical way of quantifying the amount of privacy provided by a PPRL technique and allow for comparison with other techniques in terms of privacy provision.

5.1. Realistic (synthetic) datasets: We have developed an effective synthetic data generator and corruptor tool [35, 183] that models real-world data characteristics (such as data errors, variations, and attribute dependencies) in order to synthetically generate and / or corrupt datasets exhibiting similar characteristics as real data. We have used two real-world databases (OZ - Australian telephone database and NC - North Carolina voter registration database, as detailed in Section 5.4) for the empirical study of our solutions in Chapters 6 to 10. While the NC database contains real duplicate records with errors and variations, the OZ database does not contain any duplicate records. We therefore applied several corruption functions from our data corruptor to generate duplicate records for the OZ datasets with errors and variations, and to synthetically corrupt both the OZ and NC datasets. Such synthetically corrupted datasets allowed us to evaluate the quality of approximate matching of our proposed solutions in Chapter 6 to 10 in the presence of data errors and variations.

5.3. Comprehensive PPRL evaluation: We have conducted a comprehensive evaluation of several PPRL solutions in Chapter 10 using our evaluation framework proposed in Chapter 5. This study provided us with a baseline to comparatively evaluate and discuss different PPRL solutions with respect to the three properties (or challenges) of PPRL. Conducting such extensive experimental studies is one avenue of research that is highly beneficial to better understand the characteristics of different PPRL techniques and their applicability in real-world contexts.

11.3 Future Work

We now discuss several open research questions that are left to future work. These research questions (as represented in Figure 11.1 by lighter boxes with dotted lines) are:

1.3. **PPRL on multiple databases**: A main research gap we identified through the taxonomy is efficient PPRL of databases from multiple (more than two) data sources. PPRL on multiple databases introduces additional challenges with respect to complexity, linkage quality, and privacy [36]. Complexity increases significantly with multiple parties in terms of both computational efforts and communication size. Second, private matching and classification on multiple databases is challenging due to similarity calculations of multiple values. How to efficiently calculate the similarity of multiple values using approximate comparison functions in PPRL is an open question. Finally, the risk of privacy breaches increases with multiple parties due to possible collisions between a subset of parties with the aim to learn about another (subset of) party's private data. Despite these challenges, PPRL on multiple databases is useful and required in many real-world applications (as was described in Section 1.2).

1.4. **PPRL for other adversarial models**: Another limitation of most of the existing PPRL approaches is the assumption of the honest but curious (HBC) adversarial model which is not suffice in many real-world applications, because it is suitable only when the parties essentially trust each other. The malicious adversarial model, on the other hand, provides strong privacy guarantees by assuming dishonest parties, but it is computationally expensive and is therefore difficult to be adopted in practice. Limited work has been done in PPRL for the malicious adversarial model [69, 128, 145]. Future work is required in developing PPRL solutions under appropriate models such as the covert model [8] or accountable computing [95]. The covert model guarantees that the honest parties can detect the misbehavior of an adversary with high probability [8], while the accountable computing model provides accountability for privacy compromises by the adversaries without excessive complexity and cost that incur with the malicious model [95]. Transforming perturbation privacy-based HBC PPRL protocols into these models and proving privacy of solutions under these models are difficult which necessitate further research.

2.2. **Distributed PPRL**: Distributed PPRL would help improving the scalability property of PPRL by parallelising computations among different computational resources. Initial work on parallelism in PPRL based on locality sensitive hashing using the MapReduce framework [50] has been done by Karapiperis and Verykios [107]. Investigating how parallelism can improve the scalability of our proposed algorithms without compromising privacy is one challenging yet interesting avenue for future research.

2.3. **Matching different data types**: Developing approximate comparison functions for private matching of different data types such as numeric, date, time, geographic, and other complex types of data in PPRL is a beneficial next step of research in order to improve the quality of linkage. However, calculating similarities of such data without revealing their actual values is a challenge. In addition, techniques are required that deal with missing attribute values.

2.4. Advanced classification: Another important research question with regard to improving linkage quality is how the advanced classification techniques that have been developed for record linkage (such as machine learning-based [16, 25, 62], graph-based [85, 147], collective [15, 100], and group-based [153] techniques) can be efficiently employed in a privacy-preserving context. More efforts are required in this direction of research to improve the quality of linkage in PPRL applications [36].

3.1. Theoretical privacy assessment: Many PPRL solutions that have been proposed in the literature lack in theoretical assessment of privacy provided by those solutions. Therefore, more attention is required towards this direction. We have partially addressed this gap by analyzing the privacy of our proposed solutions in Chapter 6 to 9 in terms of what can be learned by the internal adversaries (parties involved in the protocols) from the data they communicate among them during the protocols. However, conducting an extensive analysis of privacy regarding different types of attacks by using a standard set of notations (similar to the big-*O* notation [155] used for complexity analysis) would be a constructive research direction in PPRL.

4.3. Clerical review in PPRL: Clerical review of unclassified record pairs (pairs that have been classified as possible matches by a private matching and classification technique) with manual efforts is difficult in PPRL, since the actual values of record pairs cannot be revealed to human expert(s) to enable the process of reviewing and decision making. How semi-supervised active learning techniques [6] for clerical review can be efficiently and effectively applied in PPRL applications is an open research question [36].

5.2. A language for PPRL: The final research gap that is left for future work is a language for PPRL. Researchers have used various languages, datasets, and measures to evaluate their PPRL algorithms. Crucially, there is currently no overarching framework available that allows researchers to implement and integrate their novel algorithms to PPRL based on a standard language which can then be evaluated comparatively. Developing a language for PPRL with abstract modules for each of the steps in the PPRL process would provide a key standard to model different PPRL approaches, which would be valuable for PPRL researchers.

11.4 Conclusions

In this thesis we have presented comprehensive research in scalable and approximate privacy-preserving record linkage (PPRL). First we conducted an extensive survey of existing PPRL techniques based on our taxonomy of PPRL. This taxonomy covers fifteen dimensions of PPRL along which existing techniques have been characterized. Based on this analysis, we have identified several shortcomings in existing PPRL for research directions and contributed to address some of the identified gaps.

The primary contributions of this work are scalable and approximate PPRL solutions in a two-party context (without the need of a third party to perform linkage and therefore no collusion between parties is possible) using efficient privacy techniques. We have proposed novel solutions for private blocking, and for private matching and classification in two-party PPRL, that address the three crucial properties of PPRL, which are scalability, linkage quality, and privacy. Specifically, we have proposed two efficient private blocking techniques (three-party, and two-party adapted from three-party) based on sorted neighborhood clustering [52, 84] to improve the scalability property; and two effective private matching and classification solutions based on reference values [154] and Bloom filters [17], respectively, that provide high linkage quality for approximate matching. Efficient data perturbation-based privacy techniques (in terms of computation and communication complexities compared to the SMC-based techniques) including k-anonymous mapping [72, 182], reference values [154], binning [132], Bloom filters [17], and random noise [108] have been used in our proposed solutions.

Another major contribution of our work relates to the evaluation of PPRL algorithms. Since a general framework for evaluation of PPRL has been missing in the literature, we have proposed an evaluation framework for PPRL with a standard set of measures for all three properties of PPRL. In this framework, we have also presented a novel set of disclosure risk measures for privacy evaluation (which has been another key gap in the existing literature) based on an evaluation model using an external global dataset. We have empirically evaluated all our proposed solutions on real and synthetically corrupted (with realistic data characteristics) datasets using our evaluation framework. We have also conducted a comprehensive and comparative evaluation of our proposed solutions and several other state-of-the-art solutions. The empirical results validate that our approaches perform equal or superior compared to several existing approaches by addressing all three properties of PPRL.

Bibliography

1. R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *ACM SIGMOD*, pages 86–97, San Diego, 2003. (cited on pages 24, 36, 104, and 109)

2. A. Aizawa and K. Oyama. A fast linkage detection scheme for multi-source information integration. In *Web Information Retrieval and Integration*, pages 30–39, Tokyo, 2005. (cited on page 18)

3. A. Al-Lawati, D. Lee, and P. McDaniel. Blocking-aware private record linkage. In *Workshop on Information Quality in Information Systems*, pages 59–68, 2005. (cited on pages 9, 28, and 98)

4. A. Amirbekyan and V. Estivill-Castro. A new efficient privacy-preserving scalar product protocol. In *Australasian conference on Data mining and analytics*, volume 70, pages 209–214. Australian Computer Society, Inc., 2007. (cited on page 36)

5. A. Amirbekyan and V. Estivill-Castro. Practical protocol for Yao's millionaires problem enables secure multi-party computation of metrics and efficient privacy-preserving k-nn for large data sets. *Knowledge and information systems*, 21(3):327–363, 2009. (cited on page 36)

6. A. Arasu, M. Götz, and R. Kaushik. On active learning of record matching packages. In *ACM SIGMOD*, pages 783–794, 2010. (cited on pages 20, 46, and 159)

7. M. Atallah, F. Kerschbaum, and W. Du. Secure and private sequence comparisons. In *ACM Workshop on Privacy in the Electronic Society*, pages 39–44, 2003. (cited on page 25)

8. Y. Aumann and Y. Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. *Journal of Cryptology*, 23(2):281–343, 2010. (cited on pages 44, 66, and 158)

9. T. Bachteler, J. Reiher, and R. Schnell. Similarity filtering with multibit trees for record linkage. Technical report, Working Paper WP-GRLC-2013-02, German Record Linkage Center, Nuremberg, 2013. (cited on page 18)

10. T. Bachteler, R. Schnell, and J. Reiher. An empirical comparison of approaches to approximate string matching in private record linkage. In *Statistics Canada Symposium*, 2010. (cited on page 98)

11. D. Barone, A. Maurino, F. Stella, and C. Batini. A privacy-preserving framework for accuracy and completeness quality assessment. *Emerging Paradigms in Informatics, Systems and Communication*, page 83, 2009. (cited on page 45)

12. C. Batini and M. Scannapieca. *Data quality: Concepts, methodologies and techniques*. Data-Centric Systems and Applications. Springer, 2006. (cited on page 16)

13. R. Baxter, P. Christen, and T. Churches. A comparison of fast blocking methods for record linkage. In *ACM SIGKDD Workshop on Data Cleaning, Record Linkage and Object Consolidation*, Washington DC, 2003. (cited on pages 2, 15, 16, 17, and 18)

14. J. J. Berman. Zero-check: a zero-knowledge protocol for reconciling patient identities across institutions. *Archives of pathology & laboratory medicine*, 128(3):344–346, 2004. (cited on page 118)

15. I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data, 2007*, 1(1), 2007. (cited on pages 2, 17, 20, and 159)

16. M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *ACM SIGKDD*, pages 39–48, Washington DC, 2003. (cited on pages 20 and 159)

17. B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970. (cited on pages 37, 43, 117, 142, 156, and 160)

18. L. Bonomi, L. Xiong, R. Chen, and B. Fung. Frequent grams based embedding for privacy preserving record linkage. In *ACM CIKM*, pages 1597–1601, 2012. (cited on pages 9, 30, and 37)

19. H. Bouzelat, C. Quantin, and L. Dusserre. Extraction and anonymity protocol of medical file. In *AMIA Annual Symposium*, pages 323–327, 1996. (cited on pages 23 and 35)

20. A. Broder, M. Mitzenmacher, and A. Mitzenmacher. Network applications of Bloom filters: A survey. In *Internet Mathematics*, 2002. (cited on page 37)

21. E. Brook, D. Rosman, and C. Holman. Public good through data linkage: measuring research outputs from the Western Australian Data Linkage System. *Aust NZ Journal of Public Health*, 32:19–23, 2008. (cited on page 4)

22. R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000. (cited on pages 35, 40, and 50)

23. P. Christen. A comparison of personal name matching: Techniques and practical issues. In *IEEE ICDM Workshop on Mining Complex Data*, Hong Kong, 2006. (cited on pages 2, 16, 18, 19, 30, 36, 98, 100, 115, 138, and 148)

24. P. Christen. Privacy-preserving data linkage and geocoding: Current approaches and research directions. In *IEEE ICDM Workshop on Privacy Aspects of Data Mining*, Hong Kong, 2006. (cited on pages 1, 5, 7, 23, 34, and 50)

25. P. Christen. Automatic record linkage using seeded nearest neighbour and support vector machine classification. In *ACM SIGKDD*, pages 151–159, 2008. (cited on pages 19, 20, and 159)

26. P. Christen. Febrl: An open source data cleaning, deduplication and record linkage system with a graphical user interface. In *ACM SIGKDD*, pages 1065–1068, 2008. (cited on pages 15 and 16)

27. P. Christen. Development and user experiences of an open source data cleaning, deduplication and record linkage system. *SIGKDD Explorations*, 11(1):39–48, 2009. (cited on pages 16 and 30)

28. P. Christen. Geocode matching and privacy preservation. In *KDD workshop on Privacy, Security, and Trust*, pages 7–24. Springer, 2009. (cited on pages 7, 34, and 50)

29. P. Christen. *Data Matching*. Data-Centric Systems and Applications. Springer, 2012. (cited on pages 1, 2, 4, 19, 20, 21, 26, 44, 61, 62, 64, 65, 81, 85, 94, 118, 120, and 122)

30. P. Christen. A survey of indexing techniques for scalable record linkage and deduplication. *IEEE Transactions on Knowledge and Data Engineering*, 24(9):1537–1555, 2012. (cited on pages 7, 8, 16, 17, 18, 20, 61, 67, 74, 88, 127, and 141)

31. P. Christen. Preparation of a real voter data set for record linkage and duplicate detection research. Technical report, Australian National University, 2013. (cited on pages 51, 56, 57, 59, 60, 63, 68, and 84)

32. P. Christen, T. Churches, and M. Hegland. Febrl – A parallel open source data linkage system. In *PAKDD, Springer LNAI*, volume 3056, pages 638–647, Sydney, 2004. (cited on pages 15 and 66)

33. P. Christen and K. Goiser. Quality and complexity measures for data linkage and deduplication. In F. Guillet and H. Hamilton, editors, *Quality Measures in Data Mining*, volume 43 of *Studies in Computational Intelligence*, pages 127–151. Springer, 2007. (cited on pages 2, 15, 16, 17, 20, 21, 45, and 61)

34. P. Christen and A. Pudjijono. Accurate synthetic generation of realistic personal information. In *PAKDD, Springer LNAI*, volume 5476, pages 507–514, Thailand, 2009. (cited on pages 41 and 63)

35. P. Christen and D. Vatsalan. Flexible and extensible generation and corruption of personal data. In *ACM CIKM*, pages 1165–1168, 2013. (cited on pages 46, 61, 64, 154, and 157)

36. P. Christen, D. Vatsalan, and V. S. Verykios. Challenges for privacy preservation in data integration. *Journal of Data and Information Quality (JDIQ)*, 2014. (cited on pages 154, 158, and 159)

37. T. Churches and P. Christen. Some methods for blindfolded record linkage. *BioMed Central Medical Informatics and Decision Making*, 4(9), 2004. (cited on pages 3, 21, 22, 26, and 95)

38. T. Churches, P. Christen, K. Lim, and J. X. Zhu. Preparation of name and address data for record linkage using hidden Markov models. *BioMed Central Medical Informatics and Decision Making*, 2(9), 2002. (cited on page 16)

39. D. E. Clark. Practical introduction to record linkage for injury research. *Injury Prevention*, 10:186–191, 2004. (cited on pages 4 and 5)

40. C. Clifton, M. Kantarcioglu, A. Doan, G. Schadow, J. Vaidya, A. Elmagarmid, and D. Suciu. Privacy-preserving data integration and sharing. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 19–26, 2004. (cited on pages 3, 5, and 23)

41. C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Zhu. Tools for privacy preserving distributed data mining. *SIGKDD Explorations*, 4(2):28–34, 2002. (cited on pages 2, 7, 36, 47, 95, and 117)

42. W. Cohen. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems*, 18(3):288–321, 2000. (cited on page 20)

43. W. W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IJCAI Workshop on Information Integration on the Web*, pages 73–78, 2003. (cited on pages 16, 19, and 26)

44. W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *ACM SIGKDD*, pages 475–480, 2002. (cited on pages 2, 18, 20, and 67)

45. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006. (cited on page 29)

46. G. Cormode, C. M. Procopiuc, E. Shen, D. Srivastava, and T. Yu. Empirical privacy and empirical utility of anonymized data. In *IEEE Data Engineering Workshops (ICDEW)*, pages 77–82, 2013. (cited on page 49)

47. T. Dalenius. Finding a needle in a haystack-or identifying anonymous census record. *Journal of official statistics*, 2(3):329–336, 1986. (cited on page 2)

48. F. K. Dankar and K. El Emam. A method for evaluating marketer re-identification risk. In *ACM EDBT/ICDT Workshops*, page 28, 2010. (cited on page 55)

49. T. de Vries, H. Ke, S. Chawla, and P. Christen. Robust record linkage blocking using suffix arrays and Bloom filters. *ACM Transactions on Knowledge Discovery from Data*, 5(2), 2011. (cited on page 18)

50. J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008. (cited on page 158)

51. S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990. (cited on page 29)

52. U. Draisbach, F. Naumann, S. Szott, and O. Wonneberg. Adaptive windows for duplicate detection. In *ICDE*, pages 1073–1083, 2012. (cited on pages 9, 67, 71, 79, 83, 156, and 160)

53. W. Du and M. Atallah. Protocols for secure remote database access with approximate matching. In *ACM Workshop on Security and Privacy in E-Commerce*, pages 87–111. Springer, 2000. (cited on pages 25 and 37)

54. G. T. Duncan and M. J. Elliot. *Statistical confidentiality: principles and practice*. Springer, 2011. (cited on pages 47, 49, 50, 51, 53, and 65)

55. G. T. Duncan, S. A. Keller-McNulty, and S. L. Stokes. Disclosure risk vs. data utility: The RU confidentiality map. Technical Report LA-UR-6428, Statistical Sciences Group, Los Alamos National Laboratory, 2001. (cited on page 53)

56. E. Durham. *A framework for accurate, efficient private record linkage*. PhD thesis, Faculty of the Graduate School of Vanderbilt University, Nashville, TN, 2012. (cited on pages 9, 10, 13, 30, 37, 44, 48, 53, 57, 61, 72, 73, 74, 84, 87, 98, 118, 130, 132, 133, 142, 146, and 156)

57. E. Durham, Y. Xue, M. Kantarcioglu, and B. Malin. Private medical record linkage with approximate matching. In *AMIA Annual Symposium*, page 182, 2010. (cited on pages 26 and 37)

58. E. Durham, Y. Xue, M. Kantarcioglu, and B. Malin. Quantifying the correctness, computational complexity, and security of privacy-preserving string comparators for record linkage. *Elsevier Information Fusion*, 13(4):245–259, 2011. (cited on pages 3, 23, 33, 95, and 117)

59. E. A. Durham, C. Toth, M. Kuzu, M. Kantarcioglu, Y. Xue, and B. Malin. Composite bloom filters for secure record linkage. *IEEE Transactions on Knowledge and Data Engineering*, 2013. (cited on pages 10, 13, and 148)

60. L. Dusserre, C. Quantin, and H. Bouzelat. A one way public key cryptosystem for the linkage of nominal files in epidemiological studies. *Medinfo*, 8:644–647, 1995. (cited on pages 23, 35, and 36)

61. C. Dwork. Differential privacy. *International Colloquium on Automata, Languages and Programming*, pages 1–12, 2006. (cited on page 37)

62. M. G. Elfeky, V. S. Verykios, and A. K. Elmagarmid. TAILOR: A record linkage toolbox. In *IEEE ICDE*, San Jose, 2002. (cited on pages 20, 61, 62, and 159)

63. A. Elmagarmid, P. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16, 2007. (cited on pages 1, 16, and 20)

64. V. Estivill-Castro and A. HajYasien. Fast private association rule mining by a protocol for securely sharing distributed data. In *IEEE Intelligence and Security Informatics*, pages 324–330, 2007. (cited on page 36)

65. I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Society*, 64(328):1183–1210, 1969. (cited on pages 1, 15, 16, 17, 20, 26, 31, 44, 67, and 133)

66. J. Ferro, L. Singh, and M. Sherr. Identifying individual vulnerability based on public data. In *IEEE Privacy, Security and Trust (PST)*, pages 119–126, 2013. (cited on page 56)

67. S. E. Fienberg. Confidentiality and disclosure limitation. *Encyclopedia of Social Measurement*, 1:463–69, 2005. (cited on pages 3, 50, 53, and 134)

68. O. for National Statistics. Matching anonymous data. In *Beyond 2011*, 2013. (cited on page 3)

69. M. Freedman, Y. Ishai, B. Pinkas, and O. Reingold. Keyword search and oblivious pseudorandom functions. *Theory of Cryptography*, 2005. (cited on pages 24, 36, and 158)

70. B. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys (CSUR)*, 42(4):14, 2010. (cited on page 47)

71. S. R. Ganta, S. P. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *ACM SIGKDD*, pages 265–273, 2008. (cited on pages 48, 49, and 53)

72. A. Gionis, A. Mazza, and T. Tassa. k-anonymization revisited. In *IEEE ICDE*, pages 744–753, 2008. (cited on pages 9, 37, 43, 56, 68, 79, 142, 155, 156, and 160)

73. O. Goldreich. *Foundations of cryptography: Basic applications*, volume 2. Cambridge University Press, 2004. (cited on pages 35, 36, 40, and 50)

74. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *ACM symposium on Theory of computing*, pages 218–229, 1987. (cited on page 36)

75. S. Grannis, J. Overhage, and C. McDonald. Analysis of identifier performance using a deterministic linkage algorithm. In *AMIA Symposium*, page 305, 2002. (cited on page 20)

76. L. Gu and R. Baxter. Decision models for record linkage. In *Selected Papers from AusDM, Springer LNCS 3755*, 2006. (cited on pages 16 and 19)

77. P. Hall and G. Dowling. Approximate string matching. *ACM Computing Surveys*, 12(4):381–402, 1980. (cited on page 19)

78. R. Hall and S. Fienberg. Privacy-preserving record linkage. In *Privacy in Statistical Databases, Springer LNCS 6344*, pages 269–283, Corfu, Greece, 2010. (cited on pages 2, 3, 21, 35, 47, 50, 73, 86, 95, 109, 117, and 128)

79. J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006. (cited on page 97)

80. J. Han, M. Kamber, and J. Pei. *Data mining: Concepts and techniques*. Morgan Kaufmann, 3rd edition, 2011. (cited on page 1)

81. B. Hawashin, F. Fotouhi, and T. Truta. A privacy preserving efficient protocol for semantic similarity join using long string attributes. In *ACM Workshop on Privacy and Anonymity in the Information Society*, 2011. (cited on page 29)

82. X. He and P. Niyogi. Locality preserving projections. *Neural Information Processing Systems (NIPS)*, 103, 2004. (cited on page 29)

83. M. A. Hernandez and S. J. Stolfo. The merge/purge problem for large databases. In *ACM SIGMOD*, San Jose, 1995. (cited on pages 18, 64, and 95)

84. M. A. Hernandez and S. J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Mining and Knowledge Discovery*, 2(1):9–37, 1998. (cited on pages 2, 9, 18, 20, 67, 94, 156, and 160)

85. M. Herschel, F. Naumann, S. Szott, and M. Taubert. Scalable iterative graph duplicate detection. *IEEE Transactions on Knowledge and Data Engineering*, 2011. (cited on pages 2, 20, and 159)

86. T. Herzog, F. Scheuren, and W. Winkler. *Data quality and record linkage techniques*. Springer Verlag, 2007. (cited on page 1)

87. G. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):530–549, 2003. (cited on page 28)

88. J. Hoag and C. Thompson. A parallel general-purpose synthetic data generator. *ACM SIGMOD*, 36:19–24, 2007. (cited on page 41)

89. A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, R. Lenz, J. Naylor, E. Schulte-Nordholt, E. Seri, and P. de Wolf. Handbook on statistical disclosure control (version 1.2). *A Network of Excellence in the European Statistical System in the field of Statistical Disclosure Control*, 2010. (cited on page 53)

90. A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, E. S. Nordholt, K. Spicer, and P.-P. De Wolf. *Statistical disclosure control*. John Wiley & Sons, 2012. (cited on page 47)

91. A. Inan, M. Kantarcioglu, E. Bertino, and M. Scannapieco. A hybrid approach to private record linkage. In *IEEE ICDE*, pages 496–505, Cancun, Mexico, 2008. (cited on pages 28, 29, 37, and 87)

92. A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino. Private record matching using differential privacy. In *EDBT*, 2010. (cited on pages 29 and 37)

93. M. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *Journal of the American Statistical Association*, 84(406):414–420, 1989. (cited on page 19)

94. W. Jiang and C. Clifton. A secure distributed framework for achieving k-anonymity. *International Journal on Very Large Data Bases*, 15(4):316–333, 2006. (cited on page 29)

95. W. Jiang, C. Clifton, and M. Kantarcıoğlu. Transforming semi-honest protocols to ensure accountability. *Data and Knowledge Engineering*, 65(1):57–74, 2008. (cited on pages 44, 66, and 158)

96. L. Jin, C. Li, and S. Mehrotra. Efficient record linkage in large data sets. In *Database Systems for Advanced Applications*, pages 137–146, 2003. (cited on pages 18, 44, and 67)

97. P. Jokinen, J. Tarhio, and E. Ukkonen. A comparison of approximate string matching algorithms. *Software-Practice and Experience*, 26(12):1439–1458, 1996. (cited on page 19)

98. J. Jonas and J. Harper. Effective counterterrorism and the limited role of predictive data mining. *Policy Analysis*, (584), 2006. (cited on page 4)

99. J. J. Jones, R. M. Bond, C. J. Fariss, J. E. Settle, A. D. Kramer, C. Marlow, and J. H. Fowler. Yahtzee: An anonymized group level matching procedure. *PloS one*, 8(2), 2013. (cited on page 27)

100. D. Kalashnikov and S. Mehrotra. Domain-independent data cleaning via analysis of entity-relationship graph. *ACM Transactions on Database Systems*, 31(2):716–767, 2006. (cited on pages 17, 20, and 159)

101. M. Kantarcioglu, W. Jiang, and B. Malin. A privacy-preserving framework for integrating person-specific databases. In *Privacy in Statistical Databases*, pages 298–314. Springer, 2008. (cited on pages 25, 37, and 44)

102. A. Karakasidis and V. Verykios. Advances in privacy preserving record linkage. In *E-activity and Innovative Technology, Advances in Applied Intelligence Technologies Book Series*. IGI Global, 2010. (cited on pages 7, 23, 33, 95, and 117)

103. A. Karakasidis and V. Verykios. Secure blocking+secure matching = secure record linkage. *Journal of Computing Science and Engineering*, 5:223–235, 2011. (cited on pages 30, 37, 98, 102, 127, and 128)

104. A. Karakasidis and V. Verykios. Reference table based k-anonymous private blocking. In *ACM Symposium on Applied Computing*, Riva del Garda, Italy, 2012. (cited on pages 9, 10, 13, 31, 36, 37, 61, 68, 74, 87, 98, 142, and 156)

105. A. Karakasidis, V. Verykios, and P. Christen. Fake injection strategies for private phonetic matching. In *Springer Data Privacy Management and Autonomous Spontaneus Security*, pages 9–24, Leuven, Belgium, 2011. (cited on pages 30, 36, 37, 40, 53, 57, and 58)

106. A. Karakasidis and V. S. Verykios. A sorted neighborhood approach to multidimensional privacy preserving blocking. In *IEEE Data Mining Workshops (ICDMW)*, pages 937–944, 2012. (cited on pages 9 and 67)

107. D. Karapiperis and V. S. Verykios. A distributed framework for scaling up lsh-based computations in privacy preserving record linkage. In *ACM Balkan Conference in Informatics*, pages 102–109, 2013. (cited on pages 31, 44, and 158)

108. H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *IEEE ICDM*, pages 99–106, 2003. (cited on pages 7, 37, 47, 142, 156, and 160)

109. H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. Random-data perturbation techniques and privacy-preserving data mining. *Knowledge and Information Systems*, 7(4):387–414, 2005. (cited on pages 7 and 47)

110. C. W. Kelman, J. Bass, and D. Holman. Research use of linked health data – A best practice protocol. *Aust NZ Journal of Public Health*, 26:251–255, 2002. (cited on page 4)

111. H. Keskustalo, A. Pirkola, K. Visala, E. Leppänen, and K. Järvelin. Non-adjacent diagrams improve matching of cross-lingual spelling variants. In *String Processing and Information Retrieval*, pages 252–265. Springer, 2003. (cited on page 19)

112. D. Kifer. Attacks on privacy and deFinetti's theorem. In *ACM SIGMOD*, pages 127–138, Providence, Rhode Island, USA, 2009. (cited on page 48)

113. H. Kim and D. Lee. Harra: fast iterative hashed record linkage for large-scale data collections. In *EDBT*, pages 525–536, Lausanne, Switzerland, 2010. (cited on pages 18 and 67)

114. L. Kissner and D. Song. Private and threshold set-intersection. In *Technical Report*. Carnegie Mellon University, 2004. (cited on pages 36, 104, and 109)

115. C. Kothari. *Research methodology: methods and techniques*. New Age International, 2004. (cited on page 11)

116. H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-hashing for message authentication. In *Internet RFCs*, 1997. (cited on pages 35, 36, 48, and 102)

117. T. G. Kristensen, J. Nielsen, and C. N. Pedersen. A tree-based method for the rapid screening of chemical fingerprints. *Algorithms for Molecular Biology*, 5(1):9, 2010. (cited on pages 18 and 31)

118. K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992. (cited on page 19)

119. H.-C. Kum and S. Ahalt. Privacy-by-design: Understanding data access models for secondary data. 2013. (cited on page 27)

120. H.-C. Kum, A. Krishnamurthy, A. Machanavajjhala, and S. Ahalt. Population informatics: Tapping the social genome to advance society: A vision for putting "big data" to work for population informatics. 2013. (cited on page 4)

121. H.-C. Kum, A. Krishnamurthy, A. Machanavajjhala, M. K. Reiter, and S. Ahalt. Privacy preserving interactive record linkage (ppirl). *Journal of the American Medical Informatics Association*, 2013. (cited on pages 27, 37, 46, and 51)

122. M. Kuzu, M. Kantarcioglu, E. Durham, and B. Malin. A constraint satisfaction cryptanalysis of Bloom filters in private record linkage. In *Privacy Enhancing Technologies*, pages 226–245. Springer, 2011. (cited on pages 26, 30, 39, 48, 52, 53, 118, 129, and 133)

123. M. Kuzu, M. Kantarcioglu, E. A. Durham, C. Toth, and B. Malin. A practical approach to achieve private medical record linkage in light of public resources. *Journal of the American Medical Informatics Association*, 20(2):285–292, 2013. (cited on page 37)

124. M. Kuzu, M. Kantarcioglu, A. Inan, E. Bertino, E. Durham, and B. Malin. Efficient privacy-aware record integration. In *ACM EDBT*, 2013. (cited on pages 9, 10, 13, 31, 37, 61, 74, 98, 142, and 156)

125. P. Lai, S. Yiu, K. Chow, C. Chong, and L. Hui. An Efficient Bloom filter based Solution for Multiparty Private Matching. In *International Conference on Security and Management*, 2006. (cited on pages 24, 37, and 44)

126. L. Lakshmanan, R. T. Ng, and G. Ramesh. On disclosure risk analysis of anonymized itemsets in the presence of prior knowledge. *ACM Transactions on Knowledge Discovery from Data*, 2(3):13, 2008. (cited on pages 48 and 53)

127. T. Lancaster. *Effective and efficient plagiarism detection*. PhD thesis, South Bank University, 2003. (cited on page 5)

128. F. Li, Y. Chen, B. Luo, D. Lee, and P. Liu. Privacy preserving group linkage. In *Scientific and Statistical Database Management*, pages 432–450. Springer, 2011. (cited on pages 27 and 158)

129. J. Li, R.-W. Wong, A.-C. Fu, and J. Pei. Anonymization by local recoding in data with attribute hierarchical taxonomies. *IEEE Transactions on Knowledge and Data Engineering*, 20(9):1181–1194, 2008. (cited on page 37)

130. N. Li, T. Li, and S. Venkatasubramanian. T-closeness: Privacy beyond k-anonymity and l-diversity. In *IEEE ICDE*, pages 106–115, 2007. (cited on page 37)

131. D. Lin. An information-theoretic definition of similarity. In *International conference on machine learning*, volume 1, pages 296–304. San Francisco, 1998. (cited on page 29)

132. Z. Lin, M. Hewett, and R. B. Altman. Using binning to maintain confidentiality of medical data. In *AMIA Symposium*, page 454, 2002. (cited on pages 37, 156, and 160)

133. Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology - CRYPTO*, pages 36–54. Springer, 2000. (cited on page 7)

134. Y. Lindell and B. Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. *Advances in Cryptology - EUROCRYPT*, pages 52–78, 2007. (cited on page 35)

135. Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Confidentiality*, 1(1):5, 2009. (cited on pages 2, 7, 35, 36, 40, 47, 50, 52, 95, and 117)

136. H. Liu, H. Wang, and Y. Chen. Ensuring data storage security against frequency-based attacks in wireless networks. In *Distributed Computing in Sensor Systems*, pages 201–215. Springer, 2010. (cited on pages 36 and 48)

137. M. Luby and C. Rackoff. How to construct pseudo-random permutations from pseudo-random functions. In *Advances in Cryptology - CRYPTO*, volume 85, page 447, 1986. (cited on page 36)

138. A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1):3, 2007. (cited on page 37)

139. B. A. Malin, K. El Emam, and C. M. O'Keefe. Biomedical data privacy: problems, perspectives, and recent advances. *Journal of the American Medical Informatics Association*, 20(1):2–6, 2013. (cited on pages 4 and 5)

140. C. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 59. MIT Press, 1999. (cited on pages 21 and 61)

141. A. Marcus, E. Wu, D. Karger, S. Madden, and R. Miller. Human-powered sorts and joins. *VLDB Endowment*, 5(1):13–24, 2011. (cited on page 46)

142. A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *ACM SIGKDD*, pages 169–178, Boston, 2000. (cited on pages 18 and 20)

143. X. Meng and D. Rubin. Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika*, 80(2):267, 1993. (cited on page 20)

144. M. Mitzenmacher and E. Upfal. *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005. (cited on page 132)

145. N. Mohammed, B. Fung, and M. Debbabi. Anonymity meets game theory: secure data integration with malicious participants. *International Journal on Very Large Databases*, 20(4):567–588, 2011. (cited on pages 29, 37, 44, 45, and 158)

146. M. Murugesan, W. Jiang, C. Clifton, L. Si, and J. Vaidya. Efficient privacy-preserving similar document detection. *International Journal on Very Large Databases*, 19(4):457–475, 2010. (cited on page 5)

147. F. Naumann and M. Herschel. An introduction to duplicate detection. *Synthesis Lectures on Data Management*, 2(1):1–87, 2010. (cited on pages 20 and 159)

148. G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33:31–88, 2001. (cited on pages 19, 30, 105, and 116)

149. G. Navarro-Arribas and V. Torra. Information fusion in data privacy: A survey. *Information Fusion*, 2012. (cited on page 53)

150. D. Newman, S. Hettich, C. Blake, and C. Merz. UCI repository of machine learning databases. 1998. (cited on page 30)

151. J. Nin, V. Muntes-Mulero, N. Martinez-Bazan, and J.-L. Larriba-Pey. On the use of semantic blocking techniques for data cleansing and integration. In *IEEE Database Engineering and Applications Symposium*, pages 190–198, Banff, Canada, 2007. (cited on page 17)

152. C. O'Keefe, M. Yung, L. Gu, and R. Baxter. Privacy-preserving data linkage protocols. In *ACM Workshop on Privacy in the Electronic Society*, pages 94–102, 2004. (cited on pages 22, 24, 36, 44, and 95)

153. B. On, N. Koudas, D. Lee, and D. Srivastava. Group linkage. In *IEEE ICDE*, pages 496–505, 2007. (cited on pages 2, 20, and 159)

154. C. Pang, L. Gu, D. Hansen, and A. Maeder. Privacy-preserving fuzzy matching using a public reference table. *Intelligent Patient Management*, pages 71–89, 2009. (cited on pages 9, 10, 28, 36, 43, 68, 72, 73, 79, 84, 87, 96, 99, 110, 142, 155, 156, and 160)

155. C. Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003. (cited on pages 39, 45, 62, and 159)

156. C. Phua, K. Smith-Miles, V. Lee, and R. Gayler. Resilient identity crime detection. *IEEE Transactions on Knowledge and Data Engineering*, 24(3), 2012. (cited on page 4)

157. E. H. Porter and W. E. Winkler. Approximate string comparison and its effect on an advanced record linkage system. Technical Report RR97/02, US Bureau of the Census, 1997. (cited on pages 19 and 38)

158. C. Quantin, H. Bouzelat, F. Allaert, A. Benhamiche, J. Faivre, and L. Dusserre. How to ensure data security of an epidemiological follow-up: quality assessment of an anonymous record linkage procedure. *International Journal of Medical Informatics*, 49(1):117–122, 1998. (cited on pages 23, 35, and 44)

159. C. Quantin, H. Bouzelat, F.-A. Allaert, A.-M. Benhamiche, J. Faivre, and L. Dusserre. Automatic record hash coding and linkage for epidemiological follow-up data confidentiality. *Methods of Information in Medicine*, 37(3):271–277, 1998. (cited on pages 23 and 35)

160. C. Quantin, H. Bouzelat, and L. Dusserre. Irreversible encryption method by generation of polynomials. *Medical Informatics and the Internet in Medicine*, 21(2):113–121, 1996. (cited on page 36)

161. V. Raghavan, P. Bollmann, and G. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Information Systems*, 7(3):205–229, 1989. (cited on pages 8, 21, and 61)

162. E. Rahm and H. H. Do. Data cleaning: Problems and current approaches. *IEEE Data Engineering Bulletin*, 23(4):3–13, 2000. (cited on page 16)

163. A. Ramachandran, L. Singh, E. Porter, and F. Nagle. Exploring re-identification risks in public domains. In *IEEE Privacy, Security and Trust (PST)*, pages 35–42, 2012. (cited on page 56)

164. S. M. Randall, A. M. Ferrante, J. H. Boyd, and J. B. Semmens. Privacy-preserving record linkage on large real world datasets. *Journal of Biomedical Informatics*, 2013. (cited on page 31)

165. P. Ravikumar, W. Cohen, and S. Fienberg. A secure protocol for computing string distance metrics. In *ICDM Workshop on Privacy and Security Aspects of Data Mining*, pages 40–46, 2004. (cited on page 26)

166. M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998. (cited on page 54)

167. C. Rudin and K. L. Wagstaff. Machine learning for science and society. *Machine Learning*, pages 1–9, 2013. (cited on page 49)

168. M. Sadinle, R. Hall, and S. Fienberg. Approaches to multiple record linkage. In *International Statistical Institute*, Dublin, Ireland, 2011. (cited on page 44)

169. G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of*. Addison-Wesley, 1989. (cited on pages 19 and 26)

170. M. Scannapieco, I. Figotin, E. Bertino, and A. Elmagarmid. Privacy preserving schema and data matching. In *ACM SIGMOD*, pages 653–664, 2007. (cited on pages 28, 29, 30, 36, 37, 44, and 68)

171. B. Schneier. *Applied cryptography: Protocols, algorithms, and source code in C, 2nd Edition*. John Wiley & Sons, Inc., New York, 1996. (cited on pages 35, 36, 48, and 103)

172. R. Schnell. Efficient private record linkage of very large datasets. 2013. (cited on pages 9 and 31)

173. R. Schnell, T. Bachteler, and S. Bender. A toolbox for record linkage. *Austrian Journal of Statistics*, 33(1-2):125–133, 2004. (cited on page 36)

174. R. Schnell, T. Bachteler, and J. Reiher. Privacy-preserving record linkage using Bloom filters. *BMC Medical Informatics and Decision Making*, 9(1), 2009. (cited on pages 10, 26, 30, 31, 37, 48, 72, 73, 84, 87, 117, 118, 120, 129, 132, 134, and 146)

175. R. Schnell, T. Bachteler, and J. Reiher. A novel error-tolerant anonymous linking code. *German Record Linkage Center, Working Paper Series No. WP-GRLC-2011-02*, 2011. (cited on pages 10, 13, 30, 118, 130, 132, 133, and 146)

176. N. Sethi and G. T. Laurie. Delivering proportionate governance in the era of ehealth: Making linkage and privacy work together. *Medical Law International*, pages 168–204, 2013. (cited on page 5)

177. C. Shannon and W. Weaver. *The mathematical theory of communication*, volume 19. University of Illinois Press Urbana, 1962. (cited on pages 40, 57, and 58)

178. D. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. pages 44–55. Published by the IEEE Computer Society, 2000. (cited on pages 28 and 36)

179. X. Sun, H. Wang, J. Li, and J. Pei. Publishing anonymous survey rating data. *Data Mining and Knowledge Discovery*, 23(3):379–406, 2011. (cited on page 47)

180. X. Sun, H. Wang, J. Li, and Y. Zhang. Satisfying privacy requirements before data anonymization. *The Computer Journal*, 55(4):422–437, 2012. (cited on page 37)

181. L. Sweeney. *Computational disclosure control: A Primer on Data Privacy Protection*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2001. (cited on page 59)

182. L. Sweeney. K-anonymity: A model for protecting privacy. *International Journal of Uncertainty Fuzziness and Knowledge Based Systems*, 10(5):557–570, 2002. (cited on pages 37, 68, 71, 79, 142, 155, 156, and 160)

183. K.-N. Tran, D. Vatsalan, and P. Christen. Geco: an online personal data generator and corruptor. In *ACM CIKM*, pages 2473–2476, 2013. (cited on pages 10, 46, 64, 154, and 157)

184. S. Trepetin. Privacy-preserving string comparisons in record linkage systems: a review. *Information Security Journal: A Global Perspective*, 17(5):253–266, 2008. (cited on pages 4, 7, 23, 26, 33, 95, and 117)

185. E. Turgay, T. Pedersen, Y. Saygın, E. Savaş, and A. Levi. Disclosure risks of distance preserving data transformations. In *Scientific and Statistical Database Management*, pages 79–94. Springer, 2008. (cited on pages 49 and 53)

186. B. van Berkel and K. De Smedt. Triphone analysis: a combined method for the correction of orthographical and typographical errors. In *Conference on Applied natural language processing*, pages 77–83. Association for Computational Linguistics, 1988. (cited on page 19)

187. E. Van Eycken, K. Haustermans, F. Buntinx, A. Ceuppens, J. WEYLER, E. Wauters, O. VAN, et al. Evaluation of the encryption procedure and record linkage in the Belgian National Cancer Registry. *Archives of public health*, 58(6):281–294, 2000. (cited on page 24)

188. D. Vatsalan and P. Christen. An iterative two-party protocol for scalable privacy-preserving record linkage. In *AusDM, CRPIT 134*, Sydney, Australia, 2012. (cited on pages 10, 11, 48, 119, and 146)

189. D. Vatsalan and P. Christen. Sorted nearest neighborhood clustering for efficient private blocking. In *PAKDD*, volume 7819 of *Springer LNCS*, pages 341–352, Brisbane, Australia, 2013. (cited on pages 10, 68, 69, and 74)

190. D. Vatsalan, P. Christen, O. M. Christine, and V. S. Verykios. An evaluation framework for privacy-preserving record linkage. *Journal of Privacy and Confidentiality*, 6(1):35–75, 2014. (cited on pages 10, 50, 51, 55, 57, 58, 60, 74, 110, 133, and 142)

191. D. Vatsalan, P. Christen, and V. Verykios. An efficient two-party protocol for approximate matching in private record linkage. In *AusDM, CRPIT 121*, Ballarat, Australia, 2011. (cited on pages 10, 101, and 146)

192. D. Vatsalan, P. Christen, and V. S. Verykios. Efficient two-party private blocking based on sorted nearest neighborhood clustering. In *ACM CIKM*, pages 1949–1958, 2013. (cited on pages 10, 80, and 81)

193. D. Vatsalan, P. Christen, and V. S. Verykios. A taxonomy of privacy-preserving record linkage techniques. *Elsevier Journal of Information Systems*, 38(6):946–969, 2013. (cited on pages 4, 6, 10, 16, 21, 34, 42, 97, and 118)

194. V. Verykios, A. Karakasidis, and V. Mitrogiannis. Privacy preserving record linkage approaches. *International Journal of Data Mining, Modelling and Management*, 1(2):206–221, 2009. (cited on pages 3, 7, 21, 23, 33, 34, 50, 95, and 117)

195. G. Wang, H. Chen, and H. Atabakhsh. Automatically detecting deceptive criminal identities. *Communications of the ACM*, 47(3):70–76, 2004. (cited on page 4)

196. J. Wang, T. Kraska, M. J. Franklin, and J. Feng. Crowder: Crowdsourcing entity resolution. *Proceedings of the VLDB Endowment*, 5(11):1483–1494, 2012. (cited on page 46)

197. S. Weber, H. Lowe, A. Das, and T. Ferris. A simple heuristic for blindfolded record linkage. *Journal of the American Medical Informatics Association*, 2012. (cited on page 25)

198. Z. Wen and C. Dong. Efficient protocols for private record linkage. 2014. (cited on pages 32 and 37)

199. R. L. Wilson and P. A. Rosen. Protecting data through perturbation techniques: The impact on knowledge discovery in databases. *Journal of Database Management (JDM)*, 14(2):14–26, 2003. (cited on pages 7 and 47)

200. W. Winkler. Using the EM algorithm for weight computation in the Fellegi-Sunter model of record linkage. In *Section on Survey Research Methods, American Statistical Association*, volume 667, page 671, 1988. (cited on page 20)

201. W. Winkler. Near automatic weight computation in the Fellegi-Sunter model of record linkage. In *Annual Research Conference, US Bureau of the Census*, pages 145–155, 1989. (cited on page 20)

202. W. Winkler. String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In *Section on Survey Research Methods, American Statistical Association*, pages 778–783, 1990. (cited on page 19)

203. W. Winkler. Improved decision rules in the Fellegi-Sunter model of record linkage. In *Section on Survey Research Methods, American Statistical Association*, volume 274, page 279, 1993. (cited on pages 19 and 20)

204. W. E. Winkler. Methods for evaluating and creating data quality. *Elsevier Journal of Information Systems*, 29(7):531–550, 2004. (cited on page 17)

205. W. E. Winkler. Overview of record linkage and current research directions. Technical Report RR2006/02, US Bureau of the Census, 2006. (cited on pages 4 and 95)

206. I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes*. Morgan Kaufmann, second edition, 1999. (cited on page 103)

207. R. C.-W. Wong, A. W.-C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *International Journal on Very Large Databases*, pages 543–554, 2007. (cited on page 48)

208. R. C.-W. Wong, J. Li, A. W.-C. Fu, and K. Wang. (α, k)-anonymity: an enhanced k-anonymity model for privacy preserving data publishing. In *ACM SIGKDD*, pages 754–759, 2006. (cited on page 37)

209. L. Wu, X. Ying, and X. Wu. Reconstruction from randomized graph via low rank approximation. In *SIAM*, pages 60–71, Columbus, Ohio, USA, 2010. (cited on page 45)

210. M. Yakout, M. Atallah, and A. Elmagarmid. Efficient private record linkage. In *IEEE ICDE*, pages 1283–1286, Shanghai, 2009. (cited on pages 29, 36, 37, 44, and 68)

211. S. Yan, D. Lee, M. Y. Kan, and L. C. Giles. Adaptive sorted neighborhood methods for efficient record linkage. In *ACM/IEEE-CS joint conference on Digital Libraries*, 2007. (cited on page 83)

212. A. Yao. How to generate and exchange secrets. In *Foundations of Computer Science*, pages 162–167. IEEE, 1986. (cited on page 36)

213. E. Zheleva and L. Getoor. Preserving the privacy of sensitive relationships in graph data. In *1st ACM SIGKDD workshop on Privacy, security, and trust*, pages 153–171, San Jose, USA, 2007. Springer-Verlag. (cited on page 45)