Efficient private record linkage of very large datasets

Rainer Schnell
German Record Linkage Center, University of Duisburg-Essen, GERMANY
e-mail:rainer.schnell@uni-due.de

Abstract

Increasingly, administrative data is being used for statistical purposes, for example when conducting registry based censuses. In practice, this usually 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, the comparison of keys derived from unit identifiers is necessary. When dealing with large files like census data or population registries, comparing each possible pair of keys in the two files is impossible. Therefore, special algorithms (blocking methods) have to be used to reduce the number of comparisons. The presentation will discuss the most commonly used blocking algorithms for encrypted data. Finally, a new blocking method suitable for large encrypted datasets will be demonstrated.

Keywords: administrative data, blocking methods, bloom filter, census data, indexing methods, linking codes

1 Introduction

Due to the increasing availability of administrative information, linking different databases to determine overlap of the databases or to enhance the information available for a certain unit is widely used in statistics. For example, many current European censuses are based on registries. Linking different databases using a set of common identifiers is trivial if a unique personal identification number (PID) can be used. In practice, however, most statistical linkage operations are based on personal identifiers like name or date of birth. Such identifiers must be combined to yield an identification code. The identifiers are usually neither stable nor recorded without errors. Hence, methods allowing for small variations of identifiers are to be used.

2 Using Bloom-Filters for encoding identifiers

In 2009, we suggested the use of Bloom-Filters for cryptographic encoding of identifiers (Schnell et al. 2009). Since then, this approach has been used in different countries by different research teams and compares favourable to other approaches (Vatsalan et al. 2013). The basic principle is splitting the string representing each

identifier (like the name) into a set of unique subsets of length n (n-grams). For example, with $n = 2$ the bigram set of "PETER" is $\_P, PE, ET, TE, ER, R\_$. Each bigram of the set is mapped with k different cryptographic one-way hash functions to a bit vector of length l (a Bloom-Filter). As hash functions, keyed hash functions (HMACs), usually MD-5 and SHA-1, are used (for details on HMACs, see Martin 2012). In the initial proposal, each identifier was mapped to a separate Bloom-Filter. For the use of record-linkage, each identifier encoded in a Bloom-Filter could be used for computing the similarity of two identifiers, since the Dice-coefficient of two Bloom-Filters approximates the Dice-coefficient of the unencrypted identifiers. 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 (see Kuzu et al. 2013). Therefore, the security of separate Bloom-Filter encodings had to 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. Therefore, a key for linking must be based on those identifiers common to all administrative databases. This set is, of course, specific to local regulations, but in general this basic set of identifiers (BSID) consists of the first name, surname (at birth), sex, date of birth, country of birth and place of birth. Additional identifiers are usually not given or even more volatile than those within the BSID. A cryptographic key based on BSIDs requires as first step the standardization of the identifiers (uppercase, transforming special characters, removing titles and blanks etc.). In the next step, the set of unique n-grams of each identifier is formed. Finally, this unique set of each identifier is mapped with a different number of hash-functions and a different password to the same bit-array. The resulting binary vector (typically 500-1000 elements) is a cryptographic long-term key (CLK), which can be used for linking databases (Schnell et al. 2011). No successful attack on CLKs has been reported up to now. However, given the way CLKs are constructed, the kind of attacks used for Bloom-Filters will not work for CLKs. Therefore, the main problem for the use of CLKs in practical applications is the size of databases used for registry-based research.

4 Linking large databases with CLKs

Finding two very similar CLKs may also be seen as the 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 the range usually considered suitable for similarity computations, such as cluster analysis. Hence, groupings of cases to smaller subsets are needed. Algorithms for generating these kind 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 that need to be compared for record linkage (Christen 2012). In regard to data structures similar to CLKs, the choice of possible methods is more limited. The presentation here will be limited to suitable candidates from the set of best performing methods in the comparison study of Christen (2012).

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. External blocking is the application of the widely used *Standard Blocking* (Herzog et al. 2007) to CLKs with an external encrypted key. But if the resulting block identifiers of two records of the same person differ by just one bit, external blocking will not be able to recover this pair.

The next obvious method would be sorting. Hernandez/Stolfo (1998) introduced the *Sorted Neighbourhood Method*. Both input files are pooled and sorted according to a blocking key. A window of a fixed size is then slided 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 McCallum et al. (2000) form 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. Further canopies are built in the very same way as the first one until there are no more remaining records. The result is a set of potentially overlapping canopies. Pairs that can be formed from the records of the same canopy constitute the set of candidate pairs.

However, no blocking method shows optimal performance in all settings (Christen 2012). Therefore, the search for blocking methods still continues.

4.2 A new blocking method

In January 2013, I suggested the use of Multibit Trees for similarity filtering in record linkage (Bachteler/Reiher/Schnell 2013). Since the method described there is based on the transformation of all identifiers in a standard record linkage problem to a bit array like a CLK in a first step, this method can be used for privacy preserving record linkage with CLKs in general. If two files of CLKs are available, the so called q-gram blocking described by us consists of the query of each CLK in the smaller file within the Multibit Tree built from the CLKs of the larger file. Only CLKs in the resulting set form candidate pairs.

Before the results of a simulation are reported, some details on Multibit Trees have to be explained. Kristensen et al. (2010) introduced the Multibit Tree to search huge databases of structural information about molecules. If the query vectors are all binary, we search a query $\vec{A}$ in a database of binary vectors $\vec{B}$. We want to find all records in the database with a similarity to $\vec{A}$ above a certain threshold t.

Multibit Trees work in three steps. In the first step, the vectors of the larger file are grouped into bins, which are formed by the size of the vector (denoted by $|\vec{B}|$), which here is the number of bits set to 1 in $\vec{B}$. Therefore, all bins satisfying

$$|\vec{B}| \leq t\,|\vec{A}| \quad \text{or} \quad t|\vec{B}| \geq |\vec{A}| \tag{1}$$

can be ignored in the searching step, because $\frac{min(|\vec{B}|,|\vec{A}|)}{max(|\vec{B}|,|\vec{A}|)}$ constitutes an upper bound of $S_J(\vec{A},\vec{B})$.

In the second step the vectors of equal size are stored in a binary tree structure for each bin. Two additional lists are stored at each node: List O contains all bit positions with constant value 0 and list I all bit positions with constant value 1 in all remaining vectors below that node.

Actually, searching a vector $\vec{A}$ is done in three phases. First, all bins satisfying equation 1 are eliminated. Second, at each node of the tree currently searched, the recorded lists O and I allow the computation of an upper bound of the Jaccard similarity for all fingerprints below the current node. If the bounds fall below the similarity threshold t, all nodes below this one can be eliminated from the search. Finally, the Jaccard similarity between the query vector $\vec{A}$ and all $\vec{B}_i$ at any node not previously eliminated is computed.

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.

4.3 A simulation study of linking with CLKs

We studied the performance of Multibit Trees in a series of simulations. In the initial publication of q-gram blocking (Bachteler/Reiher/Schnell 2013), we reported comparisons inter alia between Multibit Trees, canopy clustering, sorted neighborhood and standard blocking. In most situations, Multibit Trees outperformed the methods performing best in other comparison studies. In subsequent simulations, longer CLKs (1320 bit, number of hash functions $k = 15$ per identificator field) of BSIDs were simulated with 5.000 and 50.000 records for the small file (A) and 100.000 to 2.000.000 records for the larger file (B). File A was a random sample of file B, but 10% of records in A were simulated with errors in the BSIDs. Figure 1 shows the computing time in minutes for exact matching CLKs by the use of Multibit Trees. Even with 50.000 records in A and 2 million records in B, the match needed only slightly more than a minute on a standard server (16gb RAM, 2 * Quadcore CPU, 2.8ghz, Ubuntu 10.0.4). Of more interest is a similarity match with a similarity threshold of .95 (see figure 2). In this test, 5.000 records are still matched in less than 4 minutes, but for matching 50.000 records to 2 million records about 27 minutes are needed. So the computing time seems to increase linear with the number of records in both

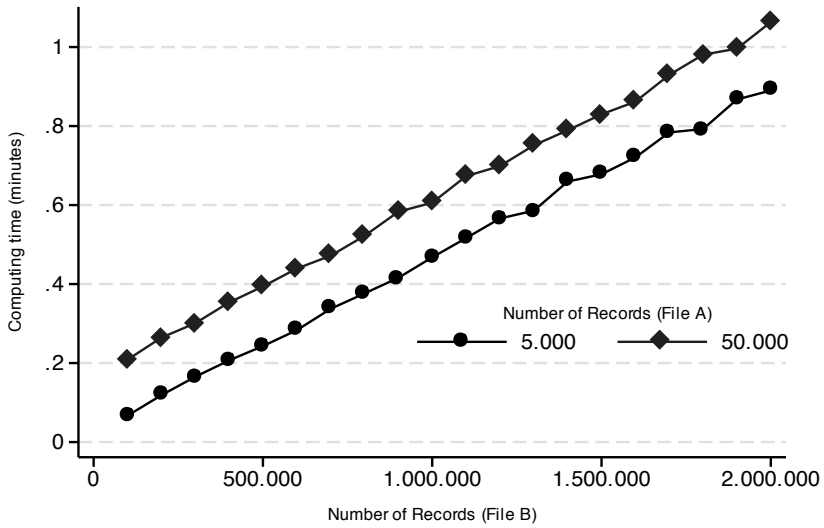

Figure 1: Computing time (in minutes) for finding 5.000 and 50.000 CLK records (File A) within a larger database (File B). Exact matches only.

files within the simulated range. The computing time will increase with decreasing similarity thresholds, since the number of pairwise comparisons will increase. However, the exceptional performance of Multibit Trees for this task even for large files is surprising: The running time for matching two files with 1 million CLKs each is less than 5 minutes for an exact match and for a similarity match (.95) less than 5:40 hours.

5 Conclusion

Privacy preserving record linkage for very large files requires blocking methods to reduce the number of comparisons. Here, the use of Multibit Trees has been suggested. Multibit Trees perform well for large datasets. Using Multibit Trees for full census size datasets of CLKs will require additional work, but with the exception of blocking on external keys, there currently seem to be few other algorithm usable for blocking large encrypted data files with comparable performance.

References

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

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

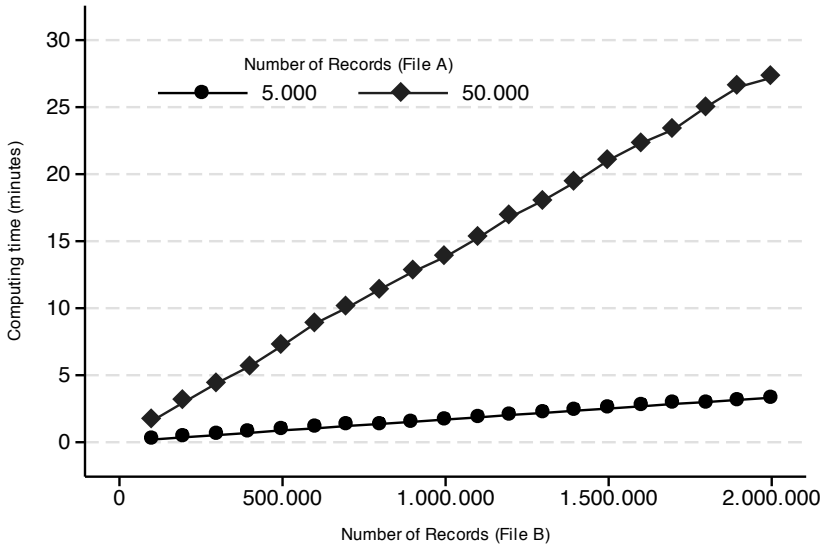

Figure 2: Computing time (in minutes) for finding 5.000 and 50.000 CLK records (File A) within a larger database (File B). Similarity threshold 0.95.

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

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

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

Kuzu, M., Kantarcioglu, M., Durham, E. A., Toth, C., and Malin, B. (2013). A practical approach to achieve private medical record linkage in light of public resources. *Journal of the American Medical Informatics Association*.

Martin, K. M. (2012). *Everyday Cryptography. Fundamental Principles and Applications*. Oxford University Press, Oxford.

McCallum, A., Nigam, K., and Ungar, L. H. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In Ramakrishnan, R., Stolfo, S., Bayardo, R., and Parsa, I., editors, *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: 20–23 August 2000; Boston*, pages 169–178, New York. ACM.

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

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

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