Privacy Protection with Genetic Algorithms

Agusti Solanas

CRISES Research Group.
UNESCO Chair in Data Privacy
Dept. Computer Engineering and Maths.
Rovira i Virgili University. Tarrragona. Catalonia (Spain)
agusti.solanas@urv.cat

Summary. In this chapter we describe some of the most important privacy problems that the Information Society has to face in a variety of fields, namely ecommerce, health care, etc. We next propose a solution based on data aggregation, which is known to be NP-hard when applied to multivariate data. Due to this computational complexity, it is necessary to use heuristics to solve the problem. We propose the use of a Genetic Algorithm (GA) conveniently tuned up for tackling the problem properly.

1 Introduction

> *"Seduced by the effortless gathering of data, we discount the costs of turning data into information, information into knowledge and knowledge into wisdom."*
>
> **B. Harris, 1987**

When Harris wrote the above sentence in 1987 he could not imagine how easy it would be in the twenty-first century to gather huge amounts of data whilst, at the same time, transforming them into information, and next into knowledge. The last step, to turn knowledge into wisdom, is still not reached by computers and is even difficult to be achieved by most humans.

We want to obtain the information we need as fast as possible, from everywhere, and at any time. To that end, our storage and communication methods have evolved e.g. from the classical paper-based mail carried by postmen to the e-mail that fleetingly crosses the world through optical fibre. This achievement is the result of years of research and efforts. Research on information and communications technologies (ICTs) is a keystone of the Europe's 7th Research Framework Programme [31] that will run until 2013. The European Union acknowledges the importance of ICTs for the growth and competitiveness of Europe, and plans to invest over € 9 billion in research on ICTs. This investment will lead to the development of more efficient techniques for gathering, storing and sharing data by using a variety of devices, namely computers, mobile phones, smart cards, PDAs and even the interactive digital terrestrial television (DTT).

The great advances achieved on ICTs in the last few years, and the new ones that we envisage, pave the way for the storage and analysis of a huge amount of information. This information, which is growing with each passing day, is gathered by a plethora of institutions, namely statistical agencies, private companies, health-care institutions and universities among others.

The balance between the right of the society to information and the right of the individuals to privacy is on the line. A number of examples of this contradictory situation can be found in our daily life.

- *Commercial information.* Supermarkets may gather information from the fidelity cards of their customers in order to improve the services they offer to them or to increase their annual benefits. On the other hand, if this information is misused, the privacy of the costumers could be in jeopardy because e.g. their consumer habits could be inferred from the data and they can be flooded with undesired advertisements.
- *E-commerce.* Electronic commerce results in the automated collection of large amounts of consumer data. This wealth of information is very useful to companies, which are often interested in sharing it with their subsidiaries or partners. Such consumer information transfer should not result in public profiling of individuals and is subject to strict regulation.
- *Statistical Agencies.* Most countries have legislation which compels national statistical agencies to guarantee statistical confidentiality when they release data collected from citizens or companies; see [25] for regulations in the European Union, [26] for regulations in Canada, and [32] for regulations in the U.S.
- *Health information.* This is one of the most sensitive areas regarding privacy. For example, in the U.S., the Privacy Rule of the Health Insurance Portability and Accountability Act (HIPAA, [13]) requires the strict regulation of protected health information for use in medical research. In most western countries, the situation is similar, which has caused e-health to become a hot topic in privacy research (see e.g. [1]).
- *Location-Based Services.* Cell phones have become ubiquitous and services related to the current position of the user are growing fast. If the queries that a user makes to a location-based server are not securely managed, it could be possible to infer the consumer habits of the user, e.g. a third party could know that a given user likes Chinese food if she usually makes queries from a Chinese restaurant.
- *RFID technology.* The massive deployment of the Radio Frequency IDentification (RFID) technology is a reality. In the near future almost every object will contain an RFID tag capable of transmitting information under the request of an RFID reader. On the one hand, this technology will increase the efficiency of supply chains and will eventually replace bar codes. On the other hand, the existence of RFID tags in almost every

object that we have, could be a privacy problem e.g. an eavesdropper in the street will be able to scan the RFID tags of someone close to him and he will be able to determine the amount of money that she has, what kind of clothes she wears and even whether she has suffered from a hip operation and she has a prosthesis.

These problems directly affect each of us in almost every aspect of our daily life. If the proper actions are not taken, our personal data could be in jeopardy due to the data storage and the advances in data mining techniques.

Releasing statistical information for public use can jeopardise the privacy of individual respondents on which the data were collected. A possible solution leveraged from the field of Statistical Disclosure Control (SDC) is the aggregation of the original data before its publication. By aggregating the data, it becomes statistically unprovable to link the released data with a given respondent and, at the same time, data utility is preserved. Unfortunately, this solution is not straightforward because the elements composing the data must be properly clustered prior to the aggregation process. The clustering of data with constraints on the size of the clusters is known to be an NP-Hard problem when the data is multivariate. Thus, it is necessary to make use of heuristics in order to properly aggregate multivariate data.

1.1 Contribution and Plan of the Chapter

In this chapter we tackle the privacy problems that the Information Society has to face. In order to address them, we propose the use of micro-aggregation as the method for distorting data and guaranteeing respondents privacy. Unfortunately, optimal micro-aggregation is known to be NP-Hard and heuristics have to be used.

We suggest the use of a GA for solving the micro-aggregation problem. Moreover, we show how to tune up a classical GA, i.e. how to codify the chromosomes, how to choose the most effective parameters for the mutation and crossover rates, etc.

The rest of the chapter is organised as follows. Section 2 gives an overview of the micro-aggregation problem and introduces some important concepts and renown micro-aggregation methods. In Sect. 3, we show how to solve the micro-aggregation problem by using a GA. Section 4 presents an improvement of the method in Sect. 3 that permits the aggregation of large files. Section 5 provides some experimental results and, finally, Sect. 6 finishes with some conclusions and future challenges.

2 Basics of Micro-Aggregation

Statistical Disclosure Control (SDC), also known as Statistical Disclosure Limitation (SDL), seeks to transform data in such a way that they can be publicly released whilst preserving data utility and statistical confidentiality, where the latter means avoiding disclosure of information that can be linked to specific individual or corporate respondent entities.

The *SDC problem* is to modify data in such a way that sufficient protection is provided whilst minimising information loss, i.e. the loss of the accuracy sought by users.

Micro-aggregation is an SDC technique consisting in the aggregation of individual data. It can be considered as an SDC subdiscipline devoted to the protection of individual data, also called micro-data. It is only recently that data collectors (statistical agencies and the like) have been persuaded to publish micro-data. Therefore, micro-data protection is the youngest SDC subdiscipline and it is experiencing continuous evolution in the last years. Micro-aggregation can be seen as a clustering problem with constraints on the size of the clusters. It is somehow related to other clustering problems (e.g. dimension reduction or minimum squares design of clusters). However, the main difference of the micro-aggregation problem is that it does not consider the number of clusters to generate or the number of dimensions to reduce, but only the minimum number of elements that are grouped in the same cluster.

When we micro-aggregate data we have to keep two goals in mind:

- *Preserving data utility.* To do so, we should introduce as less noise as possible into the data, i.e. we should aggregate similar elements instead of different ones. In the example given in Fig. 1 for a security parameter $k = 3$, groups of three elements are built and aggregated. Note that elements in the same aggregation group are similar.
- *Protecting the privacy of the respondents.* Data have to be sufficiently modified to make re-identification difficult, i.e. by increasing the number

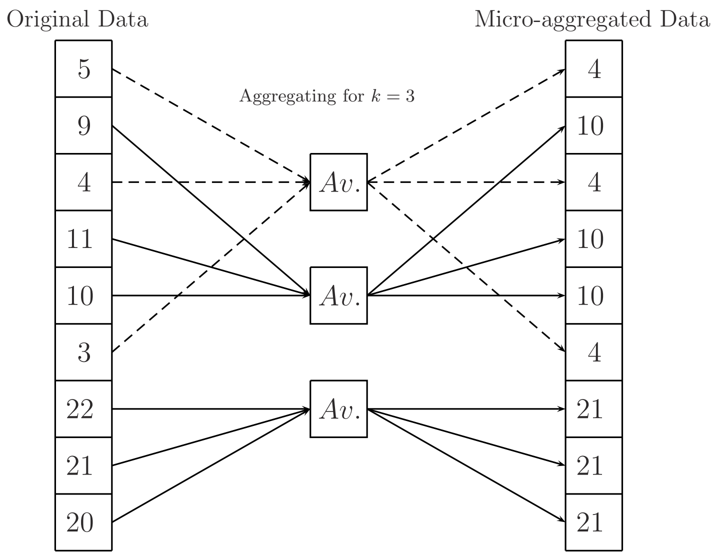

Fig. 1. k-Aggregation example with $k = 3$

of aggregated elements, we increase data privacy. In the example given in Fig. 1, after aggregating the chosen elements, it is impossible to distinguish them, so that the probability of linking any respondent is inversely proportional to the number of aggregated elements.

In order to determine whether two elements are similar, a similarity function such as the Euclidean Distance can be used. By means of this similarity function, we can define a homogeneity measure **HOM** for a given univariate data set $\mathbf{D_{uni}}$ such as the one in Expression 1.

$$\mathbf{HOM}_{\mathbf{D_{uni}}} = \sum_{i=1}^{n} \mathbf{x_i} - \bar{\mathbf{x}}, \tag{1}$$

where n is the number of elements in $\mathbf{D_{uni}}$, $\mathbf{x_i}$ is the i-th element in $\mathbf{D_{uni}}$ and $\bar{\mathbf{x}}$ is the average element, i.e. $\bar{\mathbf{x}} = \frac{\sum_{i=0}^{n} \mathbf{x_i}}{n}$. This expression can be extended to multivariate data as follows:

$$\mathbf{HOM}_{\mathbf{D_{multi}}} = \sum_{i=1}^{n} \sum_{j=1}^{d} \mathbf{x_{ij}} - \bar{\mathbf{x}}_{\mathbf{j}}, \tag{2}$$

where n is the number of elements in $\mathbf{D_{multi}}$, d is the number of dimensions of each element, $\mathbf{x_{ij}}$ is the j-th component of the i-th element in $\mathbf{D_{multi}}$ and $\bar{\mathbf{x}}_{\mathbf{j}}$ is the j-th component of the average element.

Expression 2 can be extended to multiple subsets and it is known as the Sum of Squared Errors (SSE) cf. Expression 3.

$$\mathbf{SSE} = \sum_{i=1}^{s} \sum_{j=1}^{n_i} (\mathbf{x_{ij}} - \bar{\mathbf{x}}_{\mathbf{i}})'(\mathbf{x_{ij}} - \bar{\mathbf{x}}_{\mathbf{i}}), \tag{3}$$

where s is the number of subsets, n_i is the number of elements in the i-th subset, $\mathbf{x_{ij}}$ is the j-th element in the i-th subset and $\bar{\mathbf{x}}_{\mathbf{i}}$ is the average element of the i-th subset.

Given a homogeneity measure such as the SSE and a security parameter k, which determines the minimum cardinality of the subsets, the micro-aggregation or k-micro-aggregation problem can be enunciated as follows:

> *Given a data set* $\mathbf{D}$ *built up of* $\mathbf{n}$ *elements in a characteristic space* $\mathbb{R}^d$*, the problem consists in obtaining a* k*-partition*[1] $\mathcal{P}$ *of* $\mathbf{D}$ *so that the homogeneity of* $\mathcal{P}$ *is maximised. Once* $\mathcal{P}$ *is obtained, each element of every part of* $\mathcal{P}$ *is replaced by the average element of the part.*

This problem is known to be NP-hard [23] for multivariate data sets so heuristic methods must be used to solve it.

[1] A k-partition of $\mathbf{D}$ is a partition where its parts have, at least, k elements of $\mathbf{D}$.

2.1 Multivariate Micro-Aggregation Methods

There is a plethora of methods to address the multivariate micro-aggregation problem (see e.g. [33] *for a more detailed approach*). In this section we give a brief overview of the most relevant techniques and we point out some of their shortcomings.

Minimum Spanning Tree Partitioning (MSTP)

The MSTP algorithm for micro-aggregation was proposed in [18] as a variable-size multivariate micro-aggregation method. The method is based in the well-known Minimum Spanning Tree (MST) problem. However, the standard MST partitioning algorithm does not solve the micro-aggregation problem since it does not consider the group size constraints.

In order to address this problem, the authors proposed a modification of the MSTP algorithm that takes into account the constraints on the size of the clusters. The algorithm is divided in three main steps:

1. MST construction: Constructs a MST by using the method of Prim and a Euclidean metric. Moreover, the authors augment the algorithm by adding a priority queue based on the edge length and a descendant count, that determines the number of nodes under a given node, in each node of the tree.
2. Edge cutting: The algorithm iteratively visits every MST edge in length order, from longest to shortest, and deletes the removable edges whilst retaining the remaining edges. An edge is said to be removable if the forest that results from the cutting is acceptable, i.e. each tree in the forest has at least k nodes. Note that this condition guarantees the final partition to be a k-partition.
3. Cluster generation: Once the forest has been constructed, a recursive algorithm that visits every node in the forest is applied in order to assign every node (*point*) to a cluster (*part*) (see [18] for further details). At the end of this process a k-partition is obtained.

The main limitation of this algorithm remains in its foundation, i.e. in the minimum spanning tree. Although it could be very useful when the data are distributed in clusters, it fails to properly adapt to the data when the points (elements) are distributed in a scattered way.

Maximum Distance Method (MD)

The MD was proposed in [7] as a multivariate micro-aggregation method. The main advantage of this method is its simplicity. Moreover, it achieves very good results in most data sets.

The MD algorithm builds a k-partition as follows:

1. Let r and s be the two most distant records in the data set, using the Euclidean Distance; form a group with r and the $k-1$ records closest to r; form a group with s and the $k-1$ records closest to s.
2. If there are at least $2k$ records which do not belong to any of the groups formed in Step 1, go to Step 1 taking as the new data set the previous data set minus the groups formed in the previous instance of Step 1.
3. If there are between k and $2k-1$ records which do not belong to any of the groups formed in Step 1, form a new group with those points and exit the algorithm.
4. If there are less than k records which do not belong to any of the groups formed in Step 1, add them to the closest group formed in Step 1.

At this moment, we have a k-partition of the data set. As explained above, given a k-partition, the micro-aggregated data are computed by replacing each record by the centroid of the group to which it belongs.

The main shortcoming of this method is its computational complexity, i.e. $O(n^3)$.

Maximum Distance to Average Vector Method (MDAV)

The main drawback of MD is its computational complexity because of the cost of finding the most distant records at each iteration. The Maximum Distance to Average Vector method (MDAV) improves on MD in terms of computational complexity whilst maintaining the performance in terms of resulting SSE.

MDAV was proposed in [9,15] as part of a multivariate micro-aggregation method implemented in the μ-Argus package for statistical disclosure control. A slight modification of the same construction was later described in [18] under the name centroid-based fixed size micro-aggregation.

MDAV is the most popular method used for micro-aggregating data sets. Thus, we will explain it in detail and we will compare it with our genetic approach.

MDAV builds a k-partition as follows. First, a square matrix of distances between all records is computed (line 1 in Algorithm 1). Two main approaches can be adopted to perform these distance calculations. The first approach is to compute and store the distances at the beginning of the micro-aggregation process. The second approach consists in computing the distances on the fly when they are needed. The first approach is computationally cheaper but it requires too much memory when the number of records in the data set is large.

After computing the matrix of distances, MDAV iterates (lines 4-13) and builds two groups at each iteration. In order to build these groups, the centroid c – i.e. the average vector – of the remaining records – those which have not yet been assigned to any group – is computed at the beginning of each iteration (line 5). Then the most distant record r from c is taken (line 6) and a group of

Algorithm 1: Maximum Distance to Average Vector (MDAV)

```
   Data: Data set $\mathbf{X}$, Integer k, Integer n.
   Result: $k$-partition.
1  ComputeDistanceMatrix($\mathbf{X}$);
2  $RR = n$; //Remaining Records
3  $i = 0$;
4  while $RR > 2k - 1$ do
5      $c$ = ComputeNewCentroid($\mathbf{X}$);
6      $r$ = GetFarthestRecordFromCentroid($\mathbf{X}$,$c$);
7      $s$ = GetFarthestRecordFromRecord ($\mathbf{X}$,$r$);
8      $g_i$ = BuildGroupAroundRecord($r$,$\mathbf{X}$,$k$);
9      $i = i + 1$;
10     $g_i$ = BuildGroupAroundRecord($s$,$\mathbf{X}$,$k$);
11     $i = i + 1$;
12     $RR = RR - 2k$;
13 end
14 $g_1 \dots g_s$ = AssignRemainingRec($\mathbf{X}$, $g_1 \dots g_s$);
15 return $g_1 \dots g_s$
```

k records is built around r (line 8). The group of k records around r is formed by r and the $k - 1$ closest records to r. Next, the most distant record s from r is taken (line 7) and a group of k records is built around s (line 10). The generation of groups continues until the number of remaining records (RR) is less than $2k$. When this condition is met, two cases are possible, namely $RR < k$ or $RR \geq k$. In the first case, the remaining records are assigned to their closest group. In the second case, a new group is built with all the remaining records. Note that all groups have k elements except perhaps the last one.

Finally, given the k-partition obtained by MDAV, a micro-aggregated data set is computed by replacing each record in the original data set by the centroid of the group to which it belongs. The main problem of this algorithm is its lack of flexibility. It only generates subsets of fixed cardinality k and, due to this limitation, it can suffer from a performance reduction when applied to data sets where the points are distributed in clusters of variable cardinality.

Variable-MDAV

MDAV generates groups of fixed size k and, thus, it lacks flexibility for adapting the group size to the distribution of the records in the data set, which may result in poor within-group homogeneity.

Variable-size MDAV (V-MDAV) [28] algorithm intends to overcome this limitation by computing a variable-size k-partition with a computational cost similar to the MDAV cost.

The algorithm for building a k-partition using V-MDAV is as follows:

1. Compute the distances between the records and store them in a distance matrix.
2. Compute the centroid c of the data set.
3. Whilst there are more than $k-1$ records not yet assigned to a group do:
    a) Let e be the most distant record to c.
    b) Form a group around e which contains the $k-1$ records closest to e.
    c) Extend the group, which consists of adding to the current group up to $k-1$ records using these steps:
        i. Find the unassigned record e_{min} which is closest to any of the records of the current group and let d_{in} be the distance between e_{min} and the group;
        ii. Let d_{out} be the shortest distance from e_{min} to the other unassigned records;
        iii. If $d_{in} < \gamma d_{out}$ then assign e_{min} to the current group. γ is a gain factor that has to be tuned according to the data set.
4. If less than k records remain unassigned, no new group can be formed. In that case, assign the remaining records to their closest group.

At the time of writing this chapter, MDAV is the most popular algorithm used to micro-aggregate multivariate data sets. Thus, we will use it as a reference to compare with our genetic approach.

3 Privacy Protection Through Genetic-Algorithm-Based Micro-aggregation

As we have shown in previous sections, the micro-aggregation problem is known to be NP-hard. Looking for optimal solutions in a multivariate data set is a very difficult task. Genetic algorithms are very well suited for searching solutions in complex search spaces. Therefore, a genetic-based technique seems to be a good candidate for solving this privacy problem.

We use the concept of GA described by Holland in [14]. A GA is a method for moving from one population of chromosomes to a new population by using a kind of natural selection together with the genetics-inspired operators of crossover, mutation, and inversion. Each chromosome consists of genes, each gene being an instance of a particular allele [22]. A GA depends on a number of parameters, namely population size, mutation rate, crossover rate and so on. The value of these parameters is typically initialised by following the recommendations of experts [5,10], although statistical techniques can be used to tune these parameters [4].

In a nutshell, a GA is a function optimiser that can be described in terms of a population of chromosomes, a coding sequence, a selection algorithm, a fitness function and some genetic operators. The chromosomes are candidate solutions to the problem which are encoded by means of the coding sequence. The selection algorithm determines which chromosomes will generate offspring. The fitness function evaluates how good a chromosome is and the genetic operators are used to mix the genetic information of the selected chromosomes in order to generate the offspring.

In order to address the micro-aggregation problem, a GA must be properly modified. The modifications introduced in our scheme with respect to a classical GA are intended to boost performance when micro-aggregating multivariate data. We next deal with those modifications in detail.

3.1 The Coding Sequence

The way in which candidate solutions are encoded is a key factor in the success of a GA [22].

Searching a proper coding could be a bewildering task due to the array of possible coding alternatives. However, at the same time, looking for a good coding among a plethora of alphabets is an invigorating challenge.

Codings can be mainly classified according to their alphabet in three main categories:

- Binary codings: They are based on a two-symbol alphabet (e.g. 0 and 1) and have been widely used because they were the codings initially introduced by Holland in his pioneering work [14]. Some recent examples are [19, 24, 30]. Moreover, lots of heuristics based on binary codings have been used as meta-heuristics in order to determine the value of parameters such as the crossover rate and the mutation rate [4, 12, 27].
- N-ary codings: They differ from binary codings in that each gene can take N different alleles[2]. As it will be shown next, this coding is our choice for addressing the multivariate micro-aggregation problem.
- Real-valued codings: These codings assign a real value to each gene. This kind of coding is basically used when the representation of a real number in a binary alphabet is awkward, this is, when the number's precision or range are not previously and clearly known [3, 21, 35].

In [7] a binary coding was used to solve the problem of **univariate** micro-aggregation. The idea was to sort univariate records in ascending order and to represent the i-th record by the i-th symbol in a binary string (chromosome). Thus, a chromosome represents a k-partition as follows. If a cluster starts at the i-th record, its bit in the chromosome is initialised with a "1" symbol; otherwise it is set to "0". This coding makes sense for univariate data sets because they can be sorted. Unfortunately, no generalisation to the multivariate case is possible because there is no way of sorting multivariate records

[2] It is clear that all problems that can be encoded using N-ary codings can be also encoded by using a binary alphabet which would encode the N-ary alphabet. However, some problems are more easily and naturally encoded by using an N-ary alphabet [17, 34].

without giving a higher priority to one of the attributes. To overcome this limitation, we propose a new coding that can be applied to data sets containing any number of dimensions.

Micro-aggregation requires that each group in a k-partition contain at least k records. Thus, the maximum number G of groups in a k-partition is

$$G = \left\lfloor \frac{n}{k} \right\rfloor, \tag{4}$$

where n is the total number of records to be micro-aggregated. The maximum number of groups G is the number of different alleles which must be defined in order to be able to encode a valid solution of the problem within each chromosome. Thus, the cardinality of our alphabet is G and each symbol of the alphabet represents one group/part of the k-partition.

We consider that each chromosome has a fixed number of genes equalling the number of records in the data set. This is, the value of the i-th gene in a chromosome defines the group of the k-partition which the i-th record in the data set belongs to.

Example 1. Taking the data in Table 1 and considering $k = 3$ we use (4) to get that $G = 3$. We then define a 3-character alphabet (e.g. A, B and C) that will be used to encode our chromosomes. Since we have 11 records in the data set, each chromosome is built up of 11 genes. A valid chromosome could be $AAABBBCCCAA$ which encodes the following 3-partition: group $A = \{1,2,3,10,11\}$, group $B = \{4,5,6\}$ and group $C = \{7,8,9\}$

Note that using alphabets with more than G symbols is conceivable, but the *principle of minimum alphabets* states that the smallest alphabet that permits a natural expression of the problem should be selected [11].

Table 1. Example: SME data

Company name	Surface (m^2)	Employees	Turnover (€)	Net profit (€)
A&A Ltd	790	55	32,12,334	3,13,250
B&B SpA	710	44	22,83,340	2,99,876
C&C Inc	730	32	19,89,233	2,00,213
D&D BV	810	17	9,84,983	1,43,211
E&E SL	950	3	1,94,232	51,233
F&F GmbH	510	25	1,19,332	20,333
G&G AG	400	45	30,12,444	5,01,233
H&H SA	330	50	42,33,312	7,77,882
I&I LLC	510	5	1,59,999	60,388
J&J Co	760	52	53,33,442	10,01,233
K&K Sarl	50	12	6,45,223	3,33,010

"Company name" is an identifier to be suppressed before publishing the data set

3.2 Initialising the Population

The initialisation of the population is not a key issue when all the possible combinations of the symbols in the alphabet are candidate solutions. Generally, the initialisation of the chromosomes in the population is performed at random, i.e. by using a pseudo-random generator mapped over the alphabet of the coding. Pseudo-random chromosome initialisation with n records and G different alphabet symbols results in

$$V'_{(G,n)} = G^n \tag{5}$$

possible chromosomes. Unfortunately, only a small fraction of such chromosomes meet the cardinality constraints to represent valid k-partitions.

Domingo-Ferrer and Mateo-Sanz [7] proved that

Result 1 *In an optimal k-partition, each group has between k and $2k-1$ records.*

From Result 1 the minimum number of groups in a k-partitions is

$$g = \left\lceil \frac{n}{2k-1} \right\rceil . \tag{6}$$

The probability of randomly obtaining a k-partition candidate to optimality (that is, meeting the cardinality constraints of Result 1) is the total number of candidate optimal k-partitions divided by the number of possible chromosomes given in (5). Computing the exact number of k-partitions candidate to optimality is not straightforward. However, in [29] an upper bound in the number of candidates is given by following the next steps.

1. First, the number $a_{n,k}$ of *representatives*[3] of optimal k-partitions given n and k is determined by using the next recursive definition for $a_{n,k}$

$$a_{n,p} = 0 \ \text{ if } n < p, k \le p \le 2k-1;$$
$$a_{n,p} = 1 \ \text{ if } p \le n \le 2k-1, k \le p \le 2k-1;$$

otherwise, if $n \ge 2k$

$$\begin{aligned} a_{n,p} = a_{n-p,p} + a_{n-(p+1),p+1} + a_{n-(p+2),p+2} + \dots \\ \dots + a_{n-(2k-1),2k-1}, k \le p \le 2k-1; \end{aligned} \tag{7}$$

where $a_{n,p}$ denotes the number of representatives with n elements with the first subset having a cardinality $\ge p$.

Table 2 lists the values of $a_{n,k}$ for several choices of k and n.

[3] A representative is a k-partition with its groups enumerated in ascending order of cardinality. Thus, the k-partition with three groups of cardinalities 3, 4 and 5, respectively, represents all k-partitions formed by three groups with the same cardinalities (i.e. $\{3,4,5\}$, $\{3,5,4\}$, $\{4,3,5\}$, $\{4,5,3\}$, $\{5,3,4\}$ and $\{5,4,3\}$).

Table 2. Values of $a_{n,k}$ for several choices of k and n

n	$k = 3$	$k = 4$	$k = 5$
10	2	2	1
20	6	6	5
30	11	14	14
40	18	27	29
50	26	45	56
100	94	272	520
500	2,134	26,477	1,97,549
1,000	8,434	2,05,037	29,53,389

2. The representative with the maximal number G of groups is taken and the number μ of different candidate optimal k-partitions it represents is obtained by using the following multinomial-inspired formula.

$$\mu = \frac{n!}{k_1!k_2!\dots k_G!}\frac{1}{G!}, \tag{8}$$

   where k_m is the cardinality of the m-th group in the k-partition. Clearly, μ is an upper bound for the number of candidate optimal k-partition represented by any representative.
3. Finally, multiplying μ by the number of representatives $a_{n,k}$ and dividing by the number G^n of random chromosomes, an upper bound on the probability P(random candidate optimal) of hitting a candidate optimal k-partition by random initialisation is obtained:

$$P(\text{random candidate optimal}) \leq \frac{a_{n,k}\mu}{G^n}. \tag{9}$$

Example 2. Considering a data set with $n = 30$ records/elements and cardinality constraint $k = 3$, the probability of obtaining a candidate to optimal chromosome at random is

$$P(\text{random candidate optimal}) \leq 13,2977212023656 \times 10^{-12} \approx 0$$

From Example 2, it becomes clear that random initialisation is not suitable to obtain candidate optimal k-partitions. The cardinality constraints stated in Result 1 must be embedded in the initialisation procedure. Thus, Algorithm 2 is used to initialise the chromosomes with solutions that are candidate to optimality.

The application of Algorithm 2 guarantees that each group/part has at most $2k - 1$ elements. However, groups with less than k elements can exist. In order to avoid this problem, elements from the most populated groups are randomly moved to the groups with less than k elements. At the end of this process, chromosomes representing k-partitions which are candidate to be optimal are obtained.

Algorithm 2: Chromosome initialisation with constraints

```
    Data: inout chromosome, in k.
    Result: $k$-partition.
1 gene_number = 0
2 while gene_number < chromosome_size do
3     group_number = rand() mod G
4     if records_in_group[group_number]$< 2k - 1$ then
5         chromosome[gene_number]←group_number
6         records_in_group[group_number] += 1
7         gene_number += 1
8     end
9 end
```

3.3 The Fitness Function

The evaluation of a chromosome in the population is the costliest part in the whole evolution process of the GA. We want to obtain a measure of the homogeneity of the groups in the k-partition represented by a given chromosome.

As we have introduced in Sect. 2, the most frequently used homogeneity measure for k-partitions is the within-group sum of squares *SSE* [7, 16]. The *SSE* is the sum of square distances from the centroid of each group to every element (i.e. record) in the group. For a k-partition, *SSE* can be computed as:

$$SSE = \sum_{i=1}^{s} \sum_{j=1}^{n_i} (\mathbf{x}_{ij} - \bar{\mathbf{x}}_i)'(\mathbf{x}_{ij} - \bar{\mathbf{x}}_i), \tag{10}$$

where s is the number of groups in the k-partition and n_i is the number of records in the i-th group. Expression (10) is applied to the data after standardising them, this is, after subtracting from the values of each attribute the attribute mean and dividing the result by the attribute standard deviation[4].

During the evolution process, some chromosomes can appear which are not candidate optimal k-partitions due to the effect of genetic operators. This is, some groups can appear whose cardinality is no longer in the range $[k, 2k-1]$. In these cases, we do not remove the chromosome from the population; instead, we penalise it in order to nearly preclude its reproduction.

Our goal is to minimise *SSE*. Thus, the fitness value that we finally assign to a chromosome is

$$Fitness = \frac{1}{SSE + 1}. \tag{11}$$

[4] Standardisation does not affect attribute correlations.

3.4 Selection Scheme and Genetic Operators

Our GA uses the well-known roulette-wheel selection algorithm. This selection algorithm is based on weighting the chromosomes of the population according to their fitness. Thus, the probability of a chromosome for being selected to pass to the next generation is proportional to its fitness. This method tends to converge quickly because it does not preserve the diversity of the population. Even though this entails some risk of missing good solutions, experimental results show that this risk is small and it can be afforded for the sake of speed.

We have used the most common genetic operators, this is, one-point crossover and mutation.

4 A Hybrid Approach

In the previous section, we have introduced how to build a GA to let it properly face the problem of micro-aggregating a multivariate data set. Although the proposed method is correct, it suffers from some performance limitations when the number of elements in the data set is large.

On the one hand, the GA approach is very efficient in terms of SSE, but it cannot deal with large data sets. On the other, distance-based methods like MDAV can deal with very large data sets, but they are not as good as the GA in terms of SSE.

The solution consists in taking the advantages of both methods and mix them to obtain a new hybrid method able to micro-aggregate large data sets, whilst maintaining a good performance in terms of SSE.

The hybrid method, that we call two-step partitioning [20], can be summarised as follows:

1. Let k be a small value (e.g. $k = 3$).
2. Let K be larger than k and divisible by k, small enough to be suitable for our GA (e.g. $K = 21$).
3. Use a fixed-size heuristic to build a k-partition.
4. Taking as input records the average vectors obtained in the previous step, apply the fixed-size heuristic to build *macro-groups* (i.e. sets of average vectors) of size K/k.
5. For each given *macro-group*, replace the average vectors by the k original records to obtain a K-partition.

Figure 2 shows two macro-groups obtained by the two-step partitioning method using MDAV two times with $K = 24$ and $k = 3$.

Finally, apply the GA to each macro-group in the K-partition in order to generate an optimal or near optimal k-partition of the macro-group. The composition of the k-partitions of all macro-groups yields a k-partition for the entire data set. An important issue when using the GA on a macro-group is to include the fixed-size k-partition of the macro-group induced by the

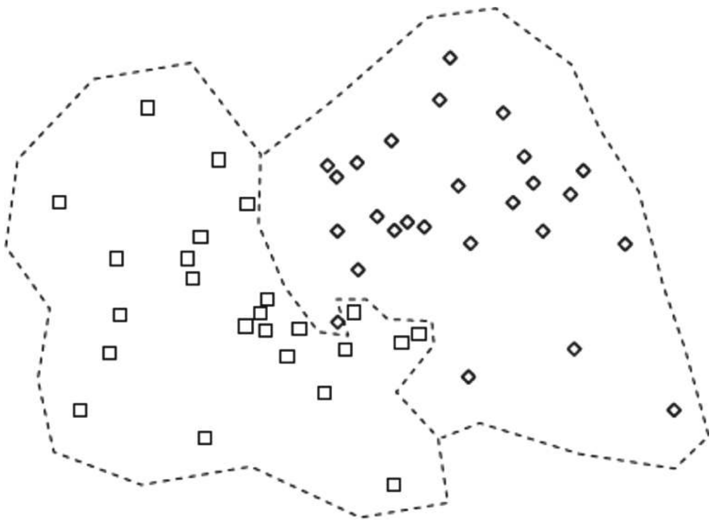

Fig. 2. Two macro-groups obtained by the two-step partitioning method using MDAV two times with $K = 24$ and $k = 3$

k-partition of the entire data set computed in Step 3 above as one of the chromosomes of the initial population fed to the GA.

The following result holds:

Lemma 1. *The SSE of the k-partition obtained using a two-step partition based on a fixed-size heuristic followed by a GA is not greater than the SSE of the k-partition output by the fixed-size heuristic alone.*

Proof. By assumption, the fixed-size k-partition of each macro-group induced by the fixed-size k-partition of the data set obtained at Step 3 is one of the chromosomes of the initial population fed to the GA. Therefore, for each macro-group, the k-partition output by the GA is at least as good as the fixed-size k-partition in terms of the fitness function. This implies that the *SSE* of the k-partition of the entire data set obtained by composition of the macro-group k-partitions output by the GA is not greater than the *SSE* of the k-partition of Step 3. □

5 Experimental Results

In this section we summarise the most relevant experimental results obtained with the proposed GA-based micro-aggregation algorithm and the hybrid micro-aggregation approach. We compare these results with the ones obtained with the MDAV algorithm and by exhaustive search (ES). Note that ES is only possible with tiny data sets of up to 11 elements.

We classify the experiments in three main groups:

1. Experiments with the example data set: These experiments have been carried out in order to clearly show the main properties of the GA approach in a tiny data set. The obtained results are not statistically relevant, but they provide a first intuition of what the GA approach is able to do.
2. Experiments with small data sets: This set of experiments are intended for a complete analysis of the behaviour of the GA-based micro-aggregation.
3. Experiments with large and real data sets: In these case, the hybrid approach is on the line. Well-known data sets are used to demonstrate that the hybrid approach performs properly.

Each experiment consists of several runs of the GA. At each run of the GA, a different combination of parameters is used i.e. the Mutation rate (M_r) has been varied from 0 to 1, with a step of 0.1, the Crossover rate (C_r) has been varied from 0 to 1, with a step of 0.1 and, the population size has been varied from 10 to 100 chromosomes with a step of 10.

The above implies 1210 different parameter settings. For each setting, the GA was run 10 times, so that altogether each test comprised 12,100 runs of the GA[5].

5.1 Results for the Running Example

With the aim of giving a first intuition of how our GA-based micro-aggregation method performs, we have used it to micro-aggregate the data set in Table 1.

Table 3 gives a comparison between ES, MDAV and GA in terms of running time and *SSE* for the example data set in Table 1. The running time of the GA depends on the number of generations. Figure 3 shows a histogram of the convergence generation of the tests. It can be observed that most of the tests converge in less than 5,000 iterations. Thus, the running time for the GA in Table 3 corresponds to 5,000 iterations.

Even though MDAV is faster than the GA approach, the SSE obtained with the GA is better. In fact, in 91% of the 12,100 runs, the GA reaches the optimal $SSE = 14.82$.

Table 3. Performance of ES, MDAV and GA on the running example

$n = 11$, $d = 4$	ES	MDAV	GA
Time	717 s	<1 s	1.4 s
SSE	14.82	18.29	14.82

[5] Tests were performed on a Pentium 4 PC running at 3 GHz with a Debian Linux operating system. Programs were coded in C++.

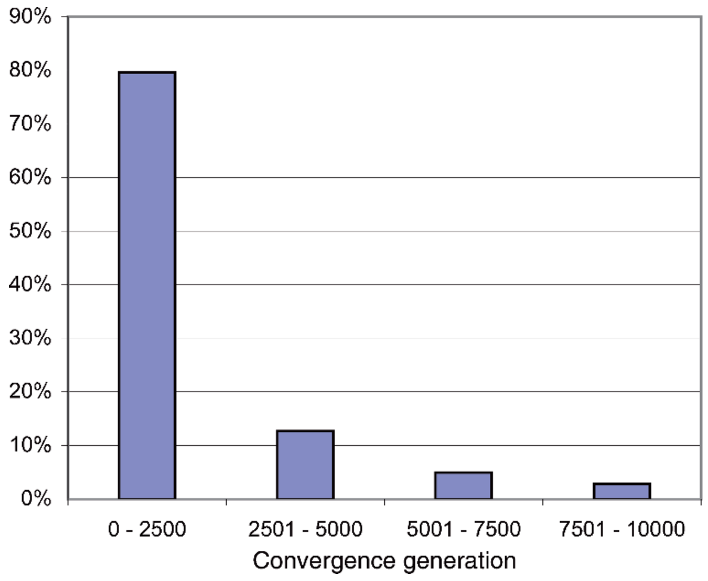

Fig. 3. Histogram of the convergence generation of the tests on the example data set

Table 4. Minimum SSE values for tests with larger data sets

Data set	MDAV	GA$_m$	M_r	C_r	pop.size
20×2	5.39	4.29	0.1	0.4	60
35×2	5.01	3.13	0.1	0.1	90
50×2	5.92	4.02	0.1	0.1	50
100×2	5.48	15.37	0.1	0.2	50

5.2 Results in Small Data Sets

After observing the promising results achieved by the GA approach over the data set in Table 1, we have carried out a deeper analysis of its performance on a variety of small data sets.

Random data sets $n \times d$ have been constructed for several numbers n of records and several numbers of attributes d. Attribute values were drawn from the $[-10000, 10000]$ interval by simple random sampling. For data sets with $n = 11$, $n = 20$ and $n = 35$ the number of generations (number of iterations) was 10^4. For the data set with $n = 50$, the number of generations was 10^5. Note that for micro-data sets with $n = 11$ records, the optimal k-partition could be found by ES in a reasonable time. For $n > 11$ the *SSE* obtained with GA was only compared to the *SSE* obtained using MDAV.

Table 4 shows the best results obtained with the GA over data sets of fixed dimensionality $d = 2$ and a variable number of elements (i.e. 20, 35, 50 and 100). From these results it becomes clear that the mutation rate should

Table 5. Average of the minimum SSE values for tests with larger data sets

Data set	MDAV	GA$_A$	M_r	C_r
20×2	5.39	4.61	0.1	0.7
35×2	5.01	3.59	0.1	0.5
50×2	5.92	4.67	0.1	0.5
100×2	5.48	23.78	0.1	0.0

Table 6. *SSE* for data sets with $n = 11$ and different d values

Data set	SE	MDAV	GA	Av.Conv.E.
11×5	29.36	39.11	29.36	931
11×7	43.28	52.47	43.28	839
11×10	68.66	71.14	68.66	705

be low[6] (e.g. 0.1). Moreover, it is also clear that the GA improves the MDAV heuristic when the number of elements is lower than 100. This result must be emphasised because it represents the main limitation of the proposed method (i.e. it cannot deal with large data sets).

Table 5 shows the averages of the minimum values obtained in the same data sets and confirms the previous conclusions.

In both the 11×2 and 11×3 data sets the optimal *SSE* was obtained. In order to check whether the optimal *SSE* is obtained for larger values of d, we have tested our GA over random data sets with $d = 5$, $d = 7$ and $d = 10$. In all cases, the k-partition generated for $M_r \neq 0$ and $C_r \neq 0$ equals the optimal k-partition output by ES. Table 6 shows the results of these tests. In all three data sets, around a 90% of the tests output the optimal *SSE* value.

In [29] a detailed sensitivity analysis of the parameters of the GA based on the Analysis Of Variance (ANOVA) and the Tukey and Scheffé test is given.

5.3 Results in Real and Large Data Sets

The main limitation of our GA-based micro-aggregation technique is that it cannot deal with large data sets. Thus, the hybrid technique is needed.

In order to test the hybrid technique we have used the next data sets:

- Synthetic data sets :"Scattered" and "Clustered" are data sets that contain $n = 1{,}000$ records with $d = 2$ attributes. The former is *scattered* in the sense that no natural clusters are apparent. Attribute values were drawn from the $[-10000, 10000]$ range by simple random sampling. The second data set is considered to be *clustered* because its records are grouped in natural clusters. The attribute values were uniformly drawn from $[-10000, 10000]$ as in the previous case, but records were then group-wise shifted by a random amount in order to form clusters of size between

[6] The use of higher values of the mutation rate leads to the generation of too much non-optimal chromosomes.

Table 7. Summary of experimental results with the hybrid approach ($k = 3$)

			Two-step MDAV partitioning		
Data set	K	SSE_{MDAV}	# clus.	SSE	% impr.(%)
"Scattered"	12	4.71	83	4.28	9
	18	4.71	56	4.33	8
	27	4.71	38	4.58	3
"Clustered"	12	3.57	84	2.84	20
	18	3.57	56	2.14	40
	27	3.57	41	2.09	41
"Census"	12	799	91	768	4
	18	799	61	767	4
	27	799	41	789	1
"EIA"	12	217	342	189	13
	18	217	228	186	14
	27	217	152	197	9

3 and 5 records each; more details on the generation of this data sets can be found in [6].

- Real data sets: "Census" is a data set that contains 1,080 records with 13 numerical attributes. "EIA" contains 4092 records with 11 numerical attributes. These data sets were proposed as reference micro-data sets during the CASC project [2] and have been widely used [7–9, 18].

Table 7 provides a summary of the obtained results. It can be observed that the hybrid method always improves MDAV. The main improvement takes place when the data set is clustered. This is due to the fact that MDAV is a fixed-size heuristic i.e. it generates groups of k elements. This lack of flexibility results in a poor performance when the elements in the micro-aggregated data set are clustered in groups of variable size. On the contrary, the hybrid approach makes use of the flexibility of the GA and clearly improves the SSE results (cf. [20] for details).

6 Conclusions and Future Challenges

In this chapter we have shown some of the privacy problems that the Information Society has to face. Some of these privacy threads can be tackled by using well-known techniques from the field of statistical disclosure control such as micro-aggregation.

We have introduced the main concepts of micro-aggregation and we have shown that, due to the complexity of the problem, no optimal methods exist and novel heuristics must be developed.

We have pointed out the possibility of using a GA to solve the micro-aggregation problem and we have shown how to modify a classical GA in order to let it cope with the micro-aggregation problem. Moreover, we have reported the main limitation of the GA approach (i.e. it can only deal with small data sets) and we have proposed an improvement that consists in the mixture of our GA and a classical distance-based heuristic like MDAV.

The reported experimental results demonstrate the usefulness of the proposed methods and open the door to an invigorating research line. Lots of questions remain open, next we conclude with some future challenges:

- Look for better codings.
- Test the efficiency of other selection algorithms.
- Evaluate the importance of genetic operators such as multiple-point crossover or inversion.

Disclaimer and Acknowledgements

The author is solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organisation. This work was partly supported by the Spanish Ministry of Education through projects TSI2007-65406-C03-01 “E-AEGIS” and CONSOLIDER CSD2007-00004“ARES”, and by the Government of Catalonia under grant 2005 SGR 00446.

References

1. C. Boyens, R. Krishnan, and R. Padman. On privacy-preserving access to distributed heterogeneous healthcare information. In IEEE Computer Society, (ed.), *Proceedings of the 37th Hawaii International Conference on System Sciences HICSS-37*, Big Island, HI, 2004.
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, 2002. European Project IST-2000-25069 CASC, http://neon.vb.cbs.nl/casc.
3. W. Chien and C.-C. Chiu. Using nu-ssga to reduce the searching time in inverse problem of a buried metallic object. *IEEE Transactions on Antennas and Propagation*, 53(10):3128–3134, 2005.
4. A. Czarn, C. MacNish, K. Vijayan, B. Turlach, and R. Gupta. Statistical exploratory analysis of genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 8(4):405–421, 2004.
5. K. A. DeJong. *Analysis of the Behaviour of a Class of Genetic Adaptive Systems.* PhD thesis, Dept. Comput. Commun. Sci., University of Michigan, 1975.
6. J. Domingo-Ferrer, A. Martínez-Ballesté, J. M. Mateo-Sanz, and F. Sebé. Efficient multivariate data-oriented microaggregation. *The VLDB Journal*, 15(4):355–369, 2006.
7. J. Domingo-Ferrer and J. M. Mateo-Sanz. Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and Data Engineering*, 14(1):189–201, 2002.

8. J. Domingo-Ferrer, F. Sebe, and A. Solanas. A polynomial-time approximation to optimal multivariate microaggregation. *Computers & Mathematics with Applications, In Press*, Corrected Proof, Available online 19 July 2007. (http://www.sciencedirect.com/science/article/B6TYJ-4P77G9K-1/2/65a8e45541c49e7862a14870253124ba)
9. J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogenerous k-anonymity through microaggregation. *Data Mining and Knowledge Discovery*, 11(2):195–212, 2005.
10. B. Freisleben and M. Härtfelder. Optimization of genetic algorithms by genetic algorithms. In *In Proc. Int. Conf. Artificial Neural Networks and Genetic Algorithms*, pages 392–399, Vienna, 1993.
11. D. E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley Publishing, 1989.
12. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1):122–128, 1986.
13. HIPAA. Health insurance portability and accountability act, 2004. `http://www.hhs.gov/ocr/hipaa/`.
14. J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975. Second edition: MIT, 1992.
15. A. Hundepool, A. Van de Wetering, R. Ramaswamy, L. Franconi, A. Capobianchi, P.-P. DeWolf, J. Domingo-Ferrer, V. Torra, R. Brand, and S. Giessing. *μ-ARGUS version 4.0 Software and User's Manual*. Statistics Netherlands, Voorburg NL, may 2005. http://neon.vb.cbs.nl/casc.
16. Y. Jung, H. Park, D.-Z. Du, and B. L. Drake. A decision criterion for the optimal number of clusters in hierarchical clustering. *Journal of Global Optimization*, 25:91–111, 2005.
17. K. Knodler, J. Poland, A. Mitterer, and A. Zell. Optimizing data measurements at test beds using multi-step genetic algorithms. In N. Mastorakis, (ed.), *Advances In Fuzzy Systems and Evolutionary Computation (Proceedings of WSES EC 2001)*, pages 277–282, 2001.
18. M. Laszlo and S. Mukherjee. Minimum spanning tree partitioning algorithm for microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, 17(7):902–911, 2005.
19. W. C. Liu. Design of a cpw-fed notched planar monopole antenna for multiband operations using a genetic algorithm. *IEE Proceedings on Microwaves, Antennas and Propagation*, 152(4):273–277, 2005.
20. A. Martinez-Balleste, A. Solanas, J. Domingo-Ferrer, and J. M. Mateo-Sanz. A genetic approach to multivariate microaggregation for database privacy. In *23rd Internationl Conference on Data Engineering. Workshop on Privacy Data Management*, pages 180–185. IEEE Computer Society, April 2007.
21. G. M. Megson and I. M. Bland. Synthesis of a systolic array genetic algorithm. In *IPPS/SPDP*, pages 316–320, 1998.
22. M. Mitchell. *An Introduction to Genetic Algorithms*. MIT, 1999. Fifth printing.
23. A. Oganian and J. Domingo-Ferrer. On the complexity of optimal microaggregation for statistical disclosure control. *Statistical Journal of the United Nations Economic Comission for Europe*, 18(4):345–354, 2001.
24. S. J. Ovaska, T. Bose, and O. Vainio. Genetic algorithm-assisted design of adaptive predictive filters for 50/60 hz power systems instrumentation. *IEEE Transactions on Instrumentation and Measurement*, 54(5):2041–2048, 2005.

25. European Parliament. DIRECTIVE 2002/58/EC of the European Parliament and Council of 12 july 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications), 2002. `http://europa.eu.int/eur-lex/pri/en/oj/dat/2002/l_201/l_20120020731en00370047.pdf`.
26. Canadian Privacy. Canadian privacy regulations, 2005. http://www.media-awareness.ca/english/issues/privacy/canadian_legislation_privacy.cfm.
27. J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proceedings of the Third International Conference on Genetic algorithms*, pages 51–60, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.
28. A. Solanas and A. Martínez-Ballesté. V-MDAV: Variable group size multivariate microaggregation. In *COMPSTAT'2006*, pages 917–925, Rome, 2006.
29. A. Solanas, A. Martínez-Ballesté, J. M. Mateo-Sanz, and J. Domingo-Ferrer. Multivariate microaggregation based on genetic algorithms. In *3rd IEEE Conference On Intelligent Systems (IEEE IS'2006)*, pages 65–70, Westminster, 2006. IEEE Computer Society Press.
30. A. Solanas, E. Romero, S. Gómez, J. M. Sopena, R. Alquezar, and J. Domingo-Ferrer. Feature selection and oulier detection with genetic algorithms and neural networks. In P. Radeva B. López, J. Meléndez and J. Vitrià, (eds.), *Artificial Intelligence Research and Developement*, pages 41–48, Amsterdam, NL, 2005. IOS Press.
31. European Union. The Seventh Framework Programme (FP7), 2006. http://ec.europa.eu/research/fp7/understanding/index.html.
32. USPrivacy. U.S. privacy regulations, 2005. `http://www.media-awareness.ca/english/issues/privacy/us_legislation_privacy.cfm`.
33. L. Willenborg and T. DeWaal. *Elements of Statistical Disclosure Control*. Springer-Verlag, New York, 2001.
34. A. S. Wu and I. I. Garibay. The proportional genetic algorithm: Gene expression in a genetic algorithm. *Genetic Programming and Evolvable Machines*, 3(2):157–192, 2002.
35. Y. K. Yu, K. H. Wong, and M. M. Y. Chang. Pose estimation for augmented reality applications using genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 35(6):1295–1301, 2005.