Protección de la Privacidad mediante Microagregación Multivariante basada en Algoritmos Genéticos: Selección por Ruleta vs. Selección Uniforme

U. González-Nicolas y A. Solanas, *Member, IEEE*

***Resumen*—La microagregación es una técnica de control de la revelación estadística que tiene por objetivo proteger la privacidad de los usuarios, cuya información es almacenada en bases de datos. Se ha demostrado que microagregar de forma óptima un conjunto de datos multivariante es un problema NP, por ello en la actualidad se utilizan métodos heurísticos.**

En este artículo, presentamos un sistema de protección de la privacidad individual mediante microagregación basada en algoritmos genéticos. Detallamos el sistema de codificación, selección y reproducción. Además, comparamos dos métodos de selección alternativos: el método de selección por ruleta y el método de selección uniforme de *fitness*.

Los resultados experimentales demuestran la utilidad del método propuesto. Además constatamos que el método de selección uniforme de *fitness* produce mejores resultados en términos de pérdida de información que el método de selección por ruleta.

***Palabras clave*— Privacidad, Control de la revelación estadística, Microagregación, Algoritmos Evolutivos.**

I. Introducción

LA PROTECCIÓN de la privacidad es una tarea fundamental en una sociedad en la que es cada día más sencillo recoger y almacenar grandes cantidades de datos de carácter personal. Incluso la Declaración Universal de los Derechos humanos recoge en el artículo 12 la importancia de la privacidad[1].

Las nuevas tecnologías de la información y las comunicaciones (TIC) facilitan enormemente la recogida de información procedente de una gran variedad de canales entre los que destacan los siguientes:

- Comercio Electrónico: Gracias a Internet, el comercio electrónico se ha convertido en un sector económico esencial. El número de transacciones económicas realizadas en Internet crece cada año. Esta gran actividad es monitorizada y una inmensa cantidad de información es almacenada en bases de datos. Alguna de esta información es privada, y su incorrecta manipulación podría conducir a la revelación indebida de datos confidenciales. Por ello es necesario cumplir con una serie de normas que regulan el correcto funcionamiento de este tipo de transacciones [1].
- Medicina: En los últimos años se está generalizando el uso de herramientas de monitorización de enfermos crónicos y ancianos fuera de los hospitales. Además, historiales, radiografías, resultados de pruebas, etc, son almacenados en formato digital. Esta información altamente sensible debe ser tratada con la máxima privacidad, por lo que la cesión de este tipo de información médica está sujeta a regulaciones muy estrictas [2] [3].
- Servicios basados en localización: El gran uso de los teléfonos móviles y dispositivos inalámbricos ha propiciado la aparición de los llamados servicios basados en localización. Estos servicios responden a la necesidad de los usuarios de obtener información relativa a su actual posición (*e.g.* la farmacia más próxima, la ruta más rápida, etc.). La privacidad de los usuarios de este tipo de servicios merece una atención especial ya que si los datos no son tratados de forma correcta, los hábitos de consumo de los usuarios pueden quedar al descubierto [4] [5].
- Agencias estadísticas: Son muchos los datos personales que recogen las agencias estadísticas. Uno de los objetivos fundamentales de estas agencias es conjugar el derecho de la sociedad a la información y el derecho de los individuos a la privacidad. Es necesario disponer de los métodos que nos permitan hacer pública la información sin por ello poner en riesgo la privacidad de los usuarios a los que pertenecen los datos [6] [7] [8].

Aunque existe una clara convicción internacional de que la privacidad debe ser protegida, podemos encontrar múltiples ejemplos en los que esta privacidad ha sido vulnerada, ya sea por negligencia o incompetencia.

El 20 de Noviembre de 2007, se comunicó al Parlamento británico que dos unidades de disco con datos personales de más de 25 millones de británicos habían sido perdidos. El paradero de los discos es desconocido y la privacidad de los individuos, cuyos datos fueron perdidos, está en riesgo. Éste es tan solo el último caso de una larga lista de despropósitos. En Octubre de 2007 el "Her Majesty's Revenue and Customs"perdió otro disco que contenía información sobre las

Úrsula González-Nicolas y Agusti Solanas son miembros de la Cátedra UNESCO en Privacidad de Datos y del grupo de investigación CRISES del Departamento de Ingeniería Informática y Matemáticas de la Universidad Rovira i Virgili, 43007 Tarragona, Cataluña, España e-mail:{ursula.gonzaleznicolas,agusti.solanas}@urv.cat

[1]"Nadie será objeto de injerencias arbitrarias en su vida privada, su familia, su domicilio o su correspondencia, ni de ataques a su honra o a su reputación. Toda persona tiene derecho a la protección de la ley contra tales injerencias o ataques."

pensiones de 15.000 británicos. La privacidad de más de 26 millones de Americanos fue puesta en peligro cuando un disco fue robado de la casa de un empleado del "Department of Veterans Affairs.[en] 2006. En Agosto del mismo año, 658.000 consultas fueron reveladas por el buscador de American On-Line (AOL) y nuevamente el derecho a la privacidad se vio vulnerado. Estos son solo unos cuantos ejemplos que ilustran lo fácil que es poner en peligro la privacidad en nuestra sociedad.

En este artículo nos centramos en proteger la privacidad de los individuos, cuya información es almacenada en bases de datos. Concretamente proponemos un nuevo método de microagregación multivariante basado en algoritmos genéticos. Estudiamos dos métodos de selección: el sistema de selección por ruleta y el sistema de selección uniforme. Los resultados experimentales demuestran que el método propuesto es útil para microagregar pequeños conjuntos de datos. Además, demostramos que el método de selección uniforme es, por lo general, mejor que el método de ruleta.

El resto del artículo está organizado de la siguiente manera. En la Sección II explicamos algunos conceptos básicos de microagregación. En la Sección III detallamos nuestro método de microagregación mediante algoritmos genéticos. La Sección IV muestra los resultados experimentales y la comparativa entre los dos métodos de selección estudiados. Finalmente, el artículo concluye en la Sección V.

II. Conceptos previos sobre Microagregación

La microagregación es una técnica de protección de la privacidad en microdatos inicialmente propuesta en el marco del control de la revelación estadística[2]. Fijado un parámetro de seguridad k, la microagregación de un conjunto de datos numérico se lleva a cabo en dos etapas:

1. Agrupación: Los registros contenidos en el conjunto original se agrupan en subconjuntos de cardinalidad por lo menos k mediante algún criterio de similitud (*e.g.* la distancia Euclídea). El resultado de este proceso es una k-partición[3].
2. Substitución: Cada registro del conjunto original es substituido por el registro medio del subconjunto al cual ha sido asignado en la etapa anterior.

Tras microagregar un conjunto de datos, todo registro tiene, por lo menos, $k-1$ registros idénticos. Así pues, identificar unívocamente a un usuario no es posible. La Figura 1 muestra un ejemplo de microagregación univariante. Nótese, que los elementos del conjunto original han sido agrupados en función de su similitud. De este modo, cuando los elementos originales son substituidos por el valor medio del grupo al que pertenecen la pérdida de información es menor.

La pérdida de información se calcula mediante la suma del error cuadrático. Nosotros hemos usado la notación inglesa *Sum of Square Errors* (SSE) para referirnos a esta medida de pérdida de información detallada en la Ecuación 1,

[2]Statistical Disclosure Control

[3]Una k-partición es una partición en la que cada una de las partes tiene por lo menos k elementos.

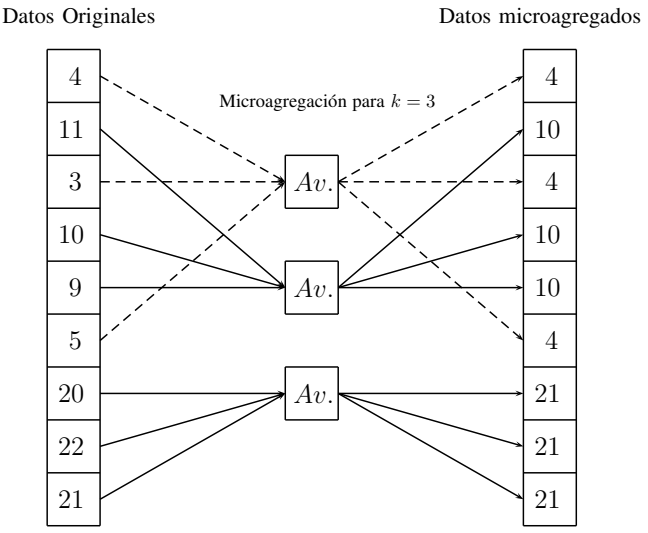


Figura 1. Ejemplo de microagregación univariante para $k = 3$

$$SSE = \sum_{i=1}^{s} \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_i)'(x_{ij} - \bar{x}_i) \quad (1)$$

donde s es el número de subconjuntos generados, n_i es el número de elementos del subconjunto i, x_{ij} es el elemento j-ésimo del subconjunto i y $\bar{x_i}$ es el valor medio del subconjunto i.

La microagregación óptima es un problema NP [9] cuando se aplica a conjuntos de datos multivariantes, por ello, se usan métodos heurísticos para resolverlo. Existe una gran cantidad de estos métodos para abordar el problema de la microagregación multivariante [10]. Algunos de ellos están basados en la generación de estructuras en árbol que conectan los elementos del conjunto de datos y su posterior división. Ejemplos de este tipo son el método de partición basado en el árbol de expansión mínima [11] y la recientemente propuesta μ-aproximacion [12]. La principal limitación del método basado en árboles de expansión mínima reside en el propio árbol. Aunque puede ser muy útil cuando los datos originales están distribuidos en clusters, el algoritmo no se adapta correctamente a ellos cuando se distribuyen de forma dispersa.

En lugar de estructurar los datos en árboles, es común el uso de algoritmos ansiosos (i.e. de optimización local), que tratan de agrupar los elementos por proximidad, intentando minimizar a cada paso el SSE. Algunos ejemplos de este tipo son el algoritmo de máxima distancia (MD) [13] y el algoritmo de distancia máxima al vector promedio (Maximum Distance to Average Vector (MDAV)) [14], [15]. Una pequeña modificación sobre esta misma línea fue descrita en [11] bajo el nombre de Microagregación de tamaño fijo basada en centroide. Por otro lado, en [16] Solanas et al. propusieron una mejora sobre MDAV llamada Variable-MDAV. La principal ventaja de estos métodos es su simplicidad conceptual, sin embargo, su coste computacional asociado es equivalente al de los métodos basados en árboles y en algunos casos incluso superior.

III. Microagregación basada en Algoritmos Genéticos

Los algoritmos genéticos (AG) son métodos de optimización que pretenden imitar el proceso evolutivo de las especies animales descrito por C. Darwin [17]. La idea consiste en determinar los individuos mejor adaptados al entorno y seleccionarlos para su reproducción. Idealmente, estos individuos mejor adaptados generarán descendencia mejor adaptada al entorno y por consiguiente tendrán más posibilidades de sobrevivir.

En un AG, una población de n cromosomas (i.e. la representación de un individuo) evoluciona por medio del intercambio de material genético entre ellos. Resolver problemas mediante AG implica definir la codificación de los individuos (i.e. el ADN), la función a optimizar (i.e. el entorno), la función de evaluación (i.e. su adaptación al entorno) y los métodos de intercambio genético (i.e. la reproducción). En [13] se propuso el uso de AG para la resolución del problema de la microagregación univariante. Sin embargo, la codificación binaria propuesta no permitía extender la solución al problema de la microagregación multivariante. En [18] se propuso por primera vez la utilización de AG para tratar el problema de la microagregación multivariante. Basándonos en este último artículo, proponemos diversas modificaciones del esquema clásico de los AG. Las modificaciones introducidas en el modelo están pensadas para mejorar el rendimiento (i.e. reducir el SSE) de la microagregación multivariante de datos. A continuación describimos los puntos fundamentales de nuestro método:

III-A. Codificación

La codificación del problema es fundamental para la correcta resolución del mismo. Cada cromosoma representa una posible solución. Aunque existen diversas alternativas para codificar la información, usualmente se suelen clasificar en tres categorías según el alfabeto utilizado:

- Codificación binaria: introducida originalmente por Holland [19] es la codificación más habitual. Los cromosomas son secuencias de ceros y unos, que toman un significado u otro en función del problema a resolver.
- Codificación n-aria: permite codificar las soluciones como cadenas de enteros, donde cada entero representa algún aspecto particular de la solución. Todos los problemas que se pueden codificar mediante una codificación n-aria también pueden representarse mediante una codificación binaria que codifique el alfabeto n-ario. De todas formas la utilización de un alfabeto n-ario permite codificar el problema de una forma más simple y natural. Veremos que este es el método utilizado en microagregación multivariante.
- Representación directa: este tipo de codificación se usa en problemas que utilizan valores difíciles de representar, como por ejemplo números reales. Se trata de problemas donde la codificación binaria sería demasiado complicada. En este caso, el cromosoma es visto como una secuencia de valores, y se entiende como una unidad.

La codificación binaria se puede utilizar, por ejemplo, para codificar las soluciones de un proceso de microagregación univariante. En este tipo de microagregación los registros se ordenan de forma ascendente o descendente de acuerdo con un criterio determinado. Una vez ordenados los registros, la idea de esta codificación es representar el registro i-ésimo por el i-ésimo símbolo del cromosoma. Una posible forma de representar una k-partición es utilizar el símbolo "1" en el gen del cromosoma que representara el registro a partir del cual comienza el grupo, y utilizar el símbolo "0" para el resto. Esta solución no la podemos aplicar en el caso de la microagregación multivariante, ya que no existe ninguna forma de ordenar datos multidimensionales sin dar mayor prioridad a un atributo en concreto.

Dado que la cardinalidad mínima de los subconjuntos de la k-partición es k, el máximo número de subconjuntos *G* que podemos obtener es

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

donde n es el número total de registros a microagregar. *G* se puede interpretar como el máximo número de símbolos necesarios para codificar las soluciones del problema, ya que cada símbolo puede ser usado para representar un subconjunto de la k-partición. Por tanto la cardinalidad de nuestro alfabeto es *G*.

Cuando codificamos un cromosoma consideraremos un número fijo de genes igual al número de registros a microagregar (n), con lo cual cada gen representa un registro. Así, el valor del i-ésimo gen del cromosoma indica el grupo al que pertenece el registro *i*.

III-B. Inicialización de la población

La generación de la población inicial se acostumbra a hacer de modo pseudo-aleatorio, así cada gen puede contener alguno de los posibles valores del alfabeto con una probabilidad uniforme. Conviene que la población inicial sea lo más variada posible y contenga individuos muy diferentes para así poder explorar el espacio de búsqueda de forma más extensa. Por otro lado, se debe intentar que el número de cadenas que representen individuos no factibles (i.e. que representen una solución que no cumpla las restricciones del problema) sea el más pequeño posible.

Si planteamos un problema de microagregación con *n* registros tal que nuestro alfabeto esté compuesto de *G* valores diferentes, el número de posibles cromosomas que podemos obtener es G^n. En nuestro problema sólo una pequeña parte representaran soluciones válidas (i.e. aquellas en que la cardinalidad de los grupos esté en el rango [*k*, 2*k*-1], cf. [13]). Por tanto una inicialización pseudo-aleatoria pura no nos sirve.

El Algoritmo I es el algoritmo de inicialización que proponemos para generar cromosomas que representen soluciones candidatas a ser óptimas. Después de aplicar-lo, los cromosomas codificaran k-particiones con grupos de cardinal 2*k*-1 a lo sumo. Sin embargo, las k-particiones pueden contener grupos de cardinalidad inferior a *k*. Por eso, después de aplicar el algoritmo de inicialización se deben recomponer los grupos, moviendo aleatoriamente registros de grupos con cardinalidad mayor a *k* a grupos con cardinalidad menor a *k*.

ALGORITMO I

INICIALIZACIÓN DE UN CROMOSOMA CONSIDERANDO LAS RESTRICCIONES IMPUESTAS POR LA MICROAGREGACIÓN.

```
01) function Inicializar_Cromosoma (inout cromosoma, in k) is
02)     integer reg_por_grupo[numero_genes]
03)     integer num_gen
04)     integer num_grupo
05)     begin
06)       num_gen = 0
07)       while num_gen < tamaño_cromosoma do
08)         num_grupo = rand() mod G
09)         if reg_por_grupo[num_grupo]< 2k − 1 then
10)           cromosoma[num_gen]←num_grupo
11)           reg_por_grupo[num_grupo] += 1
12)           num_gen += 1
13)         end if
14)       end while
15) end function
```

ALGORITMO II

REASIGNACIÓN DE LOS GRUPOS EN UNA K-PARTICIÓN COMPUESTA POR ALGÚN GRUPO DE CARDINALIDAD MENOR A k.

```
01) function reasignar (inout cromosoma, inout reg_por_grupo) is
02)     integer pendientes, g, gen
03)     begin
04)       pendientes = deshacer_grupos(cromosoma, reg_por_grupo)
05)       while pendientes > 0 do
06)         if ∃ grupo con cardinalidad < k then
07)          selección aleatoria de un grupo 'g' con cardinalidad < k
08)           reg_por_grupo[g] += 1
09)           gen = obtener_gen_libre(cromosoma)
10)           cromosoma[gen] = g
11)           pendientes -=1
12)         else if pendientes >= k and ∃ grupo con card. = 0 then
13)          selección aleatoria de un grupo 'g' con cardinalidad = 0
14)           for all 'i' such that 0 ≤ i ≤ k do
15)             gen = obtener_gen_libre(cromosoma)
16)             cromosoma[gen] = i
17)           end for
18)           reg_por_grupo[i] += k
19)           pendientes -=k
20)         else
21)           selección aleatoria de un grupo 'g' con card. < 2k-1
22)           gen = obtener_gen_libre(cromosoma)
23)           cromosoma[gen] = g
24)           pendientes -=1
25)         end if
26)       end while
27) end function
```

En el Algoritmo II presentamos el esquema propuesto para redistribuir los elementos hacia conjuntos de cardinalidad inferior a k. La función *deshacer_grupos* consiste en eliminar los grupos con cardinalidad inferior a *k*, devolviendo el número de registros a reasignar. De este modo se eliminan los valores de aquellos genes del cromosoma que representan a grupos con cardinalidad inferior a *k*, dejando por tanto un conjunto de registros por asignar. Así se conservan sólo aquellos grupos con cardinalidad entre *k* y 2*k*-1.

El algoritmo de reasignación tiene por objeto asegurar la creación de cromosomas que codifiquen soluciones factibles, es decir, k-particiones. Su funcionamiento se basa en la formación de grupos de k elementos con los registros sobrantes en el proceso de inicialización, o bien trata de añadir registros a algún grupo ya formado con cardinalidad inferior a 2k-1. Puede ocurrir que al finalizar este proceso sigan habiendo grupos con cardinalidad inferior a *k*. Por lo tanto se deberá repetir el proceso tantas veces como sea necesario, hasta obtener una k-partición factible. Una vez hecho esto, los diferentes cromosomas representan soluciones candidatas a óptimas.

III-C. Función de fitness

Para que el AG pueda evolucionar hacia poblaciones cada vez mejores se debe conocer el grado de adaptación de los individuos. Por tanto, debemos disponer de una función de aptitud/adaptación (*fitness*) que nos permita valorar la calidad de cada individuo de la población.

En el caso particular de la microagregación, nos interesa medir la homogeneidad de los grupos que conforman una k-partición, para determinar la calidad del proceso de microagregación. Para ello existen diferentes formas basadas en diferentes definiciones del concepto de distancia (e.g. distancia Euclidiana, distancia de Minkowski, distancia de Chebyshev, etc.).

La forma más utilizada para medir la homogeneidad de una k-partición consiste en calcular la suma de errores cuadráticos *SSE* expuesta en la Ecuacion 1. El SSE se calcula sobre los datos después de estandarizarlos, es decir, después de restar al valor de cada atributo el promedio de dicho atributo y dividiendo el resultado por su desviación típica.

Considerando que la k-partición óptima es aquella que consigue tener el SSE mínimo, nuestro objetivo es minimizarlo. A raíz de esto, la función de evaluación que aplicaremos a cada cromosoma será la siguiente:

$$fitness = \frac{1}{SSE + 1} \tag{2}$$

Cuanto mayor sea el fitness de un cromosoma más apto lo consideraremos, ya que cuanto menor sea el *SSE*, mayor será el fitness obtenido. Se ha sumado un valor entero al *SSE* para evitar una división por cero en caso de que éste tuviese un valor nulo. De este modo podemos evaluar cada individuo y compararlo con el resto de cromosomas y determinar su calidad respecto al resto de la población.

III-D. Algoritmos de selección

Para que la población evolucione debe existir un proceso de selección consistente en escoger un subconjunto de la

población que, mediante algún método de cruce, forme la siguiente generación. En nuestra implementación, el AG es capaz de trabajar con dos métodos de selección: la ruleta (*roulette-wheel*) [19] y el FUSS (*Fitness Uniform Selection Scheme* [20]).

El algoritmo de selección por ruleta determina los individuos (i.e. cromosomas) que podrán reproducirse de forma aleatoria ponderada al *fitness* del individuo *cf.* Figura 2. De este modo los individuos mejor adaptados tienen más probabilidad de ser seleccionados. El principal inconveniente de este método es que tiende a converger rápidamente, ya que no preserva la diversidad en la población.

Mediante el esquema de selección *FUSS* se pretende evitar los inconvenientes del método de la ruleta, preservando la diversidad genética. Este método consiste en seleccionar de forma aleatoria un valor de *fitness f*, seleccionado uniformemente en el rango de valores posibles (F), en nuestro caso $F \in (0, 1]$. El individuo de la población con *fitness f* (o con el más cercano a *f*) pasará a la siguiente generación (*cf* Figura 3).

Podemos definir el algoritmo FUSS más detalladamente como sigue:

- Sean f_{min} y f_{max} los valores de *fitness* mínimo y máximo respectivamente de la población actual *P*.
- Se selecciona un valor de fitness f de forma uniforme en el intervalo [f_{min}, f_{max}].
- El individuo $i \in P$ con el fitness más cercano a f será seleccionado para pasar a la siguiente generación, posiblemente después de aplicarle los operadores genéticos que correspondan. En caso de que varios individuos posean la misma distancia de cercanía al valor de fitness f, se escogerá a uno de forma aleatoria.

La diferencia principal entre ambos métodos radica en como aplican presión genética, es decir, sobre que sector de la población tienden a seleccionar. En el caso de la ruleta, los individuos con un *fitness* alto reciben más presión que los demás y por ello se tiende a perder la diversidad rápidamente. Por contra, el método FUSS ejerce presión genética por igual en todo el rango de posibles valores de *fitness*. Sin embargo la presión que reciben los individuos es inversamente proporcional a la densidad de elementos en un intervalo de *fitness* dado. En realidad el método FUSS tiende a seleccionar tanto individuos bien adaptados como mal adaptados y mediocres. Gracias a este comportamiento, el método permite preservar la diversidad genética y mejora la eficiencia del AG en su búsqueda de la solución óptima.

III-E. Operadores genéticos

Una vez el algoritmo de selección ha determinado los individuos que pasan a la siguiente generación, estos deben reproducirse con la esperanza de mejorar su adaptación en la siguiente generación. La reproducción se lleva a cabo mediante la operación de cruce (*crossover*) véase Figura 4. Mediante el cruce, dos progenitores intercambian su material genético y dan como resultado dos descendientes que les substituyen en la siguiente generación.

Además, también usamos el operador de mutación (*mutation*) que permite, cuando se usa correctamente, introducir

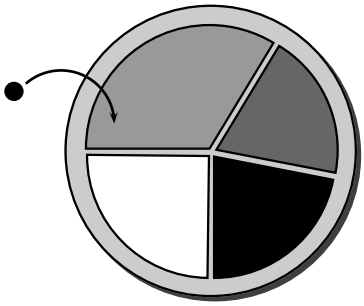


Figura 2. Ejemplo gráfico de selección mediante el algoritmo de la ruleta. Cuanto mayor es el *fitness* del cromosoma, mayor es su superficie en la ruleta y, por tanto, mayor es la probabilidad de que la bola caiga sobre él y sea seleccionado.

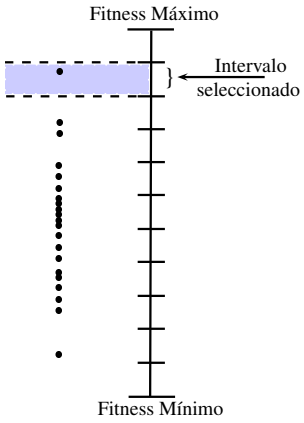


Figura 3. Ejemplo gráfico de selección mediante *fitness* uniforme. En el diagrama se muestra como los cromosomas (representados por puntos) están ordenados según su *fitness*). De forma aleatoria se selecciona un intervalo de tamaño constante entre el *fitness* máximo y el mínimo. Finalmente, el cromosoma que se encuentra en ese intervalo es seleccionado para su reproducción.

variabilidad en la población. Este operador produce un cambio aleatorio sobre un gen seleccionado aleatoriamente de un cromosoma dado. Teniendo en cuenta que nuestro método usa una codificación n-aria, el operador de mutación consiste en substituir el símbolo del gen por otro símbolo $s \in [0, G]$ de nuestro alfabeto (cf. Figura 5).

La probabilidad con la que un cromosoma se reproduce mediante cruce se conoce como *crossover rate*. La probabilidad con la que se producen mutaciones en los genes de un cromosoma se denomina *mutation rate*. Estas probabilidades suelen determinarse de forma empírica y cambian con cada problema. Nosotros hemos usado los valores recomendados en [18].

III-F. Individuos no factibles

La función de *fitness* determina la capacidad de un individuo de adaptarse a un entorno. Los diferentes individuos de la población codifican una solución particular del problema. En el caso de la microagregación multivariante, los individuos deben considerar restricciones en la cardinalidad de los subconjuntos. Tras la inicialización de la población, todos los individuos representan una k-partición candidata a ser la solución óptima del problema. Sin embargo, tras la aplicación de los operadores

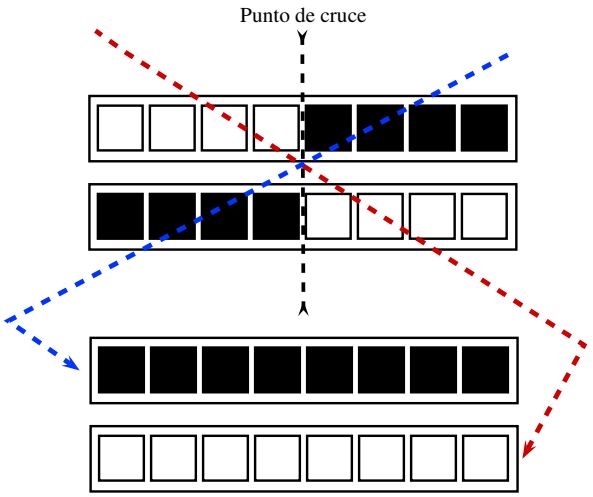


Figura 4. Ejemplo de cruce monopunto. Dado un punto de cruce seleccionado aleatoriamente, los dos progenitores intercambian su material genético para generar dos descendientes.

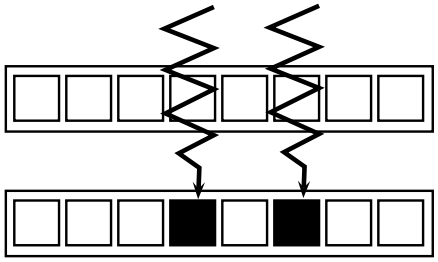


Figura 5. Ejemplo de mutación en dos genes de un cromosoma para un alfabeto binario.

de cruce y mutación, es posible que un cromosoma represente una partición en la que alguno de los grupos no presente una cardinalidad comprendida en el intervalo [k, 2k-1]. Este cromosoma representará una solución no factible para nuestro problema de microagregación multivariante.

La función de *fitness* no siempre es capaz de reflejar este hecho, por lo tanto se debe indicar de alguna forma que estos individuos no forman parte de la zona de búsqueda óptima. Existen diferentes maneras de abordar este problema:

1. Dejar que estos individuos no factibles sigan evolucionando, dando lugar a posibles individuos válidos.
2. Asignarles un valor de fitness nulo.
3. Intentar reconstruir aquellos individuos que no verifiquen las restricciones del problema. Esto se lleva a cabo a través de un nuevo operador, denominado reparador.
4. Penalizar el valor de fitness con una cantidad que guarde relación con las restricciones que viola.

Las dos primeras opciones son extremas, y pueden dar lugar a un comportamiento indeseado del AG. La tercera puede producir buenos resultados si se realiza de forma adecuada, pero tiene el inconveniente de aportar un tiempo de computación adicional. La forma de tratar a los individuos no factibles que utilizamos en nuestro método se lleva a cabo a través de un sistema de penalización. Como el objetivo de nuestro problema es maximizar la función de *fitness*, penalizamos a un individuo disminuyéndosela.

A veces es conveniente que el grado de penalización sea proporcional a lo lejos que esté el individuo de la zona de soluciones factibles. En nuestro caso esto no es fácil de determinar, y se ha preferido realizar una penalización menos precisa antes que aumentar el tiempo de procesamiento que este hecho comportaría. Encontrar un buen valor de penalización no es fácil, ya que la intensidad de las penalizaciones influirá de forma decisiva en la evolución del AG. Después de cientos de pruebas, decidimos penalizar a los individuos no factibles reduciendo su *fitness* a una décima parte del original.

IV. Resultados experimentales

En esta sección presentamos los resultados obtenidos tras la aplicación de nuestro método. Exponemos los resultados de la realización de diferentes pruebas sobre conjuntos de datos generados de forma aleatoria, comparando los resultados obtenidos mediante el método de selección FUSS, el método de selección por ruleta y el método no basado en AG, MDAV.

Para realizar los experimentos hemos usado conjuntos de datos de cardinalidad y dimensión variables. Los valores contenidos en dichos conjuntos de datos siguen una distribución uniforme en el intervalo [-10000, 10000].

Todas las pruebas que se detallan a continuación han sido realizadas en un Pentium 4 a 3 GHz bajo el sistema operativo Windows XP. Los programas están codificados en C++.

IV-A. Efecto del elitismo

El elitismo es un método que primero copia el mejor individuo (o varios) a la próxima generación. Se ha querido probar el efecto que tiene este método en el problema a resolver para comprobar si ofrece un mejor resultado. Por ello la primera prueba consiste en ejecutar el mismo AG con y sin elitismo, para los métodos de selección sometidos a estudio: *FUSS* y Ruleta. Se ha implementado el elitismo conservando en cada iteración al individuo que representa la mejor solución (i.e. el que tiene el valor de *fitness* más alto). Se han utilizado los siguientes parámetros:

- $Mr = 0{,}1$
- $Cr = 0{,}4$
- Tamaño de la población fijado a 100 cromosomas.
- Número de elementos n = 11, 20, 30, 40, 50, 100, 150, 200
- Dimensiones de los elementos $d = 2, 4, 6, 8, 10$
- Parámetro de seguridad $k = 3$
- Criterio de parada fijado a 5000 generaciones.

La configuración anterior implica el estudio de 40 configuraciones diferentes. Teniendo en cuenta que ejecutamos cada una de ellas con y sin elitismo, para ambas técnicas de selección, repitiendo cada una de las ejecuciones 10 veces, hace que hayamos realizado un total de 6400 pruebas.

Como podemos observar en las figuras 6 y 7, bajo las mismas condiciones el AG con elitismo obtiene siempre mejores resultados que sin elitismo. La falta de elitismo implica que la forma en que se escogen los individuos que pasan a formar parte de la siguiente generación se hace de forma totalmente aleatoria. Este hecho implica que no tienen porque conservarse los mejores individuos de la generación actual, lo que puede

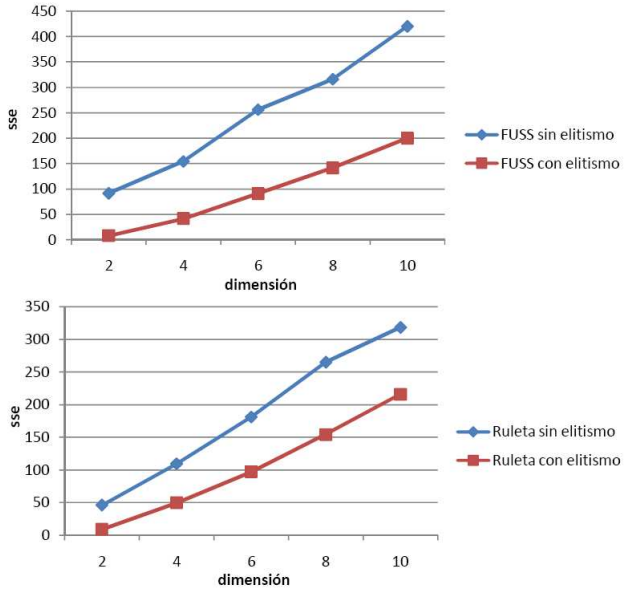


Figura 6. Media del SSE para un conjunto de datos con n = 50 ejecutando el AG con y sin elitismo mediante *FUSS* (arriba) y *Ruleta* (abajo).

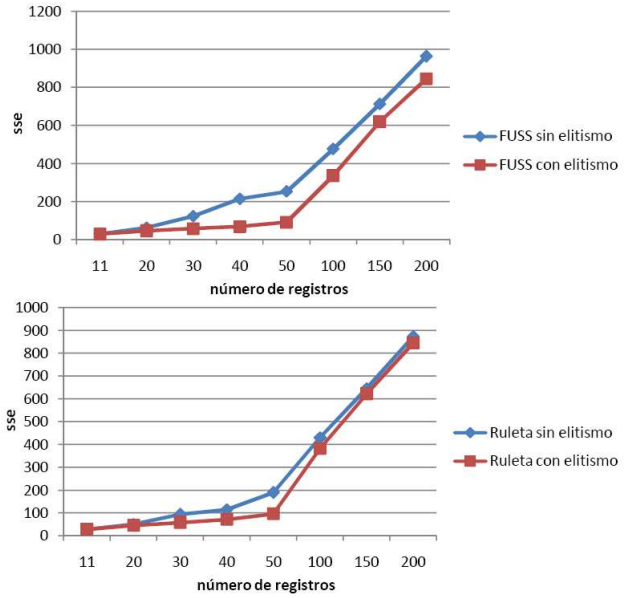


Figura 7. Media del SSE para los distintos valores de n ejecutando el AG con y sin elitismo mediante *FUSS* (arriba) y *Ruleta* (abajo), con d=6 y k=3.

producir que se pierdan los logros conseguidos en ésta y hacer que la evolución sea muy lenta. En cambio, la versión con elitismo, al mantener la mejor solución encontrada en cada iteración, consigue llegar antes a una buena solución. Por todo ello, se ha decidido trabajar con la versión con elitismo, que se utilizará en todos los resultados que se expondrán posteriormente.

IV-B. Elección de las probabilidades de mutación y cruce

En [18] Solanas et al. determinaron estadísticamente el valor más adecuado para la probabilidad de cruce y mutación cuando el método de selección utilizado es el método de ruleta. Nuestro primeros experimentos se centran en determinar si dichos valores son igualmente útiles cuando el método de selección utilizado es el método FUSS.

La primera prueba consiste en realizar una serie de ejecuciones del AG para determinar qué probabilidades de mutación y cruce son las más adecuadas para proporcionar un buen resultado cuando el método de selección es el FUSS. Se han utilizado los siguientes parámetros:

- Tasa de mutación (*mutation rate*, Mr) variándola de 0 a 1 con intervalos de 0.1.
- Probabilidad de cruce (*crossover rate*, Cr) variándola de 0 a 1 con intervalos de 0.1.
- Tamaño de la población (*population size*), i.e. el número de cromosomas fijado a 100. En [18] se prueba que este tamaño de población produce buenos resultados.
- Conjunto de datos de 20 elementos de 2 dimensiones.
- Parámetro de seguridad $k = 3$.
- Criterio de parada fijado a 1000 generaciones.

La configuración anterior implica la realización de 242 ejecuciones del AG, 121 para cada método de selección.

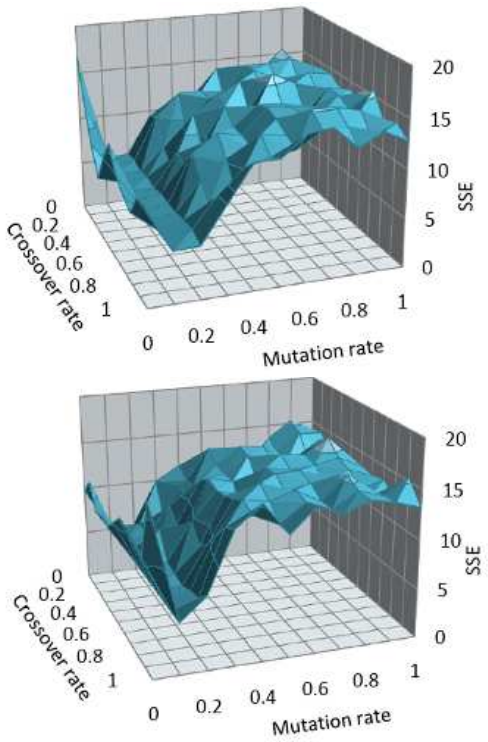


Figura 8. SSE obtenido para cada par (Cr, Mr) para un conjunto de datos 20x2 mediante *FUSS* (arriba) y *Ruleta* (abajo).

La Figura 8 muestra el SSE obtenido para cada par (Cr, Mr) mediante el método de selección *FUSS* y *Ruleta*. Se puede observar que para $Mr = 0{,}1$ se obtiene en ambos casos el SSE más bajo. También vemos que con una buena probabilidad de mutación fijada (i.e. 0.1), el valor de la probabilidad de cruce no es un factor relevante para obtener un buen resultado, a excepción del valor nulo. Teniendo en cuenta que cuanto mayor sea el tamaño de la población mayor será el coste computacional y temporal que provocará una alta probabilidad de cruce, un buen valor para asignarle sería alrededor de 0.3 o 0.4.

Estos resultados confirman los obtenidos en [18] en cuanto al método de selección por ruleta y indican que el método de selección FUSS responde de manera equivalente.

IV-C. Comparativa entre FUSS, Ruleta y MDAV

Una vez definidas las probabilidades de mutación y cruce comparamos los métodos de selección sometidos a estudio. Para ello realizamos una serie de ejecuciones del AG utilizado los siguientes parámetros:

- $Mr = 0{,}1$
- $Cr = 0{,}4$
- Tamaño de la población fijado a 100 cromosomas.
- Número de elementos n = 11, 20, 30, 40, 50, 100, 150, 200
- Dimensiones de los elementos $d = 2, 4, 6, 8, 10$
- Parámetro de seguridad $k = 3, 4, 5$
- Criterio de parada fijado a 5000 generaciones.

La configuración anterior implica el estudio de 120 configuraciones diferentes. Para cada una de ellas se ha ejecutado el AG mediante FUSS y Ruleta 10 veces. El método MDAV, al ser determinista, se ha ejecutado una sola vez, ya que para una configuración determinada siempre nos proporcionará el mismo resultado. Todo ello hace que hayamos realizado un total de 2520 pruebas, algunas de ellas realizadas en la subsección IV-A.

Nos hemos centrado en el estudio de:

1. Coste en tiempo del método de selección por ruleta respecto del método de selección FUSS
2. SSE obtenido por ambos métodos de selección.
3. SSE obtenido mediante genéticos vs. microagregación multivariante clásica (MDAV)

IV-C1. Comparación temporal: Comparamos ambos métodos de selección en cuanto al tiempo de ejecución se refiere. En las Figuras 9 y 10 se muestra la media del tiempo de 10 ejecuciones para FUSS y Ruleta en algunas de las configuraciones. En la primera fijamos $n = 50$ y variamos la dimensión d, utilizando $d = 2$, $d = 4$, $d = 6$, $d = 8$ y $d = 10$. En la segunda fijamos $d = 6$ y variamos n, utilizando $n = 20$, $n = 30$, $n = 40$, $n = 50$, $n = 100$, $n = 150$ y $n = 200$. En ambos casos utilizamos un parámetro de seguridad $k = 3$. Se puede observar como no hay variaciones en referencia al tiempo de ejecución al utilizar un método de selección u otro. Este resultado se extiende a todas las configuraciones probadas. Podemos concluir que ambos métodos son equivalentes en tiempo.

IV-C2. SSE obtenido por la microagregación basada en AG: En la Figura 11 se muestra la media del SSE obtenido en 10 ejecuciones para diferentes configuraciones. Se observa como para $n = 11$ ambos métodos consiguen llegar al mismo resultado. Es a partir de $n = 20$ donde FUSS empieza a mejorar a la Ruleta, llegándolo a superar hasta en un 20 por ciento. Los resultados completos pueden verse en las Tablas I, II y III.

Podemos concluir que el método de selección FUSS consigue obtener en el mismo tiempo un mejor resultado que el método de selección por ruleta.

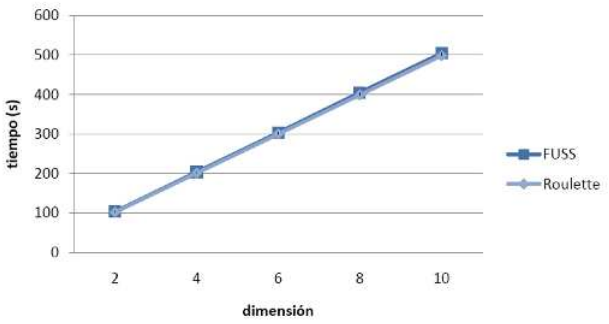


Figura 9. Media del tiempo de ejecución del AG para 50 registros mediante *FUSS* y *Ruleta*, con los distintos valores de d y $k = 3$.

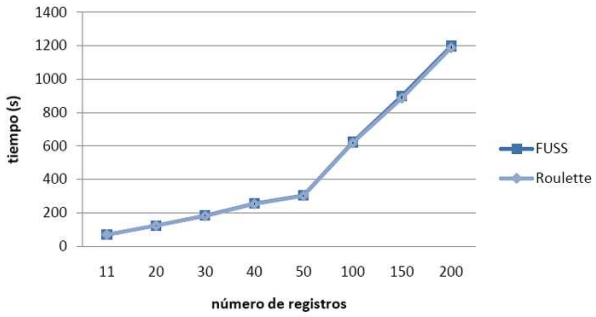


Figura 10. Media del tiempo de ejecución del AG para los distintos valores de n mediante *FUSS* y *Ruleta*, con $d = 6$ y $k = 3$.

IV-C3. MDAV vs AG: MDAV es un método de microagregación de tamaño fijo, y hasta ahora es el método heurístico que proporciona un mejor resultado en microagregación de grandes conjuntos de datos. En la Figura 12 se muestra la evolución de la media del SSE en diferentes conjuntos de datos. En concreto mostramos los resultados para $n = 11$, $n = 20$, $n = 40$ y $n = 50$, para los diferentes valores de d. Puede observarse que para $d = 11$, los AG consiguen mejorar el valor del SSE que obtiene MDAV. El sistema de microagregación mediante AG supera a MDAV hasta $n = 40$. Justo en este valor de n, cuando la dimensión crece, e.g. $d = 8$, el método MDAV empieza a superar a los AG. Es para $n = 50$ donde MDAV consigue obtener siempre mejores valores de SSE. A la luz de estos resultados queda claro que los AG son una buena solución para conjuntos de datos pequeños, mientras que MDAV funciona mejor para conjuntos mayores.

V. CONCLUSIÓN

En este artículo hemos presentado un sistema de microagregación multivariante basado en algoritmos genéticos. Hemos estudiado el efecto del sistema de selección sobre la pérdida de información asociada a la microagregación. Para ello, hemos comparado el método clásico de ruleta y el método de selección uniforme de *fitness*. Los resultados experimentales ponen de manifiesto la superioridad del método uniforme.

No obstante este trabajo pretende ser la introducción de un estudio más amplio. Por este motivo existen diversas líneas de investigación abiertas que van a ser abordadas en el futuro:

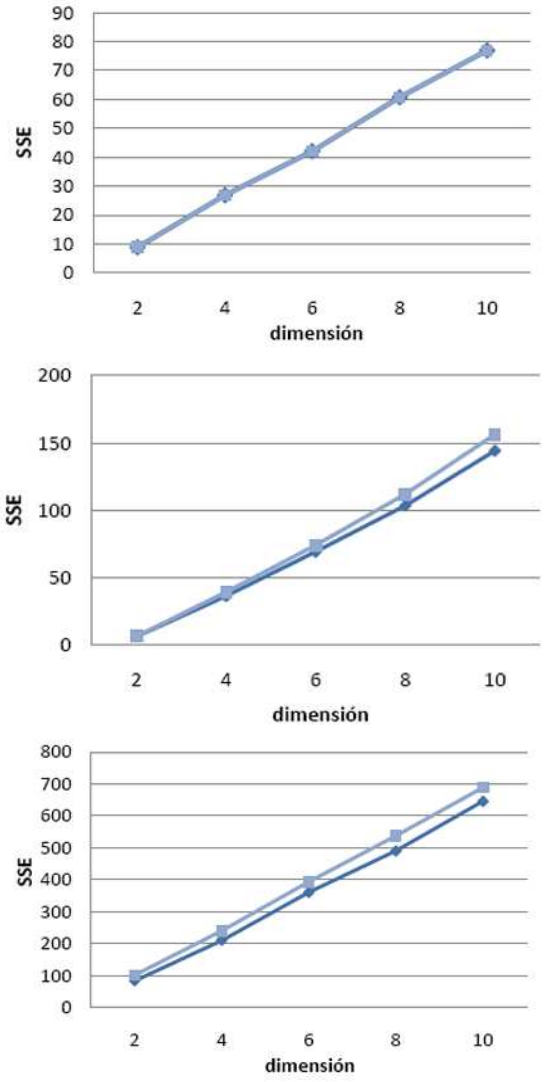


Figura 11. Media del SSE para un conjunto de datos con n = 11 (primer gráfico), n = 30 (segundo gráfico), n = 100 (tercer gráfico), para k = 4. FUSS (Azul claro), Ruleta (Azul oscuro).

n	d	k	SSE FUSS	SSE Ruleta	SSE MDAV
11	2	3	7.03	7.03	7.03
11	4	3	16.07	16.07	22.45
11	6	3	28.77	28.77	30.70
11	8	3	48.25	48.25	50.87
11	10	3	61.55	61.55	67.08
20	2	3	5.19	5.24	6.20
20	4	3	18.47	19.08	19.48
20	6	3	46.33	46.63	51.79
20	8	3	61.92	62.47	74.52
20	10	3	82.42	82.83	98.00
30	2	3	3.32	3.44	3.65
30	4	3	27.66	28.38	24.80
30	6	3	56.88	58.98	64.83
30	8	3	83.02	83.75	99.02
30	10	3	117.88	120.82	131.51
40	2	3	4.82	5.76	5.14
40	4	3	31.67	32.84	27.20
40	6	3	67.59	72.36	63.73
40	8	3	113.47	118.74	126.52
40	10	3	160.14	166.01	164.84
50	2	3	7.85	9.00	3.38
50	4	3	41.10	49.57	26.80
50	6	3	90.84	96.91	69.31
50	8	3	141.19	154.29	128.24
50	10	3	199.72	216.11	178.60
100	2	3	89.90	110.91	4.54
100	4	3	204.47	245.59	42.22
100	6	3	336.59	381.68	111.64
100	8	3	465.11	522.83	223.43
100	10	3	595.22	642.02	309.57
150	2	3	193.83	196.93	4.36
150	4	3	405.15	407.52	57.26
150	6	3	619.59	622.26	148.99
150	8	3	835.73	843.82	287.70
150	10	3	1056.79	1057.68	428.18
200	2	3	264.55	263.39	4.66
200	4	3	552.68	555.75	72.11
200	6	3	846.32	845.01	182.70
200	8	3	1132.48	1135.47	357.02
200	10	3	1428.90	1429.50	534.88

Cuadro I
RESULTADOS PARA DIVERSOS CONJUNTOS DE DATOS DE TAMAÑO n Y DIMENSIÓN d PARA $k = 3$

- Utilización de distintas técnicas de selección, como torneo o ranking.
- Extensión del método propuesto para tratar las nuevas formas de privacidad como la l-diversity [21] y la t-closeness [22].
- Estudio e implementación del operador reparador. Es posible que el tiempo de cálculo necesario para realizar la reparación de cromosomas no factibles, se vea compensado por la mejora en el rendimiento del AG.
- Estudio de nuevas formas de codificación que permitan la aplicación de los AG a conjuntos de datos mayores.
- Utilización de técnicas para evitar la convergencia prematura de la población y mantener su diversidad , tales como *nitching*.

AGRADECIMIENTOS

Los autores forman parte de la Cátedra UNESCO de Privacidad de Datos, pero sus opiniones no necesariamente reflejan la posición de la UNESCO ni comprometen a dicha organización. Este trabajo ha sido parcialmente subvencionado por el Ministerio Español de Educación y Ciencia mediante los proyectos TSI2007-65406-C03-01 "E-AEGIS" y CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES", y por la Generalitat de Catalunya mediante la subvención 2005 SGR 00446.

REFERENCIAS

[1] "Directive 2002/58/EC on privacy and electronic communications," http://europa.eu.int/eur-lex/pri/en/oj/dat/2002/201/201200207 31en00370047.pdf.

[2] C. Boyens, R. Krishnan, and R. Padman, "On privacy-preserving access to distributed heterogeneous healthcare information," in *Proceedings of the 37th Hawaii International Conference on System Sciences HICSS-37*, I. C. Society, Ed., Big Island, HI., 2004.

[3] HIPAA, "Health insurance portability and accountability act," 2004, http://www.hhs.gov/ocr/hipaa/.

[4] A. Solanas and A. Martínez-Ballesté, "Privacy protection in location-based services through a public-key privacy homomorphism," in *Fourth European PKI Workshop: theory and practice*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2007, pp. 362 – 368, palma de Mallorca, Spain.

n	d	k	SSE FUSS	SSE Ruleta	SSE MDAV
11	2	4	8.89	8.89	8.89
11	4	4	26.92	26.92	27.13
11	6	4	42.13	42.13	45.25
11	8	4	60.82	60.82	63.14
11	10	4	76.99	76.99	81.58
20	2	4	8.47	8.57	8.85
20	4	4	27.35	31.81	30.59
20	6	4	52.64	56.29	54.20
20	8	4	75.43	81.17	80.73
20	10	4	97.37	105.17	117.34
30	2	4	6.32	6.58	6.36
30	4	4	36.49	39.53	42.55
30	6	4	69.34	74.24	89.60
30	8	4	103.50	112.23	112.68
30	10	4	144.24	156.45	163.08
40	2	4	7.15	7.16	6.17
40	4	4	39.54	39.67	36.86
40	6	4	85.76	88.95	89.67
40	8	4	140.14	146.56	156.66
40	10	4	195.95	198.14	207.18
50	2	4	10.39	10.75	6.70
50	4	4	53.08	54.00	46.10
50	6	4	113.73	117.84	116.36
50	8	4	172.65	181.81	163.75
50	10	4	245.81	251.04	247.23
100	2	4	83.67	101.51	6.61
100	4	4	210.37	240.38	64.57
100	6	4	360.72	393.49	153.35
100	8	4	490.29	537.86	298.91
100	10	4	645.80	689.25	395.63
150	2	4	192.86	215.67	7.68
150	4	4	429.60	444.72	84.43
150	6	4	655.08	680.30	209.77
150	8	4	892.60	907.03	380.52
150	10	4	1128.33	1142.19	568.41
200	2	4	294.66	292.61	6.68
200	4	4	603.60	605.89	93.57
200	6	4	927.41	926.32	247.72
200	8	4	1239.28	1242.95	460.94
200	10	4	1550.48	1552.23	678.27

Cuadro II
RESULTADOS PARA DIVERSOS CONJUNTOS DE DATOS DE TAMAÑO n Y DIMENSIÓN d PARA $k = 4$

n	d	k	SSE FUSS	SSE Ruleta	SSE MDAV
11	2	5	9.90	9.90	9.90
11	4	5	27.65	27.65	28.51
11	6	5	43.10	43.10	43.53
11	8	5	62.41	62.41	64.56
11	10	5	76.99	76.99	84.95
20	2	5	10.62	12.52	10.79
20	4	5	34.30	37.87	35.85
20	6	5	60.78	69.52	73.97
20	8	5	85.02	91.72	86.32
20	10	5	111.58	119.71	125.84
30	2	5	7.70	7.74	7.41
30	4	5	45.17	45.95	47.91
30	6	5	80.90	86.07	103.51
30	8	5	120.94	126.48	127.86
30	10	5	164.48	174.18	176.18
40	2	5	9.31	8.09	9.34
40	4	5	47.75	52.29	41.16
40	6	5	96.72	98.98	101.15
40	8	5	161.94	164.29	179.73
40	10	5	220.61	220.56	237.41
50	2	5	13.18	13.22	8.75
50	4	5	64.06	62.91	55.08
50	6	5	131.28	133.48	131.59
50	8	5	195.14	198.90	188.08
50	10	5	275.14	278.07	276.58
100	2	5	79.86	102.39	8.92
100	4	5	220.00	257.15	77.88
100	6	5	366.03	407.69	182.36
100	8	5	519.88	559.89	345.65
100	10	5	674.09	716.56	479.67
150	2	5	211.76	224.21	9.93
150	4	5	435.84	465.92	99.53
150	6	5	675.98	700.46	239.67
150	8	5	914.43	958.67	440.93
150	10	5	1152.24	1187.76	648.47
200	2	5	309.07	308.75	10.20
200	4	5	631.14	639.83	108.68
200	6	5	957.51	965.70	281.10
200	8	5	1294.13	1300.49	544.80
200	10	5	1620.00	1635.63	825.12

Cuadro III
RESULTADOS PARA DIVERSOS CONJUNTOS DE DATOS DE TAMAÑO n Y DIMENSIÓN d PARA $k = 5$

[5] ——, "A ttp-free protocol for location privacy in location-based services," *Computer Communications*, vol. 31, no. 6, pp. 1181 – 1191, 2008, doi:10.1016/j.comcom.2008.01.007.

[6] E. 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.

[7] C. Privacy, "Canadian privacy regulations," 2005, http://www.media-awareness.ca/english/issues/privacy/canadian_legislation_privacy.cfm.

[8] USPrivacy, "U.S. privacy regulations," 2005, http://www.media-awareness.ca/english/issues/privacy/us_legislation_privacy.cfm.

[9] 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*, vol. 18, no. 4, pp. 345–354, 2001.

[10] L. Willenborg and T. DeWaal, *Elements of Statistical Disclosure Control*. Springer-Verlag, New York, 2001.

[11] M. Laszlo and S. Mukherjee, "Minimum spanning tree partitioning algorithm for microaggregation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 7, pp. 902–911, 2005.

[12] J. Domingo-Ferrer, F. Sebé, and A. Solanas, "A polynomial-time approximation to optimal multivariate microaggregation," *Comput. Math. Appl.*, vol. 55, no. 4, pp. 714–732, 2008.

[13] J. Domingo-Ferrer and J. M. Mateo-Sanz, "Practical data-oriented microaggregation for statistical disclosure control," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 1, pp. 189–201, 2002.

[14] J. Domingo-Ferrer and V. Torra, "Ordinal, continuous and heterogenerous k-anonymity through microaggregation," *Data Mining and Knowledge Discovery*, vol. 11, no. 2, pp. 195–212, 2005.

[15] A. Hundepool, A. V. 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*. Voorburg NL: Statistics Netherlands, may 2005, http://neon.vb.cbs.nl/casc.

[16] A. Solanas and A. Martínez-Balleste, "V-MDAV: Variable group size multivariate microaggregation," in *COMPSTAT'2006*, Rome, 2006, pp. 917–925.

[17] C. Darwin, *The origin of Species by means of natural selection. On the preservation of favoured races in the struggle for life*. John Murray. London, 1859.

[18] A. Solanas, A. Martinez-Balleste, J. Mateo-Sanz, and J. Domingo-Ferrer, "Multivariate microaggregation based on genetic algorithms," *Intelligent Systems, 2006 3rd International IEEE Conference on*, pp. 65–70, 2006.

[19] J. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[20] M. Hutter, "Fitness uniform selection to preserve genetic diversity," *Submitted to IEEE Transactions Evolutionary Computation*, no. IDSIA-01-01, p. 13 pages, Jan. 2001. [Online]. Available: http://www.hutter1.net/ai/pfuss.htm

[21] A. Machanavajhala, J. Gehrke, D. Kiefer, and M. Venkitasubramanian, "L-diversity: privacy beyond k-anonymity," in *Proceedings of the IEEE ICDE 2006*, 2006.

[22] N. Li, T. Li, and S. Venkatasubramanian, "T-closeness: privacy beyond k-anonymity and l-diversity," in *Proceedings of the IEEE ICDE 2007*, 2007.

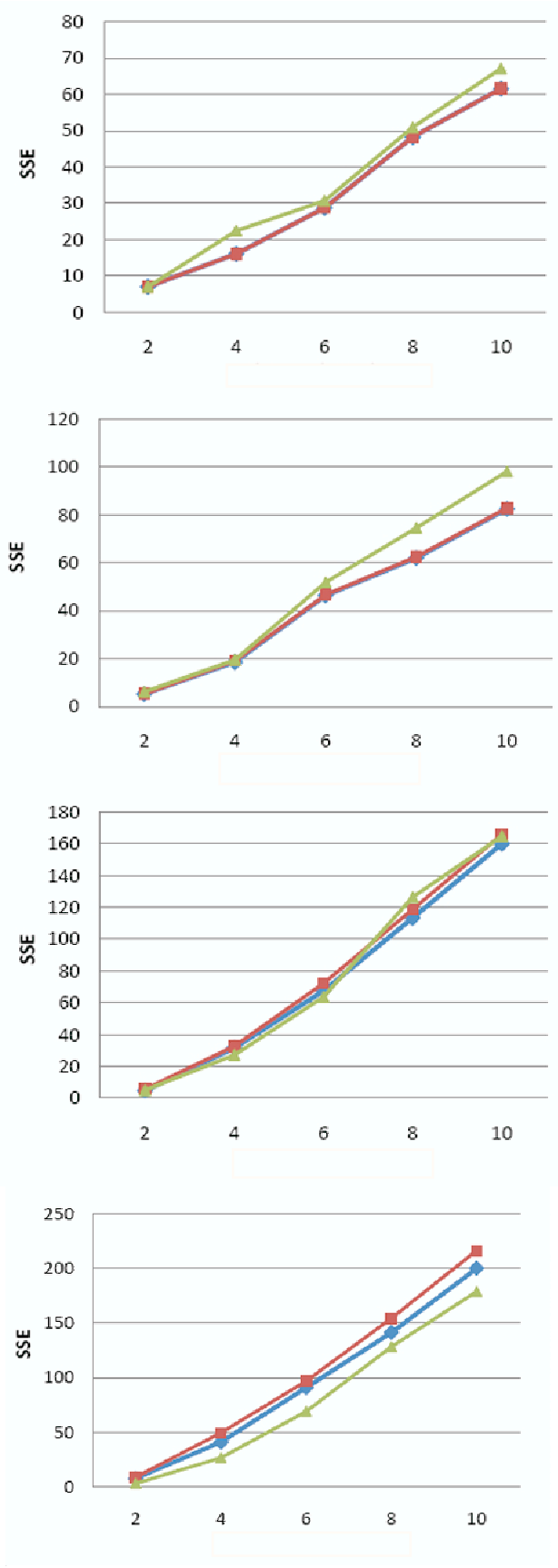


Figura 12. Media del SSE para un conjunto de datos con n = 11 (primer gráfico), n = 20 (segundo gráfico), n = 40 (tercer gráfico), y n = 50 (cuarto gráfico), para k = 3. Verde (MDAV), azul (FUSS), rojo (Ruleta).