Generalization-based *k*-anonymization

Eva Armengol and Vicenç Torra

[1] IIIA - Artificial Intelligence Research Institute,
CSIC - Spanish Council for Scientific Research,
Campus UAB, 08193 Bellaterra, Catalonia (Spain)
email: eva@iiia.csic.es
[2] University of Skövde,
Sweden
Email: vtorra@his.se

Abstract. Microaggregation is an anonymization technique consisting on partitioning the data into clusters no smaller than k elements and then replacing the whole cluster by its prototypical representant. Most of microaggregation techniques work on numerical attributes. However, many data sets are described by heterogeneous types of data, i.e., numerical and categorical attributes. In this paper we propose a new microaggregation method for achieving a compliant k-anonymous masked file for categorical microdata based on generalization. The goal is to build a generalized description satisfied by at least k domain objects and to replace these domain objects by the description. The way to construct that generalization is similar that the one used in growing decision trees. Records that cannot be generalized satisfactorily are discarded, therefore some information is lost. In the experiments we performed we prove that the new approach gives good results.

Key words: k-anonymity, generalization

1 Introduction

Data privacy is a key issue when data bases are published to disseminate information: it is important to protect the individuals and entities and, at the same time, the data base has to be useful to extract representative information from it. For this reason, the research on protection methods becomes of capital importance.

Masking methods form a family of methods that given a data base modify it previous to its release so that published information is similar to the original one but not equal. In this way, disclosure is more difficult but data is still useful for analysis (i.e., information loss is low). To evaluate how good is a protection method, there are two commonly used measures: information loss [18] and disclosure risk [**?**]. As both measures are in contradiction, it is necessary to achieve a trade-off between the two aspects. Different methods have been developed which differ on how data is modified. They have different performance with respect to the level of disclosure risk and information loss they achieve. See e.g. [13, 12] for details on data privacy and masking methods.

There are three general categories of protection methods: perturbative, non-perturbative and synthetic data generators. Perturbative methods perform some distortion of the original data by adding some error. Microaggregation, rank swapping and noise addition are examples of perturbative methods. Non-perturbative methods do not rely on distortion of the original data but on partial suppressions or reductions of detail. These methods include different algorithms based on generalization and suppression. Finally, synthetic data generators methods generate synthetic data that preserve some desired characteristics of the original data. In this paper we focus on *masking* methods and, more particularly, we introduce a non-perturbative method based on generalization.

Different definitions exist for assessing disclosure risk or for establishing when a data set can be released without compromising sensitive information. Identiity and attribute disclosure are the two main types of disclosure. Record linkage has been used extensively for identity disclosure risk assessment [17, 18]. See [1] for a study of the worst-case scenario using record linkage. Differential privacy [15] and k-anonymity, [14] are two definitions to establish a suitable level of privacy, the former focusing on attribute disclosure and the latter on identity disclosure. k-Anonimity is satisfied when for each records there are other $k-1$ identical records, which avoids re-identification.

Files compliant with k-anonymity can be constructed by means of generalization, supression and microaggregation. See e.g. Mondrian and Incognito [9, 10] as methods to achieve k-anonymity based on generalization. On the contrary, [11] is about the use of microaggregation for achieving k-anonymity.

In order that these methods lead to a file compliant with k-anonymity, they have to be applied to all the variables available to the intruder. When applied to subsets of these variables, k-anonymity is not ensured. In this type of situations disclosure risk assessment needs to be evaluated. Some microaggregation algorithms and Mondrian [8] have been used in this way.

Concerning the data included in the data bases, objects are represented by attributes. Different types of attributes have been considered in the literature on masking methods. For example, to name a few, there are methods for numerical, categorical (ordinal and nominal), time series, dates, text. In this work we focus on the case of categorical attributes.

Mondrian and Incognito [9, 10], mentioned above, are examples of algorithms available for categorical data. [8] propose an algorithm to protect categorical attributes using clustering techniques. The original data are used to create clusters and then each cluster is protected independently. Li and Sankar [**?**] propose a protection method in two steps: first a linear programming formulation is applied in order to preserve the first-order marginal distribution, and then a simple Bayes-based swapping procedure assures the preservation of the joint distribution. Guo and Wu [**?**] investigate whether data mining or statistical analysis tasks can be conducted on randomized data when distortion parameters are not disclosed to data miners. There are also some approaches focusing on categorical attributes that use generalization. Thus, Wang et al [?] explore the data generalization concept as a way to hide detailed information. Once the data is masked, standard data mining techniques can be applied without modification. Some other authors, such as Sweeney [?], Samarati and Sweeney [14] use ontologies of concepts that allow the generalization of the values of an attribute.

In this paper we propose a new method for achieving a compliant k-anonymous masked file for categorical microdata. The approach is based on generalization, building generalizations that accommodate k records and thus achieving k-anonymity. At the same time records that cannot be generalized satisfactorily are discarded. Compared with other methods in the literature, our approach is able to deal with missing values, and we do not need to start with an ontology of generalizations. In addition, our method is evaluated satisfactorily with respect to the performance of classifiers built from the protected data set. Note that classifiers are standard tools in machine learning to build models of the data.

The structure of the paper is as follows. In Section 2 we introduce the notation we use. In Section 3 the algorithm we propose for k-anonymization is explained in detail. In Section 4 we present the experiments we carried out on the Adult data set from the UCI Machine Learning Repository. The paper finishes with some conclusions.

2 Preliminaries

In this work we follow the notation and approach of common literature on microdata protection. Concerning the attributes, they can be divided into three classes:

– *Identifiers* are the attributes that unambiguously identify a single individual or entity (for instance, the passport number), so they are usually removed or encrypted.
– *Quasi-identifiers* attributes that are those that identify an individual with some degree of ambiguity, but a combination of quasi-identifiers provides an unambiguous identification of some records, so they have to be masked.
– *Confidential* attributes that are those containing sensitive information that could be useful for statitic analysis, so they are usually not modified.

In [14] authors prove that removing the identifiers is no enough to protect the identity of an individual. Therefore, the protection must be done on the quasi-identifiers.

3 An Algorithm for k-anonymization

In this section we introduce an algorithm for k-anonymization. It is based on the generalization of a set of original records. Such generalization plays the role of a representative of a cluster in other microaggregation methods. The main goal is to search for a subset of records similar enough and set the attributes

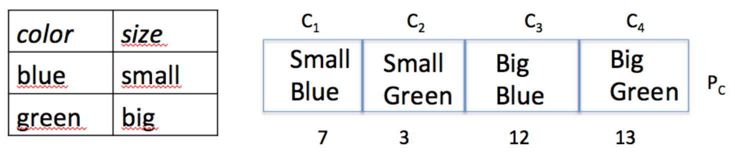
```
Function ANONYMIZE ($DB$, $K$, $C$, $long$)
    ;; $DB$ := all records
    ;; $C$ := set of confidential attributes
    ;; $long$ := minimum length for the new registers
    $Pt$ := set of patterns (initially $\emptyset$)
    $P_C$ := correct partition according to the confidential attributes
    analize-partition($DB$, $K$, $P_C$, c1, $long$)
end-function
```

Fig. 1. Initializations of the generalized-based k-anonymization. The main part of the process is to call the Analyze-partition function.

with different values to an *indifferent* value. The algorithm allows to choose the generalization degree, that is, how many attributes can have *indifferent* values. However, because we do not want to obtain generalizations very different of the original records, in the experiments we work with anonymized records having only one or two *indifferent* values.

Figure 1 shows the Anonymize algorithm proposed to anonymize a data base. The input parameters are K, C and *long*, where K is the minimum number of records that have to be put together; C is the set of attributes that the algorithm considers as confidential, i.e., they have not been modified during the process of anonymization; and, *long* is the minimum length of the anonymized records, i.e., it controls number of *indifferent* values in order to prevent from too generalized records.

To illustrate the algorithm, let us suppose a data base composed of 35 records described by four attributes texture, material, color and size and that both color and size are confidential attributes. Figure 2 shows the values that these attributes can take. The first step is to build P_C, namely the *correct partition*, having as many sets as the number of possible combinations of values hold by the confidential attributes. In the example, P_C is composed of 4 sets since there are two confidential attributes taking two different values each. Let us suppose that the class C_1 has 7 records, the class C_2 has 3 records, the class C_3 has 12



color	*size*
blue	small
green	big

Fig. 2. Correct partition P_C induced taking into account two attributes: *color* and *size*. The correct partition has as many sets as the combination of values of both attributes. Numbers indicate how many records are contained in each partition set.

```
Function Analize-partition ($DB$, $K$, $P_i$, $condition$, $long$)
    loop
        for each set $C_i$ of $P_i$ do
            if Cardinality($C_i$) $\geq k$ then
                ident := $analize\text{-}set$($K$, $C_i$, $condition$, $long$)
            end-if
            if ident = $\emptyset$ then end process
                else $DB$ := $DB$ - ident
                    $P_i'$ := update($P_i$)
                    Analize-partition ($DB$, $P_i'$, $condition$, $long$)
            end-if
        end-for
    end-loop
end-function
```

Fig. 3. The main goal of the Analize-partition function is to recursively call the function Analize-set for partition sets having more than K records.

records, and the class C_4 has 13 records. All the sets of P_C having less than K records are discarded. If we take $K = 5$, the records in the class C_2 are rejected. Notice that the rejection of these records can be seen as the rejection of outliers, since it means that there are few records with a given combination, so they could be easily re-identified. Then function Analyze-partition is called for each set of P_C. This function can be called several ways taking different *conditions*. This possibility will be explained in detail in Section 4.

Analyze-partition (Fig. 3) is a recursive function with three input parameters: a set of records DB (initially the whole data base), a partition P_i(initially P_C) and a *condition*. This function is a loop that analyzes each set C_i of P_i and, if the cardinality of that set is bigger than K, the function Analyze-set is called and returns the subset of records that are identified by some new created pattern. When no new records have been identified, the process ends; otherwise Analyze-partition is recursively called with an update of DB obtained by rejecting the newly identified records; and P_n' that is P_n without the identified records. The intuition behind this procedure is to induce partitions from the remaining records, i.e., those that have not still been used for generalization. Because at each step the number of records decreases, some sets of the partition can have a cardinality lower than K.

Analyze-set is also a recursive function (Fig. 4). In the first step, Analyze-set constructs a pattern AU using the anti-unification concept [2]. Such pattern is a record formed by all the attributes that have the same value in all the records in C_i. In this context, we call *description length* to the number of attributes describing AU. The remaining attributes are considered *indifferent.* If AU is either empty or has a length below *long* the input set has to be partitioned in subsets; otherwise there is a pattern long enough that satisfies more than K records.

```
Function Analize-set ($K$, $C_i$, $condition$, $long$)
    $AU$ := anti-unification($C_i$)
    if $A_c = \emptyset$ or description-length($AU$) $< long$ then
        $P_A$ := set of partitions on $C_i$ induced by each quasi-confidential
        $P_{cond}$ := subset of $P_A$ selected according to $condition$
        for each $P_{a_i} \in P_{cond}$ do
            for each set $C_j$ of $P_{a_i}$ do
                if Cardinality($C_j$) $\geq k$ then
                    $analize\text{-}set$($K$, $C_j$, $condition$, $long$)
                end-if
        end-for
    else $Pt := Pt \cup \{AU\}$
        return $C_i$
    end-if
end-function
```

Fig. 4. The recursive function Analyze-set constructs the patterns that generalize more than K records.

Notice that if AU is empty it means that the records in C_i have not attributes with common values. When the length of AU is lower than *long* it means that the pattern that could be extracted is too general, i.e., many attributes should have *indifferent* value. For instance, Fig. 5 shows the description of two records, namely *adult-8* and *adult-10*, described with 9 attributes (description-length = 9). Both objects have in common the value of only three of these attributes (i.e., the length of AU is 3), therefore this generalization is not useful since it has lost many information with respect to the complete description of a record.

When the elements in C_i have at least *long* attributes with the same value, the function constructs a pattern (i.e., the anti-unification of the elements in C_i) and finishes by returning the set C_i.

ADULT-8: (AGE B_25-29) (WORKCLASS PRIVATE) (EDUCATION BACHELORS)
(MARITAL-STATUS DIVORCED) (OCCUPATION PROF-SPECIALTY)
(RACE BLACK) (SEX FEMALE) (NATIVE-COUNTRY CUBA) (SALARY LESS_50)

ADULT-10: (AGE B_35-39) (WORKCLASS PRIVATE) (EDUCATION MASTERS)
(MARITAL-STATUS MARRIED-CIV-SPOUSE) (OCCUPATION EXEC-MANAGERIAL)
(RACE WHITE) (SEX FEMALE) (NATIVE-COUNTRY UNITED-STATES) (SALARY LESS_50)

AU: (WORKCLASS PRIVATE) (SEX FEMALE) (SALARY LESS_50))

Fig. 5. The records Adult-8 and Adult-8 are described by 9 attributes, however in only three of them they have the same value. The AU record shows these common attributes and their value.

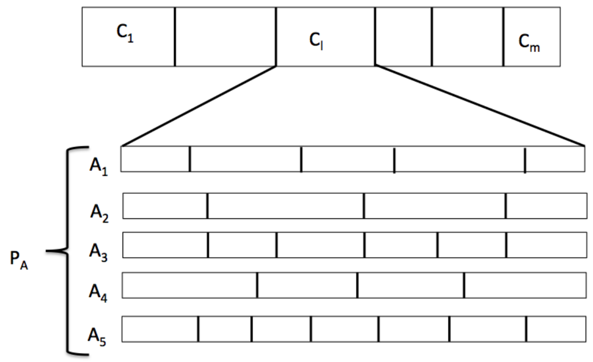


Fig. 6. From a set C_i of a partition, it can be induced a partition for each attribute A_j taking into account the values that such attribute holds in each one of the records of the set.

Let us suppose now that in Analyze-set there is the situation such that AU is lower than *long* and non-empty. The next step is, for each quasi-confidential attribute A_i describing the records, to induce a partition of the elements in C_i according to the values of A_i. The idea is to take a small number of records trying to find more commonalities among them, i.e., AU with an appropriate length. The way to do this is to take one of the remaining attributes and induce a partition according to the values that this attribute can take. Each partition has a certain number of sets. A partition with a high number of sets corresponds to a situation where each set has a small number of records but it is more likely than these records have many common attributes. Therefore, AU is specific and satisfied by a small number of records. Conversely partitions with small number of sets corresponds to a situation where each set has a high number of records but it is more likely than these records have not many common attributes. This means that if we are able to find an AU of appropriate length (higher than *long*), it will generalize many objects. However, most of time this will not be the case and Analyze-set should be called again in order to reduce the number of records included in a set.

Let P_A be the set of all the induced partitions. This set could be ordered in several ways, so the input parameter *condition* indicates how the partitions in P_A have to be sorted. There are two simple sorts:

– to use $<$ as *condition*, meaning that partitions with the lowest number of sets are given first. In the example of Fig. 6 the attributes should be ordered in the following way: A_2, A_4, A_1, A_5, A_3 and A_6;
– to use $>$ as *condition*, meaning that partitions with the highest number of sets are given first. In the example of Fig. 6 the attributes should be ordered in the following way: A_6, A_3, A_1, A_5, A_2 and A_4.

Because several attributes can induce partitions with the same number of sets, let P_{cond} be the subset of P_A of the partitions with the same number of sets selected according to *condition*. In the example show in Fig. 6, $P_{<} = \{A_2, A_4\}$ and $P_{>} = \{A_6\}$.

Given a partition $P_{a_i} \in P_{cond}$, for each set C_j of P_{a_i} with cardinality higher than K, the function Analyze-set is recursively called. At the end of the process, Analyze-set has build a set of patterns each one representing the anonymization of at least K original records.

During the anonymization process, the sets with cardinality lower than K are discarded. This means that, at the end of the process, some records that do not satisfy any of the patterns can remain. For this reason, what we propose is to repeat the whole process on the subset of no identified records using a different *condition* (see Section 4).

Notice that, some records can satisfy more than one pattern. This implies that the risk of re-identification can be, for some records, much less than $1/K$ because the anonymity set for them can be K or $2K$, or even larger.

4 Experiments

We performed experiments on the Adult data set from the Machine Learning Repository [3]. This data set is composed of 48842 records (with unknown values) described by 14 atributes. As it was done by Iyengar [5] we considered only eight of these attributes: *age, workclass, education, marital status, occupation, race, sex, native country* and, in addition, the class label *salary*. The attribute *age* is numerical and we discretized it in intervals of 5 (i.e., [20-25), [25, 30) and so on). We also considered the labels *low-19* and *high-90* to include those records placed on both sides of the global age range. All the other attributes are considered categorical. As in [6] we considered the class *salary* as confidential, being all the other attributes quasi-confidential. Commonly, authors [5, 6] discard around 3000 records due to unknown values. In our experiments we do not need to do so because the algorithm is able to deal with unknown values. The data set as it is downloaded from the Machine Learning Repository, is already split in a training set having 32561 records, and a test set having 16281 records.

In the experiments, we address the classification task. The goal is to classify people with salary up to 50K and down to 50K. Therefore the correct partition has only two sets. To use the anonymization algorithm we need to fix the input parameters K and *long* and also to determine the *condition* under which the partitions are selected. We experimented with $K = 5, 10, 20$ and 30. Concerning *long*, we have set it to 2, that is to say, the maximum number of attributes that can have value *indifferent* in the patterns is 2. We also experimented with two combinations of *condition*:

– Combination $(<, >)$: first Analyze-partition is called with $<$. At the end of the process, the remaining records are used as input of Analyze-partition using $>$ as *condition*;
– Combination $(>, <)$: first Analyze-partition is called with $>$. At the end of the process,the remaining records are used as input of Analyze-partition using $<$ as *condition*.

Once the anonymization process is finished, what we observe first is that many records are discarded since they are not satisfied by any pattern. Thus, when $K = 5$ the number of discarded records goes from around 2500 to around 3300 depending on the combination of conditions. When K increases the number of discarded records also increases, since as long as the process advances, the sets of the partitions have low cardinality. This represents the lost of between a 7% when $K = 5$ and to the 30% when $K = 30$ of the records of the training set. Because we want to address the classification task, we have to evaluate how this lost affects the predictivity of the model we obtain after anonymization.

A simple way to test the equivalence of the original data base and the anonymized one for the classification task, is to induce a domain model (for instance, using a decision tree) for each data base and then evaluate the accuracy of the models on a test set. To construct the models of both the anonymized data base and the original one, we used the J48 inductive learning method, a clone of C4.5 [7] provided by Weka [4]. To test the accuracy of the models, we cannot use 10-fold cross-validation on the anonymized data base, but we have to use original records as test set. This is because we want to evaluate how well the model induced from the patterns represents the original records. Notice that a cross-validation on the anonymized data base will test the patterns (generalized records) instead of the original records. Thus, we carry out experiments with three test sets:

– *Experiment 1*: We randomly selected a subset of 11486 records, namely $S1$, from the original data base to act as test set.
– *Experiment 2*: The test set of 16281 records, namely *uci*, as it was downloaded from the UCI Repository.
– *Experiment 3*: The whole original data base (DB) is used as test set.

Therefore the process we carry out in the experiments has been the following:

1. To induce a decision tree from DB and evaluate the accuracy of that model on $S1$, DB itself, and *uci*;
2. To anonymize DB using the combination $(<, >)$, then induce a decision tree from the patterns that have been generated and, finally, evaluate the accuracy of that model on $S1$, DB, and *uci*;
3. To anonymize DB using the combination $(>, <)$, then induce a decision tree from the patterns that have been generated and, finally, evaluate the accuracy of that model on $S1$, DB, and *uci*.

Table 1 shows the accuracies obtained by each model with different values of K on each one of the test sets. Concerning the accuracy, we see that the combination $(>, <)$ is better than $(<, >)$ for all K except for $K = 30$. We have not any satisfactory explanation for this. However our intuition is that because the combination $(>, <)$ selects first the partitions having highest number of sets, the cardinality of these sets will be low. Probably, small sets are composed of more similar objects whose anti-unification will give a satisfactorily long pattern.

Table 1. Accuracy for $K = 5, 10, 20$ and 30 of the models with the original records (DB) and the anonymized ones ($(<, >)$ and $(>, <)$) on three different test sets: $S1$ with around 16000 records randomly selected; the training set provided by the UCI Machine Learning Repository (*uci*); and, the whole data set (DB).

model	test	$K = 5$	$K = 10$	$K = 20$	$K = 30$
DB	$S1$	83.32	83.32	83.32	83.32
$(<, >)$	$S1$	73.19	72.51	80.38	80.58
$(>, <)$	$S1$	81.28	81.51	82.05	72.18
DB	*uci*	83.12	83.12	83.12	83.12
$(<, >)$	*uci*	73.23	72.38	80.34	80.82
$(>, <)$	*uci*	81.01	81.75	81.82	72.18
DB	DB	83.42	83.42	83.42	83.42
$(<, >)$	DB	73.01	72.12	80.27	80.71
$(>, <)$	DB	81.42	81.66	81.72	71.73

Table 2. Number of patterns (*patterns*) and number of discarded records (*rest*) for each one of the combinations.

	$K = 5$		$K = 10$		$K = 20$		$K = 30$	
model	*patterns*	*rest*	*patterns*	*rest*	*patterns*	*rest*	*patterns*	*rest*
$(<, >)$	2779	3299	1542	5231	821	7681	570	9671
$(>, <)$	10453	2578	4956	5042	2307	7396	1343	9613

The combination $(<, >)$ tries to avoid overfitting by selecting the attributes inducing partitions with small number of sets. However, these sets have with high probability, records with very different descriptions.

When $K = 20$, both combinations have similar accuracy to the one given by the model obtained from DB. Notice that, despite for $K = 20$ there are more than 7000 records discarded, the accuracy is the best of all the combinations we tested. This is an unexpected result since it means that the anonymization is able to construct a satisfactory model of the domain with less records. Our explanation for this result is that the original database probably contains many outliers, i.e., records that have very particular combinations of values.

Table 2 shows the number of patterns generated by each combination and the number of discarded records for different values of K. The combination $(>, <)$ tends to discard lower number of records than the combination $(<, >)$. The combination $(<, >)$ seems to produce a high partitioning of the sets and, consequently, a higher number of discarded records than using $(>, <)$ since they achieve cardinality lower than K. Specially interesting is the combination $(>, <)$ with both $K = 20$ and $K = 30$ that generate lesser than 1000 patterns and, however, the induced model has an accuracy very near to the one obtained with the whole data set.

5 Conclusions

In this paper we introduce a new method for k-anonymization of data bases where records are represented with attributes with categorical values. The approach is based on generalization, particularly we used the concept of anti-unification consisting on take only those attributes that take the same value in all the records of a given set. This approach is different from other generalization approaches in two main issues: first it is able to deal with unknown values; and, second it is not necessary to construct an ontology generalizing the values of the attributes. At the end of the process, we obtain an anonymized data base consisting on a set of patterns that have the same form than the original records but they are generalized. The generalization consists on setting some of the attributes to value unknown meaning that it is no important the value that they take. Because we only permit one or two unknown values in each pattern, we can assure that there is no overgeneralization, so we do not lost too much information.

There are some original records that cannot be generalized satisfactorily with any patterns, so they are discarded. We experimentally proved that the rejection of a (sometimes high) percentage of original records does not highly affect the predictive accuracy in classification tasks.

As future work we consider the application of this approach for microaggregation, in the sense that different subsets of attributes are generalized differently. This approach has already considered in [8] for other generalization algorithms with good results. In this microaggregation approach the analysis of disclosure risk is meaningful. In addition, as records may be generalized in several ways, the model described in [16] may be useful to study a worse case scenario.

Acknowledgments

This research is partially funded by the Spanish MICINN projects COGNITIO (TIN-2012-38450-C03-03), EdeTRI (TIN2012-39348-C02-01) and COPRIVACY (TIN2011-27076-C03-03), the grant 2009-SGR-1434 from the Generalitat de Catalunya, and the European Project DwB (Grant Agreement Number 262608).

References

1. D. Abril, G. Navarro-Arribas, V. Torra. Supervised Learning Using a Symmetric Bilinear Form for Record Linkage, Information Fusion 26 (2015) 144-153.
2. E. Armengol and E. Plaza. Bottom-up induction of feature terms. *Machine Learning*, 41(1):259–294, 2000.
3. K. Bache and M. Lichman. UCI machine learning repository, 2013.
4. Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

5. Vijay S. Iyengar. Transforming data to satisfy privacy constraints. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 279–288, New York, NY, USA, 2002. ACM.
6. Roberto J. Bayardo Jr. and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *Proceedings of the 21st International Conference on Data Engineering, ICDE 2005, 5-8 April 2005, Tokyo, Japan*, pages 217–228, 2005.
7. J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
8. Marés, J., Torra, V. (2012) Clustering-based Categorical Data Protection, Proc. PSD 2012, Lecture Notes in Computer Science 7556 78-89.
9. LeFevre, K., DeWitt, D. J., Ramakrishnan, R. (2005) Multidimensional k-anonymity, Technical Report 1521, University of Wisconsin.
10. LeFevre, K., DeWitt, D. J., Ramakrishnan, R. (2005) Incognito: Efficient Full-Domain K-Anonymity, SIGMOD 2005.
11. Domingo-Ferrer, J., Torra, V. (2005) Ordinal, Continuous and Heterogeneous k-Anonymity Through Microaggregation, Data Mining and Knowledge Discovery 11:2 195-212.
12. Duncan, G. T., Elliot, M., Salazar, J. J. (2011) Statistical confidentiality, Springer.
13. Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E. S., Spicer, K., de Wolf, P.-P. (2012) Statistical Disclosure Control, Wiley.
14. Samarati, P., Sweeney, L. (1998) Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression, SRI Intl. Tech. Rep.
15. Dwork, C. (2006) Differential privacy, Proc. ICALP 2006, Lecture Notes in Computer Science 4052, pp. 1-12.
16. T. Tassa, A. Mazza, A. Gionis (2012) k-Concealment: An Alternative Model of k-Type Anonymity, Transactions on Data Privacy 5:1 189 - 222
17. Torra, V., Stokes, K. (2012) A formalization of record linkage and its application to data protection, Int. J. Unc. Fuzz. Knowl. Based Syst. 20 907-919.
18. Winkler, W. E. (2004) Re-identification methods for masked microdata, Proc. PSD 2004, Lecture Notes in Computer Science 3050 216-230.