Statistical Journal of the IAOS 30 (2014) 263–270
DOI 10.3233/SJI-140833
IOS Press

An efficient privacy-preserving record linkage technique for administrative data and censuses

Rainer Schnell
Methodology Research Group, University of Duisburg-Essen, D-47057 Duisburg, Germany
E-mail: Rainer.Schnell@uni-due.de

Abstract. Increasingly, administrative data is being used for statistical purposes, such as for registry-based census taking. Due to privacy concerns, this often requires linking separate files containing information on the same unit without revealing the identity of the unit. If the linkage has to be done without a unique identification number, it is necessary to compare keys derived from personal identifiers. When dealing with large files such as census data, comparing each possible pair of keys for two files is impossible. Therefore, special algorithms (blocking methods) must be used to reduce the number of comparisons needed. If the identifiers have to be encrypted due to privacy concerns, the number of available algorithms for record linkage and blocking is very limited. This paper describes the combination of a recently introduced encryption method for identifiers with a novel algorithm for blocking. Simulations show that the performance of these techniques allows their use for Big Data applications, censuses and population registries.

Keywords: Indexing, bloom-filter, PPRL, blocking, multibit trees, cryptographic keys

1. Introduction

Due to the increasing availability of administrative information and the rising costs of primary data collection, linking different databases to determine their overlap or to enhance the information available for a certain unit is a commonly used strategy for statistical purposes. For example, of the forty European censuses in 2010, only twenty-one were traditional censuses while the rest were based on the linkage of registries [22].

However, official statistics agencies are confronted with growing privacy concerns. For example, the European Commission conducted a survey on “Attitudes on Data Protection” in 2010.[1] Although the majority of respondents across Europe trusted the protection of personal information by national public authorities, 28% of all respondents did not [21]. Convincing privacy protection techniques may help national agencies conducting registry based studies with these hesitant populations.

Technical solutions for linking different databases are trivial if a unique personal identification number (PID) can be used. In some countries (for example, the Scandinavian countries), a PID is available for all members of the population. If privacy concerns prevail, the PID can be encrypted differently for each linkage operation. In practice, however, most statistical linkage operations are based on personal identifiers

[1]The survey was conducted in November/December 2010 as a CAPI survey in 27 EU states with 26.574 respondents aged 15 and over. The sample is being reported as random route with closest birthday selection; no response rate was given. The question asked was: “Different authorities (government departments, local authorities, agencies) and private companies collect and store personal information. To what extent do you trust the following institutions to protect your personal information? National public authorities (e.g. tax authorities, social security authorities)?” Answer categories: Totally trust, Tend to trust, Tend not to trust, Do not trust at all, Don’t know.

such as name or date of birth. Such identifiers must be combined to yield an identification code. However, the identifiers are usually neither stable nor recorded without errors [25]. If the identifiers have to be encrypted due to privacy concerns, linking is limited to the subset of cases with exact matching identifiers only. In many applications, this subset is not a random sample of the records. To address this issue, methods allowing for small variations in the identifiers should be used. This class of methods is called "privacy preserving record linkage techniques" (for a review, see [23]). A method for privacy preserving record linkage which has recently become popular is the use of Bloom-Filters.

2. Using Bloom-Filters for encoding identifiers

Bloom-Filters for cryptographic encoding of identifiers were first suggested in 2009 [18]. Since then, this approach has been used in different countries by different research teams and compares favorably to other approaches [15,23,24].

The basic principle is to split the string representing each identifier (for example, the name) into a set of unique subsets of length n (n-grams). For example, using $n = 2$, the bigram set of "PETER" is $\_P, PE, ET, TE, ER, R\_$. Each bigram of the set is mapped with k different functions into a binary vector of length l. In computer science, functions for mapping arbitrary long strings to vectors of a fixed length are called hash functions. For cryptographic applications, one way hash functions with an additional keyword (keyed HMACs) are commonly used. Examples of HMACs are MD-5 and SHA-1 (for details on HMACs, see [13]). Named after its inventor [2], algorithms which use hash functions for mapping to a binary vector are called *Bloom-Filters*. Figure 1 shows a simple example of mapping names to Bloom-Filters using bigrams. In the example, eight identical bit positions are set to one in both Bloom-Filters. In total, eleven bits in A and ten bits in B are set to one. Any similarity measure could be used, but most commonly the Dice coefficient [6] $Dice(a, b) = \frac{2|ab|}{|a|+|b|}$ is used. In this example, the Dice similarity of the two Bloom-Filters is $(2 * 8)/(10 + 11) \approx 0.762$.

In general, the similarity between two strings can be approximated by using a similarity measure for their Bloom-Filters. In practice, the use of longer Bloom-Filters (500 or 1.000 bits) and more hash functions (typically ten to twenty) has been found useful.

In the initial proposal, each identifier was mapped to a separate Bloom-Filter. For the use in record linkage, each identifier encoded in a Bloom-Filter could be used for computing the similarity of two records. However, if a random sample of identifiers in the population is available to the attacker, a cryptographic attack on Bloom-Filters might be successful for the most frequent names [12]. Therefore, the security of separate Bloom-Filter encodings must be enhanced.

3. Bloom-Filter based privacy preserving record linkage: Cryptographic Long Term Keys (CLK)

If a PID is not available, the number of possible identifiers is quite limited in most administrative databases. Some administrative databases contain unique identifiers. For example, birth registries usually have their own PID and in addition include hour of birth, minute of birth, sequence number in the case of twins, birth weight and Apgar-Score. But in general, these special identifiers are not available in other databases. Therefore, a key for linking must be based on those identifiers common to nearly all administrative databases. This set is, of course, specific to local regulations, but typically this basic set of identifiers (*BSID*) consists of the first name, surname (at birth), sex (at birth), date of birth, country of birth and place of birth. Additional identifiers are usually not given, and if they are, they tend to be even more volatile than those within the BSID (an obvious example is address data). A cryptographic key based on BSIDs first requires the standardization of the identifiers (e.g., conversion to uppercase, transforming special characters, removing titles and blanks etc.). As a next step, the set of unique n-grams of each identifier is formed. Numerical data such as date of birth is also treated as a string and split into n-grams. Usually, each element of the date of birth (day, month, year) is used separately as a single string and handled independently. Finally, the unique set for each identifier is mapped with a different number of hash-functions and a different password for each identifier to the *same* binary vector.[2] The resulting binary vector (typically 500–1.000 elements) is a cryptographic long-term key (*CLK*), which can be used for linking databases [19,24].

The main advantage of CLKs is the fact that they are much more difficult to attack than a set of separate

[2]The number of hash-functions can be chosen to reflect the importance of the identifier for the linkage. In practice, the entropy of an identifier is a good approximation for the relative importance of the identifier.

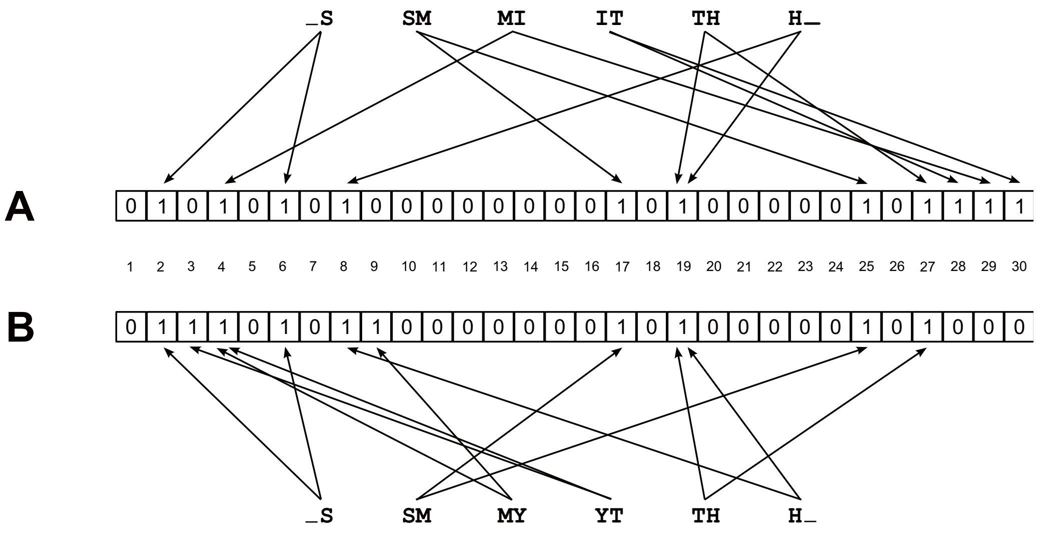

Fig. 1. Example for the mapping of two names (SMITH, SMYTH) using bigrams and two hash functions to two Bloom-Filters (A, B) with thirty bits each. Taken from [18].

Bloom-Filters. If the Bloom-Filters are not simply concatenated (as in [7]), CLKs have the additional advantage that a given bit set to one may be caused by different identifiers. This attribute increases the difficulty of attacks as described in [11] substantially.

The difficulty of an attack can be further increased by limiting the number of bigrams per identifier. This can be done with different methods, for example, by not using all bigrams of a name. This will prevent the identification of persons with longer names. For example, for unusually long names a sample of bigrams could be used. The probability for sampling could be decreased with the position of the bigram in the name so that bigrams at the beginning of the name are sampled with a higher probability. Additionally, random bits may be added; by using this carefully, the random bits have very little impact on similarity computations. No successful attack on CLKs has been reported up to now. Given the way CLKs are constructed, the kind of attacks used for Bloom-Filters will not work for CLKs.

4. Linking large databases with CLKs

Linking two databases consisting of CLKs implies the search for pairs of very similar binary vectors. This may be seen as a problem of finding nearest neighbors in a high dimensional binary space. If we have a database similar to the size of a census, we have to search the nearest neighbor among more than 100 million candidates. Therefore, a direct comparison of similarity among all pairs of CLKs is practically impossible. The number of comparisons has to be reduced to a range usually considered suitable for similarity computations, such as cluster analysis. Hence, groupings of cases to smaller subsets are needed. Algorithms for generating these kinds of groupings are called blocking methods.

4.1. Blocking methods for CLKs

There are a variety of blocking methods for reducing the number of record pairs which need to be compared for use in record linkage [4]. However, with regard to data structures similar to CLKs, the choice of possible methods is more limited. In this paper, only suitable candidates from the set of best performing methods in a recent comparison study are considered [5].

Two obvious methods are the use of external blocks and sorting. External blocks are formed by encrypting one element of the BSIDs with a different cryptographic hash function and using this code as a block. Examples for blocks are postcodes for residence, year or decade of birth or phonetic encodings of names such as *Soundex* [4]. External blocking is an application of the widely used *Standard Blocking* [9] but on CLKs with an external encrypted key. The main problem of external blocking is, that even slight variations in a block identifier of two records for the same person will usually result in a missed link for this pair.

Another obvious method is sorting. In [8] the *Sorted Neighbourhood Method* was introduced, which has become the standard method for handling large files with encrypted identifiers. Both input files are pooled and sorted according to a blocking key. A window of a fixed size is then slid over the records. Two records from different input files form a candidate pair if they are covered by the window at the same time.

Canopy Clustering, as suggested by [14], forms candidate pairs from those records placed in the same canopy. All records from both input files are pooled. The first canopy is created by choosing a record at random from this pool. This randomly chosen record constitutes the central point of the first canopy. All records within a certain loosely defined distance l from the central point are added to the canopy. Then, the central point and any records in the canopy within a certain more closely defined distance t from the former are removed from the record pool. Additional canopies are built in the same way as the first until there are no more remaining records. The result is a set of potentially overlapping canopies. Pairs which can be formed from the records of the same canopy constitute the set of candidate pairs.

Despite the satisfactory performance of these methods in specific settings, no blocking method shows optimal performance in all settings [5]. Therefore, the search for better blocking methods continues.

4.2. *A new blocking method*

In 2013, the use of Multibit Trees [10] for similarity filtering in general record linkage, without reference to privacy preserving record linkage, was suggested by the author [1]. The method described in the paper is called *q-gram Blocking*. The basic idea of *q-gram Blocking* is to transform all identifiers in a standard non-privacy preserving record linkage problem to a binary vector and then using a Multibit Tree to find nearest neighbors. By this transformation, any method for finding nearest neighbors in high dimensional binary space can be applied to the problem of finding nearest neighbors in unencrypted nominal data (or at least data treated as nominal). Therefore, this approach can be used for blocking or similarity filtering in *all* record linkage applications.

However, the suggested searching method can also be used for blocking in privacy preserving record linkage with CLKs. For the application of *q-gram Blocking* in two files of CLKs, *q-gram Blocking* is simply the search for every CLK in the smaller file, in the Multibit Tree built from the CLKs of the larger file. Only CLKs in the resulting set form candidate pairs.

Multibit Trees were introduced to search huge databases of structural information about chemical molecules [10]. If the query vectors are all binary, a query A is searched in a database of binary vectors B. All records in the database with a similarity to A above a certain threshold t should be retrieved. In chemoinformatics, very often the Tanimoto coefficient $\frac{A \cap B}{A \cup B}$ is used for measuring similarity of binary vectors.[3] The coefficient is simply the ratio of the number of 1's the vectors have in common to the number of 1's where either vector has a 1. The algorithm uses the fact stated by [20] that given the number of 1's in A and B (denoted by $|A|$ and $|B|$), the upper bound of the Tanimoto coefficient is $T_{max} = \frac{min(|A|,|B|)}{max(|A|,|B|)}$. If all vectors are stored in database indexed by $|B|$, searching for the vector A can be limited to those entries for which $S_{max} \geqslant S_{min}$. The increased speed of the algorithm is primarily due to those eliminations and the use of some special data structures. Details on the Multibit Tree algorithm and an implementation in Java can be found in [10].

For the application of Multibit Trees on files of CLKs, the tree structure is built for the first file and the records of the other file are queried sequentially. Therefore, the time required for querying the records of the second file is more important for the overall running time than the time needed to built the tree. In general, the average query time in Multibit Trees shows a linear increase with the number of records in the second file. This is an attractive scaling property of the method.

4.3. *Criteria for evaluating linkage operations*

The quality of the linkage operation is of vital importance for privacy preserving record linkage applications. Obviously, the quality of the linkage depends on the data quality of the identifiers. If names or date of birth are missing or grossly wrong, all record linkage techniques will fail. Given complete data with a reasonable amount of errors (for example, up to 20% records with one or two single letter errors in an identifier), acceptable linkage results are possible with unencrypted identifiers and privacy preserving record linkage should perform similarly. In addition, the time needed for linkage should not exceed the institutional constraints for linking sensitive data. Therefore, linkage of census operations should not exceed a few hours.

In general, the quality of linkages is evaluated using two criteria: recall and precision. Given the defini-

[3]Both Dice and Tanimoto coefficients have a range of 0–1, they are monotone related.

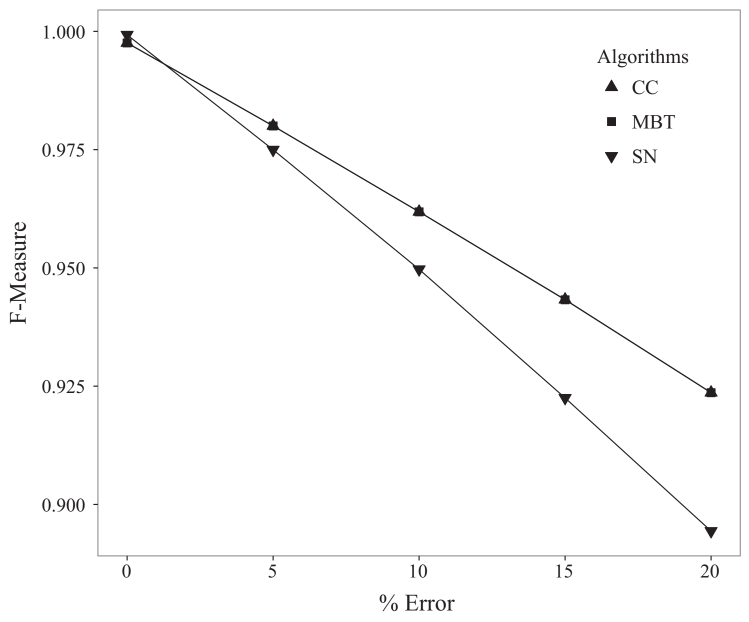

Fig. 2. F-measure for Canopy Clustering (CC), Sorted Neighborhood (SN) and Multibit Trees (MBT). Errors in 5%, 10%, 15% and 20% of the records. Results for CC and MBT are nearly identical.

tions of *false positives, fp* (pairs incorrectly considered a match) and *false negatives, fn* (pairs incorrectly considered a non-match), *recall* is defined as $tp/(tp + fn)$ and *precision* is defined as $tp/(tp + fp)$. The harmonic mean of recall and precision is the F-measure defined as $2 \cdot \frac{\text{recall}\cdot\text{precision}}{\text{recall}+\text{precision}}$.

The number of false positives in a linkage using Multibit Trees depends on the similarity threshold t. If t is close to 1.0, false positives could be reduced to nearly zero. In this case, only exact matching CLKs would be considered a match. Exact matching CLKs would in nearly all cases imply exact matching identifiers. False positives due to exact matching identifiers can not be avoided by any record linkage technique. However, restricting the links to exact matches only would induce high numbers of false negatives. Therefore, the similarity threshold must be decreased. Decreasing the similarity threshold will increase the number of false positives. Given this trade-off, the threshold must be chosen carefully, depending on the loss function for the application. The effect of variations of the threshold t will be illustrated with a simulation.

4.4. *Simulation of linking CLKs with Multibit trees*

The performance of Multibit Trees was studied in a series of simulations. In the initial publication of q-gram blocking [1], comparisons inter alia between Multibit Trees, canopy clustering, sorted neighborhood and standard blocking were reported. In most situations, Multibit Trees outperformed the other methods, even those that had performed best in other comparison studies.

For the simulation reported here, files were generated with Febrl [3]. Files with 1.000, 5.000, 10.000, 50.000, 100.000, 500.000 and 1 Million records were prepared. CLKs with 1.000 bits based on name, surname, sex and day/month/year of birth as identificators and $k = 20$ hash functions for each of the 6 fields were generated with an additional Python script. The second file was a copy of the first file, but with 0, 5, 10, 15 and 20% records containing errors.

4.4.1. *Results*

Figure 2 shows the F-measure for Canopy Clustering (CC), Sorted Neighborhood (SN) and Multibit Trees (MBT). The performance of all blocking methods decreases with increasing error rates, but more sharply for the Sorted Neighborhood technique. The results for Canopy Clustering and Multibit Trees are nearly identical. Overall, the new blocking method performs as well as the best known traditional technique.

To illustrate the behavior of Multibit Trees in more detail, some further results will be reported here.[4] As

[4]A more detailed report on the simulation is subject of a forthcoming paper with a more technical focus.

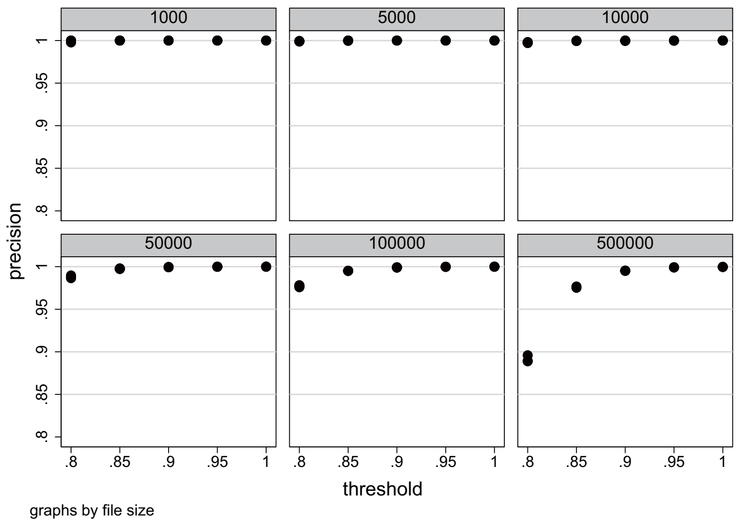

Fig. 3. Precision for linking CLKs with Multibit Trees depending on file size (1.000–500.000 records for each file) and threshold (0.8–1.00). Simulations with different error rates (0–20%) resulted in overlapping dots in the plots, therefore only one dot can be seen despite 5 different error levels.

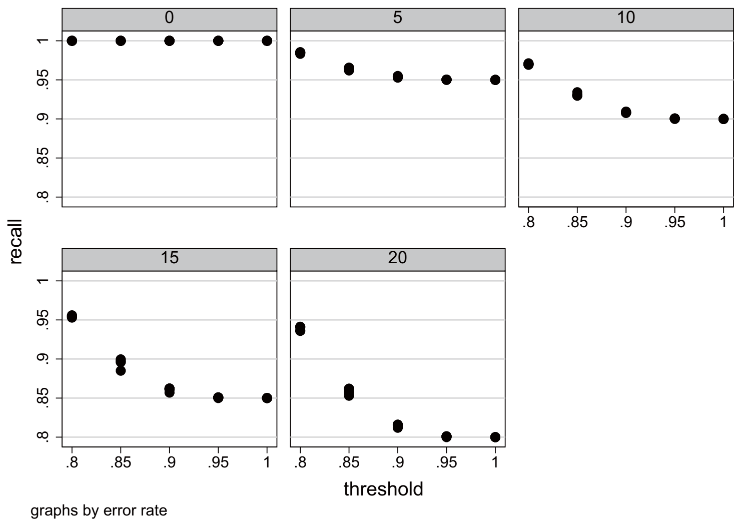

Fig. 4. Recall for linking CLKs with Multibit Trees depending on error rates (0, 5, 10, 15, 20%), file size (1.000–1 million records for each file) and threshold (0.8–1.00). Simulations with different file sizes resulted in overlapping dots in the plot.

Fig. 3 shows, the precision achieved by the trees is independent of error and independent of file size. However, with more than 100.000 records, lower thresholds than 0.9 will decrease the precision, but even with 1 million records a threshold of 0.85 will allow for a precision above 0.95. If blocks smaller than 100.000 records can be formed, nearly perfect precision with thresholds over 0.85 could be achieved.

In general, the observed recall for the trees decreases linear with error but is independent of file size. Figure 4 shows a nonlinear decrease of recall with increasing threshold. Therefore, to achieve sufficient recall, the similarity threshold must be lowered with increasing number of errors. A threshold of 0.85 will give a minimum of recall of 0.853, even with 20% errors. This threshold has shown a minimum precision of 0.95. For

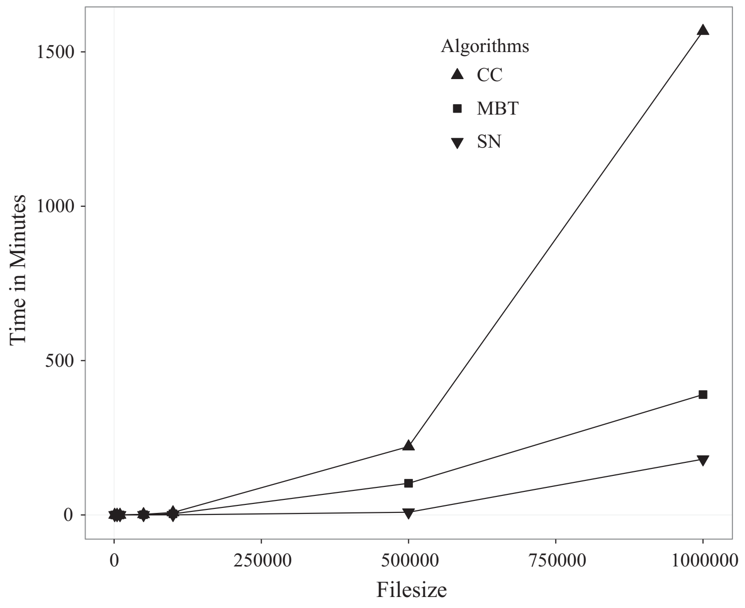

Fig. 5. Time in minutes for finding best matching pairs (100.000–1.000.000 records) for Canopy Clustering (CC), Sorted Neighborhood (SN) and Multibit Trees (MBT), 10% errors.

most applications, the 0.85-threshold therefore seem to be a reasonable start value. If the application has higher demands on precision and recall, using a threshold of 0.8 with blocks of at most 100.000 will show a minimum precision of 0.976 and a minimum recall of 0.936 despite 20% errors.

4.4.2. Computing time

Figure 5 shows the computing time in seconds on a Desktop PC with 64GB RAM and a hexa-core CPU with 3,4 Ghz under Ubuntu 12.04LTS.

The computing time for sorted neighborhood and Multibit Trees increases nearly linear with the number of records within the simulated range. This property makes them interesting options for linking CLKs.

For Multibit Trees the computing time will increase with decreasing similarity thresholds, since the number of pairwise comparisons will increase. However, even with a low similarity threshold of 0.85, two files of 500.000 records with 10% errors could be matched in 5902 seconds. For this combination of parameters, a recall of 0.931 and a precision of 0.977 was observed: Exactly the same performance as Canopy Clustering in about 40% of the time.

5. Conclusion

Privacy preserving record linkage for administrative datasets such as censuses require blocking methods to reduce the number of comparisons among encrypted identifiers. This paper illustrates the use of Multibit Trees for all identifiers encrypted in one common Bloom-Filter (CLKs).

Different simulations of this technique with large datasets showed similar or superior performance with lower run times compared to previously used methods. Within the simulated range, the suggested method shows a nearly linear increase in computing time with increasing file size. For most statistical applications, the speed and accuracy of Multibit trees will be sufficient with standard settings.[5] However, depending on the kind of identifiers and their quality, some fine tuning of the parameters will be necessary.[6] It must be kept in mind that linking with encrypted identifiers makes clerical editing impossible. Therefore, careful preprocessing and fine-tuning of parameters using already linked datasets has to be done in advance of the linkage operation. This problem arises for all privacy preserving record linkage techniques.

Additional techniques will be required for very large datasets, such as a population census with CLKs. The simplest option in this situation would be external

[5]For most applications, we used Bloom-Filters with 1.000 bits, ten to twenty hash-functions per identifier, padded bigrams and a similarity threshold of 0.85.

[6]The search for optimal settings given task parameters by response surface fitting on simulated data is subject of ongoing work.

blocking. For census operations, an obvious external block would be year of birth. If year of birth is encrypted with an HMAC such as MD-5 or SHA-1, the resulting code for year of birth would form a block and within each block CLKs with Multibit Trees could be used for linking. For European censuses, the expected size of these blocks will rarely exceed the limits of the simulations reported here. Hereby, privacy preserving record linkage with CLK and Multibit trees for a census could be done with a small cluster of servers within 24 hours.

Therefore, the techniques presented here provide a possible solution to conducting the censuses of the next decade in the current environment of increasing privacy concerns among European citizens.

Acknowledgements

This paper is an extended and revised version of two previously published conference papers [16,17]. The author is thankful for helpful comments by the editor and the reviewers. For computational assistance, I am indebted to Christian Borgs. C++ programming was done by Markus Koetter. This work has been supported by the Research Grant SCHN 586/17-1 of the German Research Foundation (DFG).

References

[1] T. Bachteler, J. Reiher and R. Schnell, Similarity filtering with multibit trees for record linkage. Working Paper WP-GRLC-2013-02, German Record Linkage Center, Nuremberg, 2013.

[2] B.H. Bloom, Space/time trade-offs in hash coding with allowable errors, *Communications of the ACM* **13**(7) (1970), 422–426.

[3] P. Christen, Febrl: An open source data cleaning, deduplication and record linkage system with a graphical user interface, in: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Y. Li, B. Liu and S. Sarawagi, eds, ACM, 2008, pp. 1065–1068.

[4] P. Christen, *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*, Data-Centric Systems and Applications, Springer, Berlin, 2012.

[5] P. Christen, A survey of indexing techniques for scalable record linkage and deduplication, *IEEE Transactions on Knowledge and Data Engineering* **24**(9) (2012), 1537–1555.

[6] L.R. Dice, Measures of the amount of ecologic association between species, *Ecology* **26**(3) (1945), 297–302.

[7] E.A. Durham, *A Framework for Accurate, Efficient Private Record Linkage*, PhD thesis, Vanderbilt University, Nashville, 2012.

[8] M.A. Hernández and S.S. Stolfo, Real-world data is dirty: data cleansing and the merge/purge problem, *Data Mining and Knowledge Discovery* **2**(1) (1998), 9–37.

[9] T.N. Herzog, F.J. Scheuren and W.E. Winkler, *Data Quality and Record Linkage Techniques*, Springer, New York, 2007.

[10] T.G. Kristensen, J. Nielsen and C.N.S. Pedersen, A tree-based method for the rapid screening of chemical fingerprints. *Algorithms for Molecular Biology* **5**(9) (2010).

[11] M. Kuzu, M. Kantarcioglu, E. Durham and B. Malin, A constraint satisfaction cryptanalysis of bloom filters in private record linkage, in: *The 11th Privacy Enhancing Technologies Symposium*, S. Fischer-Hübner and N. Hopper, eds, Berlin, 2011, Springer, pp. 226–245.

[12] 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) (2013), 285–292.

[13] K.M. Martin, *Everyday Cryptography. Fundamental Principles and Applications*. Oxford University Press, Oxford, 2012.

[14] A. McCallum, K. Nigam and L.H. Ungar, Efficient clustering of high-dimensional data sets with application to reference matching, in: *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: 20–23 August 2000; Boston*, New York, 2000. ACM, pp. 169–178.

[15] S.M. Randall, A.M. Ferrante, J.H. Boyd, J.K. Bauer and J.B. Semmens, Privacy-preserving record linkage on large real world datasets. *Journal of Biomedical Informatics* **50** (2014), 205-2012.

[16] R. Schnell, Efficient private record linkage of very large datasets. in: *Proceedings of the 59th World Statistics Congress (WSC)*, Hong Kong, 2013. International Statistical Institute, pp. 1862–1867.

[17] R. Schnell, Privacy-preserving record linkage and privacy-preserving blocking for large files with cryptographic keys using multibit trees, in: *JSM Proceedings 2013*, Alexandria, VA, 2013. American Statistical Association, pp. 187–194.

[18] R. Schnell, T. Bachteler and J. Reiher, Privacy-preserving record linkage using bloom filters, *BMC Medical Informatics and Decision Making* **9**(41) (2009), 1–11.

[19] R. Schnell, T. Bachteler and J. Reiher, A novel Error-Tolerant anonymous linking code. Working Paper WP-GRLC-2011-02, German Record Linkage Center, Nuremberg, 2011.

[20] S.J. Swamidass and P. Baldi, Bounds and algorithms for fast exact searches of chemical fingerprints in linear and sublinear time, *Journal of Chemical Information and Modelling* **47** (2007), 302–317.

[21] TNS Opinion and Social. *Special Eurobarometer 359: Attitudes on Data Protection and Electronic Identity in the European Union*, TNS Opinion & Social, Brussels, 2011.

[22] P. Valente, Census taking in europe: how are populations counted in 2010? *Population and Societies* **467** (2010), 1–4.

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

[24] V. Verykios and P. Christen, Privacy-preserving record linkage. *WIREs Data Mining and Knowledge Discovery* **3** (2013), 321–332.

[25] W.E. Winkler, Record linkage, in: *Handbook of Statistics Vol. 29A, Sample Surveys: Design, Methods and Applications*, D. Pfeffermann and C. Rao, eds, Elsevier, North-Holland, Amsterdam, 2009, pp. 351–380.