Steered Microaggregation: A Unified Primitive for Anonymization of Data Sets and Data Streams

Josep Domingo-Ferrer and Jordi Soria-Comas
Dept. of Computer Science and Mathematics, Universitat Rovira i Virgili
Av. Països Catalans 26, E-43007 Tarragona, Catalonia
E-mail {josep.domingo,jordi.soria}@urv.cat

***Abstract*—The data anonymization landscape has become quite complex in the last decades. On the methodology side, the statistical disclosure control methods designed in official statistics have been supplemented by a number of privacy models proposed by computer scientists. On the data side, static data sets now coexist with big data, and particularly data streams. In the quest for a unified and conceptually simple anonymization approach, we present here a primitive called steered microaggregation that can be tailored to enforce various privacy models both on static data sets and also on data streams. This type of microaggregation relies on adding artificial attributes that are properly initialized and weighted in order to steer the microaggregation process into meeting certain desired constraints. Although not limited to these, we demonstrate the potential of steered microaggregation by showing how it can be used to achieve t-closeness in the context of static data sets and to achieve k-anonymity of data streams while controlling tuple reordering.**

***Index Terms*—Privacy, anonymization, statistical disclosure control, microaggregation, k-anonymity, t-closeness, l-diversity, ϵ-differential privacy, data streams.**

I. Introduction

Anonymization is usually carried out by modifying the original data using a method for statistical disclosure control (SDC, [14]). The traditional approach to anonymization in official statistics was (and still largely is) to first try an SDC method with parameters that degrade utility only mildly, and then measure the disclosure risk on the anonymized data [31], [15], [11]. If necessary, re-anonymize with more aggressive parameters until the risk is low enough. In computer science, a more formal approach was introduced, based on a *privacy model*. A privacy model is a privacy condition, dependent on a parameter, that guarantees an upper bound on the risk of reidentification disclosure and perhaps also on the risk of attribute disclosure by an intruder. The model can be enforced using one or several SDC methods whose parameters derive from the model parameters.

k-Anonymity [25] is a privacy model that focuses on quasi-identifier attributes (attributes such as age, birthplace, job, etc., each of which does not uniquely re-identify a subject, but whose combination may). The model generalizes or averages quasi-identifier attribute values, so that each combination of quasi-identifier attribute values is shared by a group of at least k records (the so-called k-anonymous class). In this way, the reidentification risk (the risk that the subject corresponding to an anonymized record can be reidentified) is at most $1/k$.

While it protects against reidentification disclosure, k-anonymity fails to guard against attribute disclosure, which occurs if the value of a sensitive attribute, *e.g* salary, health condition, etc., can be inferred for an individual subject (this is possible even without reidentification, when the sensitive attribute values for the records in a k-anonymous class are very similar). Many extensions of k-anonymity exist whose purpose is to offer protection against attribute disclosure: l-diversity [18], t-closeness [16], (n,t)-closeness [17], crowd-blending privacy [12] and others.

A. Contribution and plan of this paper

Anonymization has become quite a challenging endeavor in the last decades, both due to the appearance of new types of data (in particular, dynamic vs static data) and due to the multiplicity of privacy models. In this paper, we seek to offer a primitive, called steered microaggregation, that is general enough to be able to satisfy several privacy models on static data sets and also on data streams. As its name suggests, our primitive is based on microaggregation, which is an SDC family of methods based on partitioning the data set into groups of k or more records and replacing records in each group by the group centroid; microaggregation was shown to yield k-anonymity if used on quasi-identifier attributes [8].

The goal in steered microaggregation is to define a general control mechanism to guide microaggregation algorithms. That is, rather than merely generating clusters of a given cardinality that are as homogeneous as possible, steered microaggeration allows taking into account additional constraints. This is achieved by introducing artificial attributes designed in specific ways.

After giving some background in Section II, we present steered microaggregation in Section III. Then, we first focus on a static data set, which is the scenario that most privacy models implicitly assume. As the first application of steered microaggregation, in Section IV we show how it can be used to achieve t-closeness, one of the privacy models in the k-anonymity family offering strictest privacy guarantees.

The following funding sources are gratefully acknowledged: European Commission (projects H2020-644024 "CLARUS" and H2020-700540 "CANVAS"), Government of Catalonia (ICREA Acadèmia Prize to J. Domingo-Ferrer) and Spanish Government (projects TIN2014-57364-C2-1-R "SmartGlacis" and TIN2015-70054-REDC). The views in this paper are the authors' own and do not necessarily reflect the views of UNESCO or any of the funders.

We then move to a usual big data scenario: data that change dynamically, either by periodic updates or continuously. Anonymizing incremental updates with notions inspired by k-anonymity has been addressed by [20], [34], [3], [22], [32], [27], [9]. Nevertheless, scaling these approaches to *continuously* updated data is not obvious. Anonymization in continuous data publishing is treated by [4], who proposed a method to achieve k-anonymity and l-diversity for a data stream. As another application of steered microaggregation, in Section V we show how it can be used to satisfy k-anonymity in data streams, with the novel feature of keeping under control the amount of tuple reordering. Experimental work comparing our approach with previous work on stream k-anonymity is presented in Section VI. Finally, conclusions and future research lines are gathered in Section VII.

II. Background

A. k-Anonymity

Assume a data set D from which direct identifiers have been suppressed, but that contains so-called *quasi-identifier* attributes, that is, attributes (*e.g.* age, gender, nationality, etc.) which can be used by an intruder to link records in D with records in some external database containing direct identifiers. The intruder's goal is to determine the identity of the individuals to whom the values of sensitive attributes (*e.g.* health condition, salary, etc.) in records in D correspond (*identity disclosure*). See [14] for further details on disclosure attacks.

Definition 1. *A data set D is said to satisfy k-anonymity [25], [24] if each combination of values of the quasi-identifier attributes in it is shared by at least k records.*

We use the term *equivalence class* to refer to a maximal set of records that are indistinguishable with respect to the quasi-identifiers. k-Anonymity protects against identity disclosure: given an anonymized record in D, an intruder cannot determine the identity of the individual to whom the record (and hence the sensitive attribute values in it) corresponds. The reason is that there are at least k records in D sharing any combination of quasi-identifier attribute values.

k-Anonymity is attained by modifying the values of the quasi-identifier attributes, usually either by a combination of generalization and suppression or by using microaggregation.

B. t-Closeness

t-Closeness [16] is an extension of k-anonymity which also tries to solve the attribute disclosure problem.

Definition 2. *An equivalence class is said to satisfy t-closeness if the distance between the distribution of a sensitive attribute in this class and the distribution of the attribute in the whole data set is no more than a threshold t. A data set is said to satisfy t-closeness if all its equivalence classes satisfy t-closeness.*

The specific distance used between distributions is central to evaluate t-closeness, but the original definition does not advocate any specific distance. The Earth Mover's distance (EMD) [23] is the most common choice (and the one we will adopt in this work), although other distances have also been explored [21], [7]. $EMD(P,Q)$ measures the cost of transforming one distribution P into another distribution Q by moving probability mass. EMD is computed as the minimum transportation cost from the bins of P to the bins of Q, so it depends on how much mass is moved and how far it is moved. For numerical attributes the distance between two bins is based on the number of bins between them. If the numerical attribute takes values $\{v_1,\ldots,v_m\}$, where $v_i < v_j$ if $i < j$, then $ordered\_distance(v_i,v_j) = |i-j|/(m-1)$. Now, if P and Q are distributions over $\{v_1,\ldots,v_m\}$ that, respectively, assign probability p_i and q_i to v_i, then the EMD for the ordered distance can be computed as

$$EMD(P,Q) = \frac{1}{m-1}\sum_{i=1}^{m}\left|\sum_{j=1}^{i} p_j - q_j\right|. \qquad (1)$$

C. Microaggregation

Microaggregation is a family of perturbative methods for statistical disclosure control of microdata releases. Multi-dimensional microaggregation was proposed and formalized in [5] and its application to k-anonymity was described in [8]. It consists of the following two steps:

- *Partition:* The records in the original data set are partitioned into several clusters, each of them containing at least k records. To minimize the information loss, records in each cluster should be as similar as possible.
- *Aggregation:* An aggregation operator is used to summarize the data in each cluster and the original records are replaced by the aggregated output. For numerical data, one can use the mean as aggregation operator; for categorical data, one can resort to the median or some other average operator defined in terms of an ontology (*e.g.* see [6]).

The partition and aggregation steps produce some information loss. The goal of microaggregation is to minimize the information loss according to some metric. A common information loss metric is the SSE (sum of squared errors). When using SSE on numerical attributes, the mean is a sensible choice as the aggregation operator, because for any given partition it minimizes SSE in the aggregation step; the challenge thus is to come up with a partition that minimizes the overall SSE. Finding an optimal partition in multi-dimensional microaggregation is an NP-hard problem [19]; therefore, heuristics are employed to obtain an approximation with reasonable cost. An example heuristic for the partition step of microaggregation is MDAV [8].

The limitations to re-identification imposed by k-anonymity can be satisfied without aggregating the values of the quasi-identifier attributes within each equivalence class after the partition step. It is less utility-damaging to break the relation between quasi-identifiers and sensitive attributes while preserving the original values of the quasi-identifiers. This is the

approach to attain k-anonymity-like guarantees taken in [33], [28].

III. Steered microaggregation

We define *steered microaggregation* as a form of microaggregation in which the clustering process is steered (guided) by introducing one or several artificial attributes, each having a suitable weight. A more formal definition follows.

Definition 3. *[Steered microaggregation] A steered microaggregation algorithm is an algorithm $SM(D, k, M, \mathcal{A}, \Omega)$, where D is an original data set, k is the microaggregation parameter (minimum cluster size), M is a microaggregation algorithm, $\mathcal{A}$ is a set of artificial attributes each with its corresponding values for all records in D, and Ω is a set of weights containing one weight in Ω for each attribute in $\mathcal{A}$. Then, SM operates as follows:*

1) *Augment the data set D by adding to it the attributes in $\mathcal{A}$ (thereby increasing the dimensionality of D). Call the augmented data set D'. In this way, each record in D' has now one additional value per additional attribute.*
2) *Run the microaggregation algorithm M on D', in such a way that the weight of each attribute in $\mathcal{A}$ when computing distances between records to partition D' is the corresponding weight in Ω.*

If all attributes in D' are numerical, the easiest way to enforce the weights of $\mathcal{A}$ is to normalize all attributes in D between 0 and 1 (for each attribute, subtract from each attribute value the attribute minimum over the data set and divide the result by the difference between the attribute maximum and the minimum over the data set). Then normalize each attribute in $\mathcal{A}$ between 0 and its corresponding weight in Ω. After that, M can be used on the normalized data set without any further concern about weights.

IV. t-Closeness via steered microaggregation

k-Anonymity can be attained via microaggregation of the quasi-identifier attributes. Since k-anonymity only protects against re-identification disclosure, extensions to k-anonymity (such as t-closeness) are needed to achieve protection against attribute disclosure. However, running standard microaggregation algorithms on quasi-identifiers cannot ensure the level of variability of the sensitive attribute required by such extensions, because attribute disclosure depends on the sensitive attribute.

The problem is thus how to take into account the sensitive attribute in the microaggregation process. Performing *ad hoc* modifications of specific microaggregation heuristics to satisfy each specific k-anonymity extension is certainly an option. However, it requires modifying the microaggregation algorithms and does not provide an easy way to balance cluster homogeneity (dependent on quasi-identifiers) and sensitive attribute variability. Steered microaggregation offers a more general alternative by steering standard microaggregation algorithms to enforce the specific constraints of the chosen k-anonymity extension. We focus on t-closeness in this section.

t-Closeness requires the distribution of the sensitive attribute in each k-anonymous class to be similar to the distribution of the sensitive attribute in the overall data set. Thus, to attain t-closeness, we must make sure that the sensitive attribute values of the records in each microaggregation cluster are sufficiently spread across the values in the overall data set. This can be ensured by: (i) splitting the records in the data set into b equal sized buckets $B_1, \ldots, B_b$, where B_1 contains the n/b smallest sensitive attribute values, B_2 contains the following n/b values, and so on; and (ii) steering the microaggregation algorithm into forming clusters such that each cluster contains one record of each of the previous buckets.

The value chosen for parameter b (number of buckets) determines the level of t-closeness that can be reached. In particular, as t decreases (stronger protection), b must increase (the sensitive attribute in microaggregation clusters must be more spread). For a given level of t-closeness, we need to determine the required value for b. According to Proposition 2 in [30], when using EMD as defined in Equation (1) we need $b \geq n/(2(n-1)t+1)$, where n is the number of records in the data set. In particular, since t-closeness also requires k-anonymity, we have to take

$$b = \max\left\{k, \left\lceil \frac{n}{2(n-1)t+1} \right\rceil\right\}. \tag{2}$$

We propose Algorithm 1 for t-closeness via steered microaggregation.

Algorithm 1. *[t-closeness via steered microaggregation]*

Let $Q_1, \ldots, Q_m$ *be the quasi-identifier attributes and* C *be the sensitive attribute of* D.
Let k *be the expected level of* k*-anonymity,* t *the expected level of* t*-closeness,* n *the number of records in* D*, and* w_c *the weight that signals the importance of* t*-closeness.*

1) *Normalize $Q_1, \ldots, Q_m$ for their range to be in $[0, 1]$. Let $\hat{Q}_1, \ldots, \hat{Q}_m$ be the normalized attributes.*
2) ***Set*** $b = \max\left\{k, \left\lceil \frac{n}{2(n-1)t+1} \right\rceil\right\}$.
3) *Define buckets $B_1, \ldots, B_b$ clustering the values of C, in such a way that all buckets contain the values of the same (or a similar) amount of records and the values within each bucket are maximally homogeneous. Without loss of generality, let B_1 be the bucket with the largest number of records having values of C in it, let B_2 be the bucket with the second largest number of records, and so on.*
4) *Add a bucket attribute $B(C)$ and initialize its value to 0 for all n records.*
5) **For** $i = 1$ **to** $|B_1|$ **do**
   a) *Take as the i-th record one of the records such that C falls in B_1 and $B(C) = 0$;*
   b) **Set** $B(C) := i$ *for that record.*
6) **For** $j = 2$ **to** b **do**
   a) **Let V** *be a vector with $|B_{j-1}|$ binary positions all initialized to 0.*
   b) **For** $i = 1$ **to** $|B_j|$ **do**

i) *Take as the i-th record one of the records such that C falls in B_j and $B(C) = 0$;*
ii) *Look for the record i' in B_{j-1} with $\mathbf{V}[i'] = 0$ and with closest values of normalized attributes $\hat{Q}_1, \ldots, \hat{Q}_m$ to the ones of the i-th record in B_j.*
iii) **Set** $\mathbf{V}[i'] = 1$.
iv) **Set** *$B(C)$ for the i-th record in B_j to the value of $B(C)$ for the i'-th record in B_{j-1}.*

7) *Normalize $B(C)$ for its range to be in $[0, \omega_c]$. Let $\hat{B}(C)$ be the normalized bucket attribute.*
8) *Microaggregate with group size b the projection on attributes $\hat{Q}_1, \ldots, \hat{Q}_m, \hat{B}(C)$. Call $Q'_1, \ldots, Q'_m, B(C)'$ the microaggregated attributes.*
9) *De-normalize $Q'_1, \ldots, Q'_m$ in the records returned by the microaggregation heuristic and output the records (where, for each record, the values of $Q'_1, \ldots, Q'_m, C$ are output).*

Algorithm 1 seeks homogeneity for the values of $Q_1, \ldots, Q_m, B(C)$. Attribute $B(C)$ is an artificial attribute, with weight ω_c, that introduces an additional constraint in the microaggregation process. $B(C)$ is initialized in such a way that it sequentially numbers the records whose sensitive attribute value falls in each bucket: if there are $|B_1|$ records with values of C in B_1, then $B(C)$ for those records takes values $1, 2, \ldots, |B_1|$; if there are $|B_2|$ records with values of C in B_2, then $B(C)$ for those records takes values $1, 2, \ldots, |B_2|$; and so on. Hence, if the weight ω_c attached to $B(C)$ is large enough, clusters containing records with the same value for $B(C)$ will be favored, which means clusters such that the values of C in the records of each cluster all fall in different buckets. Spreading C over all buckets within each cluster is a way to obtain a within-cluster distribution of C that resembles the distribution of C in the overall dataset. If ω_c is large enough for the value of t, this should yield t-closeness.

The following proposition (whose proof is left as an exercise) quantifies the weight ω_c needed to make sure that all values of C in a k-anonymous class fall in different buckets. This is a necessary condition for t-closeness.

Proposition 1. *If $|B_1|$ is the size of the largest bucket and the Euclidean distance is used to calculate the distance between records and $\omega_c > |B_1|\sqrt{m}$, then Algorithm 1 ensures that, in any microaggregation cluster, the values of C for all records fall in different buckets.*

By using the weight w_c proposed in the previous proposition, we ensure that t-closeness will be strictly satisfied. However, by using a lower w_c we have the ability to trade off t-closeness and utility preservation, that is, the generation of clusters that are as homogeneous as possible in terms of the quasi-identifiers.

V. k-ANONYMITY FOR DATA STREAMS VIA STEERED MICROAGGREGATION

Although the bulk of the research in SDC refers to static data sets, dynamic data are gaining relevance in the big data context, as mentioned in the introduction. In this section we focus on data streams, a particular kind of dynamic data. Attributes relevant to anonymization are identifiers, quasi-identifiers and sensitive attributes. Without loss of generality, we consider a single sensitive attribute (it can be the Cartesian product of several sensitive attributes). Attributes of other types are omitted, because they are irrelevant to anonymization. Thus:

- Let $S = \{s_i\}_{i \geq 0}$ be an original stream, that is, a sequence of continuously incoming tuples $s_i = (id, q_1, \ldots, q_m, c)$, where $s_i.id$ is the identity of the subject to whom s_i corresponds; $s_i.q_1$, ..., $s_i.q_m$ are the values for quasi-identifier attributes $Q_1, \ldots, Q_m$ and $s_i.c$ is the value of the sensitive attribute C.
- Let $S' = \{s'_j\}_{j \geq 0}$ be the anonymized version of stream S, that is, a sequence of continuously output tuples $s'_j = (q_1, \ldots, q_m, c)$, where the subject's identity has been pruned, and $s'_j.q_1$, ..., $s'_j.q_m$, s'_j, c are suitably modified versions of the quasi-identifier values and the sensitive attribute value, respectively, in the original tuple corresponding to s'_j.

Anonymizing a stream may involve some delay and reordering of the tuples in the original stream. Such delay and reordering of the tuples is one dimension of the information loss inflicted by anonymization, the other dimension being related to how much the quasi-identifier attribute values of the tuples are altered. We slightly generalize the delay constraint definition of [4].

Definition 4. *[Delay constraint] Let M be an anonymization mechanism that takes as input a data stream S and outputs an anonymized data stream S'. If δ is a positive integer, M is said to satisfy the delay constraint δ if and only if upon receiving any new tuple $s_i \in S$, M has already output in S' all anonymized tuples corresponding to tuples in S with position less than $i - \delta$.*

We next specify a formally refined version of the definition of data stream k-anonymity given in [4].

Definition 5. *[Stream k-anonymity] An anonymized data stream S' is a k-anonymized version of an original data stream S if both the following conditions hold:*

- *For each tuple $s' \in S'$, there exists in S the corresponding original tuple s.*
- *Given any tuple $s' \in S'$, a group G of anonymized tuples exists such that:*
  1) *$s' \in G$;*
  2) *$|DS(G)| \geq k$ where $DS(G)$ is the set of distinct subjects to whom tuples in G refer;*
  3) *$\forall s'' \in G$, one has $s''.q_j = s'.q_j$ for $j \in [1, m]$.*

The group G is called the k-anonymous class of s'.

In a static data set, using a standard microaggregation algorithm on the projection of the data set on the quasi-identifiers leads to k-anonymity. However, standard microaggregation algorithms cannot be used in data streams, because they operate on the entire data set, whereas in a data stream all that is available at any moment is a certain number of incoming tuples. Hence, we need to propose a microaggregation algorithm that is suitable for data streams.

Any microaggregation algorithm for data streams needs a set of tuples out of which the algorithm can generate clusters that are as homogeneous as possible. Therefore, the algorithm must buffer incoming tuples. The larger the buffer size, the more homogeneous the output clusters can be (and the less information loss is incurred), but the longer the delay (see Definition 4). Thus, the maximum affordable delay determines the buffer size.

Algorithm 2 keeps buffering tuples until there is a tuple that is about to expire (its delay is the maximum value δ). Since such a tuple must immediately be output, the algorithm generates a cluster around the expiring tuple by taking k tuples from the buffer, replaces each of the tuples in the cluster by the cluster centroid, and outputs the resulting cluster.

Algorithm 2. *[Stream microaggregation with maximum delay δ]*

Let S *be a data stream,* δ *the maximum delay allowed and* k *be the minimum cluster size.*
While $S \neq \emptyset$ ***do****:*

1) *Read the next tuple* s_i *into the buffer* $\mathcal{B}$.
2) **If** *tuple* $s_{i-\delta}$ *is in* $\mathcal{B}$ **then**

   If $|\mathcal{B}| \geq k$ ***then***

   a) *Out of the tuples in* $\mathcal{B}$*, generate a cluster of* k *tuples including tuple* $s_{i-\delta}$ *that is maximally homogeneous in terms of the quasi-identifier attributes.*
   b) *Remove the tuples in the generated cluster from* $\mathcal{B}$.
   c) *Replace the value of each quasi-identifier attribute in each of the tuples in the generated cluster by the cluster centroid value of the attribute.*
   d) *Output the resulting cluster.*

   Else *Drop* $s_{i-\delta}$.

End while

While Algorithm 2 is simple and meets the delay constraint, it makes no attempt at preserving the order of any of tuples when outputting them: tuples are reordered as much as needed to satisfy maximum quasi-identifier homogeneity within the delay constraint. This lack of regard for order preservation is a shortcoming shared by other stream k-anonymization proposals in the literature, such as the CASTLE algorithm [4].

However, preserving the order of tuples when anonymizing a data stream may also be regarded as a utility feature (the less reordering, the more utility). Steered microaggregation provides a way to integrate the order of tuples in the microaggregation process. This is done by adding an artificial attribute to the incoming tuples that contains the tuple's position number. As far as microaggregation is concerned, this artificial attribute is treated as an additional quasi-identifier attribute. By attaching a suitable weight to the position attribute during the microaggregation process, we can trade off order preservation and homogeneity regarding the other quasi-identifiers. Specifically, we propose Algorithm 3, where for simplicity we assume in the algorithm description that all quasi-identifiers are numerical (see Section III for indications about steered microaggregation on non-numerical attributes):

Algorithm 3. *[Stream k-anonymity(S,δ,ω_p,k)]*
Let S *be a data stream with quasi-identifiers* $Q_1, \ldots, Q_m$ *and sensitive attribute* C, δ *be the maximum delay allowed,* ω_p *the weight attached to the position number and* k *the minimum cluster size.*

1) *Add a new attribute,* P*, containing the position tuple of each tuple in* S *times* ω_p*, that is,* $s_i.P = i \times \omega_p$.
2) *Run Algorithm 2 taking* $Q_1, \ldots, Q_m, P$ *as quasi-identifiers.*

To understand the rationale behind Algorithm 3, note that the partition step of microaggregation forms a cluster such that the within-cluster values of $Q_1, \ldots, Q_m, P$ are as homogeneous as possible, in order to lose as little information as possible when replacing them by the centroid values in the aggregation step. Now, if the weight ω_p is large, the distance between tuples used in the partition step basically amounts to the distance between the projection of the tuples on attribute P. Hence, to maximize within-cluster homogeneity, microaggregation will output clusters whose tuples have consecutive values of P, regardless of the values of attributes $Q_1, \ldots, Q_m$. Therefore, a large ω_p ensures that no tuple reordering will occur, although substantial quasi-identifier damage can be incurred as a consequence of quite different values of $Q_1, \ldots, Q_m$ being averaged in the same cluster. On the other hand, if ω_p is small, then clusters will be formed by microaggregation by prioritizing the homogeneity of within-cluster values of $Q_1, \ldots, Q_m$. Hence, substantial reordering and delays as large as δ can occur, because the position number is not taken into account when forming clusters: in fact, Algorithm 3 with very small ω_p is equivalent to Algorithm 2.

Specifically, the following proposition (whose proof is left as an exercise) quantifies the value of ω_p beyond which no reordering of the tuples will occur.

Proposition 2. *If attributes* $Q_1, \ldots, Q_m$ *are normalized to* $[0, 1]$*, attribute* P *is normalized to* $[0, \omega_p]$*, the Euclidean distance is used to calculate the distance between tuples and* $\omega_p > (k-1)\sqrt{\frac{m}{2k-1}}$*, then Algorithm 3 will not generate clusters whose positions differ by more than* $k-1$*, that is, no reordering will occur.*

Note that the difference between the two most distant tuple positions in a cluster of k tuples is at least $k-1$. Specifically, $k-1$ is achieved when all tuples in the cluster are consecutive, which means that no reordering occurs.

Note 1. *[Subjects with several tuples in the stream]* In Definition 5, stream k-anonymity requires that the tuples in each k-anonymous class refer to at least k different subjects. In Algorithm 3 and Proposition 2, we have implicitly assumed that all incoming tuples correspond to different subjects. However, in general, there may be several tuples referring to the same subject. To make sure k-anonymous classes refer to at least k subjects, we only need to modify Algorithm 2 so that, rather than generating a cluster of k tuples, it generates a cluster of $k' \geq k$ tuples such that they correspond to k different subjects. If such k' tuples cannot be found in $\mathcal{B}$, then $s_{i-\delta}$ has to be dropped.

VI. Empirical analysis

We envision steered microaggregation as a general methodology to guide standard microaggregation algorithms into satisfying additional constraints. In this section we empirically evaluate and compare the proposed approaches to attain t-closeness for static data (Algorithm 1) and k-anonymity for data streams (Algorithm 3).

A. Data set

We have used two data sets:

- Moderately correlated data (MCD). This is a data set extracted from the CENSUS data set [2]. This same data set was used in [30], which facilitates comparisons therewith in what follows. It consists of 1,080 records containing the numerical attributes TAXINC and POTHVAL (as quasi-identifiers), and FEDTAX (as sensitive attribute). The correlation between the quasi-identifier attributes and the sensitive attribute is around 0.5.
- Perfectly correlated data (PCD). We used this data set to test Algorithm 1. It contains two perfectly correlated attributes: one quasi-identifier attribute and one sensitive attribute. The data set contains the records (i, i) for $i = 0, \ldots, 1000$. This is the worst-case scenario for t-closeness because any increase in the variability of the sensitive attribute within a cluster entails an equal increase in the variability of the quasi-identifier.

B. Risk and utility metrics

When strictly enforcing a privacy model, the tolerable disclosure risk level is set in advance and does not need to be measured: it relates to k for k-anonymity, and to k and t for t-closeness. However, since steered microaggregation allows trading off the enforcement of the privacy model against utility, we need to measure to what extent the privacy model has been satisfied. Specifically, to assess how close we are to reaching t-closeness, we use the following measures:

- *Maximum EMD.* This is the maximum EMD distance for the sensitive attribute between the clusters output by Algorithm 1 and the overall data set. This represents the actual level of t-closeness that is reached by Algorithm 1 for a certain weight ω_c.
- *Average EMD*. This is the average EMD distance for the sensitive attribute between the clusters output by Algorithm 1 and the overall data set. We use this measure to show how near we are, in global terms, to reaching the expected level of t-closeness for a certain weight ω_c.

As utility metrics we use:

- *Mean square error (MSE)*. This is a usual information loss measure in microaggregation, and we will use it to evaluate the utility loss inflicted to quasi-identifier attributes (both in the static and streaming data scenarios). MSE measures the dispersion of the records around their corresponding centroid. It is computed as the average of the squared distance between each record/tuple and its corresponding cluster centroid. For a data set D and a microaggregation partition P, MSE is computed as

$$MSE_P = \frac{1}{|D|} \sum_{x \in D} d(x, c(C_x))^2,$$

  where C_x is the cluster in P that contains record/tuple x, and $c(C_x)$ is the centroid of C_x.
- *Mean absolute error (MAE)*. MAE is similar to MSE, but it is less sensitive to outliers. It is computed as

$$MAE_P = \frac{1}{|D|} \sum_{x \in D} |d(x, c(C_x))|.$$

- *Maximum and average within-cluster tuple position differences*. These metrics measure the amount of tuple reordering in the case of stream anonymization.

C. Evaluation results

Recall that Algorithm 1 works by partitioning the sensitive attribute values into several consecutive buckets and then steers the microaggregation process to form clusters of records whose sensitive attribute values belong to different buckets. The number of buckets required to attain t-closeness is given by Expression (2). However, using the right amount of buckets does not guarantee that the desired level of t-closeness is attained. This is the task of steered microaggregation, which trades off t-closeness satisfaction and cluster homogeneity in terms of quasi-identifier attributes.

We start the evaluation of Algorithm 1 on the PCD data set with $t = 0.1$ and $t = 0.05$. In Figure 1 we show the evolution of the average and the maximum EMD distance, the MSE, and the MAE. For the displayed indicators to have similar ranges, MSE and MAE have been computed on the data set normalized in $[0, 1]$. The MSE and MAE of the original data can be computed by multiplying by $|D|^2$ and $|D|$, respectively. As expected, the maximum and the average EMD distances decrease as the weight ω_c increases. Interestingly, the desired level of t-closeness is reached when the weight is 199 (for $t = 0.1$) and 99 (for $t = 0.05$), which are weight values quite close to the bounds given in Proposition 1 (that amount to

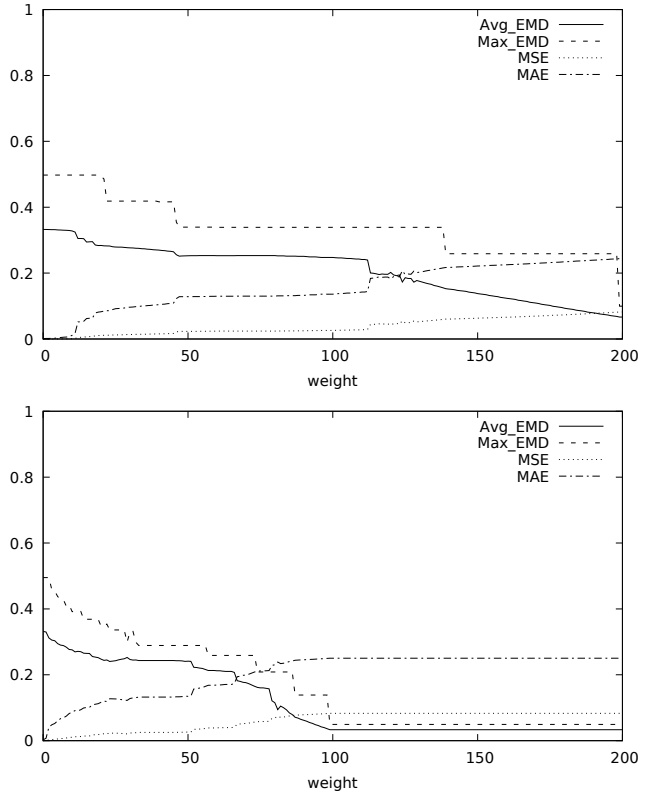

Fig. 1. Risk and utility indicators obtained from Algorithm 1 with $t = 0.1$ (top) and $t = 0.05$ (bottom) on the PCD data set as a function of the weight ω_c attached to the bucket attribute. For the displayed indicators to have similar ranges, MSE and MAE have been computed on the data set normalized in $[0, 1]$.

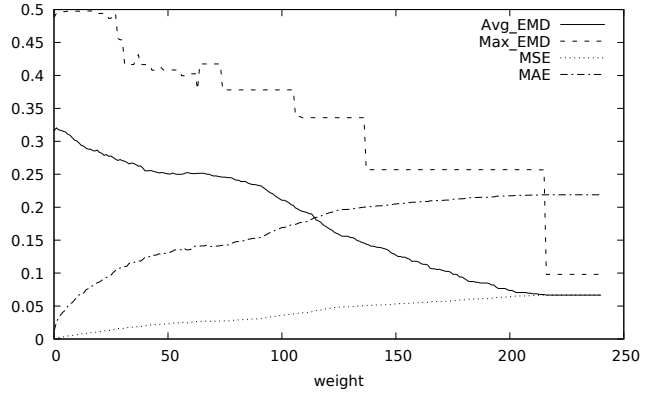

Fig. 2. Risk and utility indicators obtained from Algorithm 1 with $t = 0.1$ on the MCD data set in terms of the weight ω_c attached to the bucket attribute. MSE and MAE have been computed on normalized data as in Figure 1.

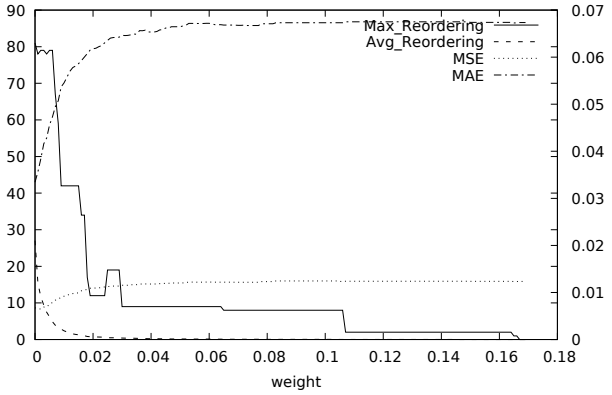

Fig. 3. Risk and utility indicators obtained from Algorithm 2 with $k = 10$ and $\delta = 100$ on the MCD data set, in terms of the weight used for the stream position.

200 for $t = 0.1$ and 100 for $t = 0.05$). In both cases, the level of t-closeness with weight 0 is around 0.5, which is the maximum possible value. The reason for this is that PCD has been constructed to make t-closeness difficult to attain. Also, it seems apparent that it is better to adjust the algorithm to the actual level of t-closeness that we want to attain. There is no benefit (rather the reverse) in setting up the algorithm with $t = 0.05$ when we want to reach $t = 0.1$: the utility losses MSE and MAE at the weight beyond which $Max\_EMD < 0.1$ are slightly greater for $t = 0.05$ than for $t = 0.1$. This is reasonable because for $t = 0.05$ the clusters are bigger, which makes them likely to be less homogeneous (w.r.t. to the quasi-identifiers).

Figure 2 shows the results obtained when running Algorithm 1 with $t = 0.1$ on the MCD data set. Interestingly, the weight at which the desired level of t-closeness is reached ($\omega_c = 215$) is also close to the bound given by Proposition 1 (which is 216 in this case). Comparing with the algorithms to attain t-closeness through microaggregation presented in [30], for $t = 0.1$, Algorithm 1 yields an MSE that is similar to the best-performing algorithm in that paper, which is considerably more complex.

Regarding Algorithm 2 for stream k-anonymity, our evaluation will focus on the effect of the weight ω_p on tuple reordering in the anonymized stream. In Figure 3 we show the results obtained on the MCD data set (normalized to $[0, 1]$) and parameters $k = 10$ and $\delta = 100$. Note that $\omega_p = 1/\delta$ would normalize the artificial position attribute in the current window to a range of 1; that is, it attaches to the artificial position attribute the same importance as to the other attributes. In our case, $1/\delta = 0.01$. We observe that the most significant effect takes place for values of the weight attribute in the range $[0, 0.02]$. With $\omega_c = 0$, we only focus on minimizing MSE and MAE while keeping the delay below 100; in this case, the maximum and average reordering are quite big (80 and 25, respectively). With weight equal to 0.02, the maximum and average reordering are significantly reduced (down to 12 and 0.7, respectively) but there is a steep increase in MSE and MAE. The optimal value of the weight depends on how important is limiting the reordering vs the within-cluster homogeneity of quasi-identifier values.

VII. Conclusions and future research

In this work we have introduced the notion of steered microaggregation, which is a general approach to guide stan-

dard microaggregation algorithms into satisfying additional criteria (beyond cluster cardinality and maximal within-cluster homogeneity). This is done by adding artificial attributes with appropriate weights that influence the microaggregation. We have described two applications of steered microaggregation and we have presented experimental results on their performance:

- Enforcement of t-closeness in a static data set. This is done by introducing an artificial attribute that controls the within-cluster variability of the sensitive attribute.
- Generation of k-anonymous streams with improved preservation of the original order of the tuples. Since standard microaggregation algorithms for static data sets are not suitable for data streams, we have first proposed a basic microaggregation algorithm for stream k-anonymity that meets a maximum delay constraint, but gives no order preservation guarantees. Then, we have shown how to use steered microaggregation to improve the preservation of the original order of the tuples in the anonymized data stream.

In the static data set setting, an interesting research avenue is to explore the use of steered microaggregation to satisfy other privacy models. Particularly relevant is the case of differential privacy [10]. Even if initially presented as a privacy mechanism for answering queries, a variety of approaches for the generation of differentially private data sets have been proposed [1], [13], [29], [26]. Thanks to the connection between t-closeness and ϵ-differential privacy described in [7], steered microaggregation could possibly be used to satisfy differential privacy under specific conditions.

In what regards data streams, a challenge is whether and how steered microaggregation can satisfy k-anonymity extensions for data streams with moderate reordering and delay.

REFERENCES

[1] A. Blum, K. Ligett and A. Roth. A learning theory approach to non-interactive database privacy. In: *Proc. of the 40th Annual Symposium on the Theory of Computing-STOC 2008*, pp. 609-618, 2008.

[2] R. Brand, J. Domingo-Ferrer, and J.M. Mateo-Sanz. Reference data sets to test and compare SDC methods for protection of numerical microdata. *European Project IST-2000-25069 CASC*. http://neon.vb.cbs.nl/casc/CASCtestsets.htm

[3] Y. Bu, A. W. C. Fu, R. C. W. Wong, L. Chen and J. Li. Privacy-preserving serial data publishing by role composition. *Proc. of the VLDB Endowment* 1(1):845-856, 2008.

[4] J. Cao, B. Carminati, E. Ferrari and K.-L. Tan. CASTLE: Continuously anonymizing data streams. *IEEE Transactions on Dependable and Secure Computing*, 8(3):337-352, 2011.

[5] J. Domingo-Ferrer and J. M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and Data Enginering* 14(1):189-201, 2002.

[6] J. Domingo-Ferrer, D. Sánchez and G. Rufián-Torrell. Anonymization of nominal data based on semantic marginality. *Information Sciences* 242:35-48, 2013.

[7] J. Domingo-Ferrer and J. Soria-Comas. From t-closeness to differential privacy and vice versa in data anonymization. *Knowledge-Based Systems*, 74:151-158, 2015.

[8] J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogeneous k-anonymity through microaggregation. *Data Mining and Knowledge Discovery* 11(2):195-212, 2005.

[9] J. Domingo-Ferrer and J. Soria-Comas. Anonymization in the time of big data. In *Proc of the Intl. Conference on Privacy in Statistical Databases-PDS 2016*. LNCS 9867, Springer, pp. 57-68, 2016.

[10] C. Dwork. Differential privacy. In: *Proc. of the 33rd Intl. Colloquium on Automata, Languages and Programming-ICALP 2006*, LNCS 4052, Springer, pp. 1-12, 2006.

[11] E.A.H. Elamir and C.J. Skinner. Record level measures of disclosure risk for survey microdata. *Journal of Official Statistics* 22(3):525-539, 2006.

[12] J. Gehrke, M. Hay, E. Lui and R. Pass. Crowd-blending privacy. In: *Advances in Cryptology-CRYPTO 2012*, LNCS 7417, pp. 479-496, 2012.

[13] M. Hardt, K. Ligett and F. McSherry. A simple and practical algorithm for differentially private data release. Preprint arXiv:1012.4763v1, 21 Dec 2010.

[14] A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, E. Schulte Nordholt, K. Spicer and P.-P. de Wolf. *Statistical Disclosure Control*, Wiley, 2012.

[15] D. Lambert. Measures of disclosure risk and harm. *Journal of Official Statistics* 9(3):313-331, 1993.

[16] N. Li, T. Li and S. Venkatasubramanian. t-Closeness: privacy beyond k-anonymity and l-diversity. In: *Proc. of the 23rd IEEE Intl. Conf. on Data Engineering-ICDE 2007*, pp. 106-115, 2007.

[17] N. Li, T. Li and S. Venkatasubramanian. Closeness: a new privacy measure for data publishing. *IEEE Transactions on Knowledge and Data Engineering* 22(7):943-956, 2010.

[18] A. Machanavajjhala, J. Gehrke, D. Kiefer, and M. Venkitasubramaniam. l-Diversity: privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data* 1(1):art. no. 3, 2007.

[19] A. Oganian and J. Domingo-Ferrer. On the complexity of optimal microaggregation for statistical disclosure control. *Statistical Journal of the United Nations Economic Commission for Europe* 18(4):345-354, 2001.

[20] J. Pei, J. Xu, Z. Wang, W. Wang and K. Wang. Maintaining k-anonymity against incremental updates. In *SSDBM'07*, IEEE, p. 5-16, 2007.

[21] D. Rebollo-Monedero, J. Forné and J. Domingo-Ferrer. From t-closeness-like privacy to postrandomization via information theory. *IEEE Transactions on Knowledge and Data Engineering* 22(11):1623-1636, 2010

[22] D. Riboni, L. Pareschi and C. Bettini. JS-Reduce: defending your data from sequential background knowledge attacks. *IEEE Transactions on Dependable and Secure Computing* 9(3):387-400, 2012.

[23] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision* 40(2):99-121, 2000.

[24] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering* 13(6):1010-1027, 2001.

[25] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. SRI International Report, 1998.

[26] D. Sánchez, J. Domingo-Ferrer, S. Martínez, and J. Soria-Comas. Utility-preserving differentially private data releases via individual ranking microaggregation. *Information Fusion*, 30:1-14, 2016.

[27] E. Shmueli and T. Tassa. Privacy by diversity in sequential releases of databases. *Information Sciences* 298(20):344-372, 2015.

[28] J. Soria-Comas and J. Domingo-Ferrer. Probabilistic k-anonymity through microaggregation and data swapping. In: *FUZZ-IEEE 2012, IEEE International Conference on Fuzzy Systems*, IEEE, pp. 1-8, 2012.

[29] J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez and S. Martínez. Enhancing data utility in differential privacy via microaggregation-based k-anonymity. *VLDB Journal* 23(5):771-794, 2014.

[30] J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez and S. Martínez. t-Closeness through microaggregation: strict privacy with enhanced utility preservation. *IEEE Transactions on Knowledge and Data Engineering*, 27(11):3098-3110, 2015.

[31] V. Torra and J. Domingo-Ferrer. Record linkage methods for multidatabase data mining. In: *Information Fusion in Data Mining*, Springer, pp. 99-130, 2003.

[32] K. Wang and B. Fung. Anonymizing sequential releases. In: *KDD'06*, ACM, pp. 414-423, 2012.

[33] X. Xiao and Y. Tao. Anatomy: simple and effective privacy preservation. In: *Proc. of the 32nd Intl. Conference on Very Large Data Bases-VLDB 2006*,pp. 139-150, 2006.

[34] X. Xiao and Y. Tao. M-Invariance: towards privacy-preserving re-publication of dynamic datasets. In *SIGMOD'07*, ACM, pp. 689-700, 2007.