Information Processing and Management 48 (2012) 476–487

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

User *k*-anonymity for privacy preserving data mining of query logs

Guillermo Navarro-Arribas [a,*], Vicenç Torra [a], Arnau Erola [b], Jordi Castellà-Roca [b]

[a] IIIA, Institut d'Investigació en Intel·ligència Artificial – CSIC, Consejo Superior de Investigaciones Científicas, Campus UAB s/n, 08193 Bellaterra, Catalonia, Spain
[b] Departament d'Enginyeria Informàtica i Matemàtiques, UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Av. Països Catalans 26, E-43007 Tarragona, Spain

ARTICLE INFO

Article history:
Received 26 February 2010
Received in revised form 27 December 2010
Accepted 5 January 2011
Available online 8 February 2011

Keywords:
Privacy
Query log
Microaggregation
k-Anonymity
Clustering
Web search

ABSTRACT

The anonymization of query logs is an important process that needs to be performed prior to the publication of such sensitive data. This ensures the anonymity of the users in the logs, a problem that has been already found in released logs from well known companies. This paper presents the anonymization of query logs using microaggregation. Our proposal ensures the *k*-anonymity of the users in the query log, while preserving its utility. We provide the evaluation of our proposal in real query logs, showing the privacy and utility achieved, as well as providing estimations for the use of such data in data mining processes based on clustering.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

The search logs generated by a web search engine (from now on WSE) is a great source of information for researchers or marketing companies, but at the same time their publication may expose the privacy of the users from which the logs were generated (Jones et al., 2007). There is at least one well known case of released search logs with poor anonymization, which have been shown to reveal enough information to re-identify some users. The release was done by AOL in an attempt to help the information retrieval research community, and ended up with not only important damage to AOL users privacy, but also a major damage to AOL itself with several class actions suits and complaints against the company (EFF, 2009; Mills, 2006).

Ideally, the search logs should be properly anonymized before they become public. The problem is that achieving a desirable degree of privacy in search logs is not easy, and presents an important trade-off between privacy and the usefulness of the data. There are several approaches (Cooper, 2008) to anonymize such data, but they are normally reduced to the deletion of specific queries or logs. Moreover, common techniques used in statistical disclosure control (SDC) have not been applied to this specific problem until very recently (Hong et al., 2009; Navarro-Arribas and Torra, 2009).

In this paper, we propose the application of microaggregation to anonymize search logs. This approach ensures a high degree of privacy, providing *k*-anonymity at user level, while preserving some of the data usefulness.

The paper is organized as follows. Section 2 introduces the problem and our motivations. In Section 3 we present the microaggregation of the query logs. Section 4 provides the evaluation of our proposal both in terms of privacy and usability, and finally, Section 5 concludes the paper.

* Corresponding author.
E-mail addresses: guille@iiia.csic.es (G. Navarro-Arribas), vtorra@iiia.csic.es (V. Torra), arnau.erola@urv.cat (A. Erola), jordi.castella@urv.cat (J. Castellà-Roca).

0306-4573/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2011.01.004

2. Motivations: query logs and privacy

As mentioned earlier, the release of query logs from AOL with poor anonymization, was a mistake normally regarded as a bad initiative taken by the company. Our objective is to provided a stronger anonymization so the release of query logs by search engine companies can be done without risking the privacy of their users.

A query log from a WSE is composed of lines of the form:

$$(id, q, t, r, u)$$

where *id* is the user identifier, *q* is the query string, *t* is a timestamp, *u* is the URL clicked by the user after the query, and *r* is the rank of the clicked URL. This format corresponds to logs released by AOL in 2006, Fig. 1 shows some real logs from the AOL data. The information provided in these logs is the same as the logs from AllTheWeb (Jansen and Spink, 2005), and it closely resembles other released data from Excite (Jansen et al., 2000) or AltaVista (Jansen et al., 2005). It is normally considered as a generic query log format.

For our work we have used the AOL query logs. As other released query logs, we have to bear in mind that they have already been anonymized by normally poor anonymization techniques. Thus, the clicked URL (u) is truncated to the domain name before publishing as a minor privacy measure. We can also assume that private information from the query terms such as social security numbers has been removed (Xiong and Agichtein, 2007).

The user identifier (id) is a unique identifier for each user. This identifier can be obtained directly by the WSE, where the user needs to log in, or indirectly by, for example, a combination of the URL, user agent, and cookies of the user from the Web server access logs. The user identifier is normally anonymized by a simple hash function or a similar approach. It has been shown that, even using such anonymization, users can be identified (Barbaro and Zeller, 2006). Moreover, hashing techniques, applied to the query terms, are vulnerable to frequency analysis (Kumar et al., 2007).

Other anonymization techniques have been developed for search logs, such as removing infrequent queries (Adar, 2007), or more sophisticated techniques to remove selected queries to preserve an acceptable degree of privacy (Poblete et al., 2010), or to choose the publishable queries (Korolova et al., 2009).

The work presented in this paper departs from an initial approach (Navarro-Arribas and Torra, 2009), which has been greatly improved both in terms of computational efficiency and on the results obtained.

2.1. Privacy by means of k-anonymity

To ensure anonymity, a well known principle used in statistical disclosure control is *k*-anonymity (Samarati, 2001; Sweeney, 2002), which states that each query to the anonymized data should return at least *k* equal records. It is clear that ensuring *k*-anonymity in search logs is a desirable goal, which somehow has been attempted by the previously mentioned related works. Nevertheless, these works did not completely ensure *k*-anonymity and the way they attempt to achieve it is by removing some queries from the log.

In this paper we present a technique to ensure *k*-anonymity in search logs by means of microaggregation, without having to explicitly remove any query from the log (although they are somehow perturbed). Moreover we provide *k*-anonymity at the user level. That is, in the protected version of the query logs, there are *k* indistinguishable users, so it is not feasible to re-identify users as in the case of the released logs from AOL (Barbaro and Zeller, 2006) for some given *k*.

We can represent each user by a simple ordered tree, grouping all its queries. Fig. 2 shows the representation of the users from the query logs of Table 1. All requests in the query log belonging to the same user are treated as a single record in the protection process.

```
24963762 myspace codes  2006-05-31 23:00:52  2  http://www.myspace-codes.com
24964082 bank of america  2006-05-31 19:41:07  1  http://www.bankofamerica.com
24967641 donut pillow  2006-05-31 14:08:53
24967641 dicontinued dishes  2006-05-31 14:29:38
24969374 orioles tickets  2006-05-31 12:31:57  2  http://www.greatseats.com
24969374 baltimore marinas  2006-05-31 12:43:40
```

Fig. 1. Example of search query log.

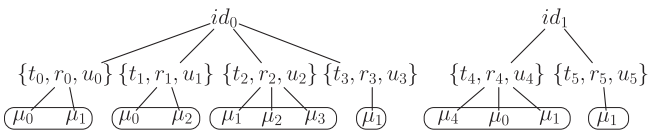

Fig. 2. Example of user query trees.

Table 1
Example of queries.

id	Query string	Timestamp	Rank	Clicked URL
id_0	(μ_0, μ_1)	t_0	r_0	u_0
id_0	(μ_0, μ_2)	t_1	r_1	u_1
id_0	(μ_1, μ_2, μ_3)	t_2	r_2	u_2
id_0	(μ_1)	t_3	r_3	u_2
id_1	(μ_4, μ_0, μ_1)	t_4	r_4	u_4
id_1	$(\mu 1)$	t_5	r_5	u_5

As we will show, our proposal aggregates different users to achieve user k-anonymity, which results in the loss of some information for each individual in the protected data. Achieving privacy in these scenarios always presents a trade-off between privacy and utility. In our case, we will show that even achieving a high degree of privacy, the data still preserve enough utility to be used in data mining processes.

In the following sections we will introduce the microaggregation of query logs, and discuss our results.

3. Microaggregation of query logs

In order to apply microaggregation to query logs we have to define the microaggregation process. Next sections introduce microaggregation and how we use it to protect query logs.

3.1. Microaggregation

Microaggregation is a statistical disclosure control technique, which provides privacy by means of clustering the data into small clusters and then replacing the original data by the centroids of the corresponding clusters.

Privacy is achieved because all clusters have at least a predefined number of elements, and therefore, there are at least k records with the same value. Note that all the records in the cluster replace a value by the value in the centroid of the cluster. The constant k is a parameter of the method that controls the level of privacy. The larger the k, the more privacy we have in the protected data.

Microaggregation was originally (Defays and Nanopoulos, 1993) defined for numerical attributes, but later extended to other domains. E.g., to categorical data in Torra (2004) (see also (Domingo-Ferrer and Torra, 2005)), and in constrained domains in Torra (2008).

From the operational point of view, microaggregation is defined in terms of partition and aggregation:

- *Partition*. Records are partitioned into several clusters, each of them consisting of at least k records.
- *Aggregation*. For each of the clusters a representative (the centroid) is computed, and then original records are replaced by the representative of the cluster to which they belong to.

From a formal point of view, microaggregation can be defined as an optimization problem with some constraints. We give a formalization below using λ_{ij} to describe the partition of the records in the sensitive data set X (each record denoted as x_j). That is, $\lambda_{ij} = 1$ if record x_j is assigned to the ith cluster. Let v_i be the representative of the ith cluster, then a general formulation of microaggregation with g clusters and a given k is as follows:

$$\text{Minimize } SSE = \sum_{i=1}^{g}\sum_{j=1}^{n}\lambda_{ij}(d(x_j, v_i))^2$$

$$\text{Subject to } \sum_{i=1}^{g}\lambda_{ij} = 1 \qquad \text{for all } j = 1, \ldots, n$$

$$2k \geqslant \sum_{j=1}^{n}\lambda_{ij} \geqslant k \quad \text{for all } i = 1, \ldots, g$$

$$\lambda_{ij} \in \{0, 1\}$$

For numerical data it is usual to require that $d(x, v)$ is the Euclidean distance. In the general case, when attributes $\mathbf{V} = (V_1, \ldots, V_s)$ are considered, x and v are vectors, and d becomes $d^2(x, v) = \sum_{V_i \in \mathbf{V}}(x_i - v_i)^2$. In addition, it is also common to require for numerical data that v_i is defined as the arithmetic mean of the records in the cluster. I.e., $v_i = \sum_{j=1}^{n}\lambda_{ij}x_i / \sum_{j=1}^{n}\lambda_{ij}$. As the solution of this problem is NP-Hard (Oganian and Domingo-Ferrer, 2001) when we consider more than one variable at a time (multivariate microaggregation), heuristic methods have been developed.

MDAV (Domingo-Ferrer and Mateo-Sanz, 2002) (Maximum Distance to Average Vector) is one of such existing algorithms. It is explained in detail in Algorithm 1, when applied to a data set X with n records and A attributes. The implementation of MDAV for categorical data is given in Domingo-Ferrer and Torra (2005).

Note that when all variables are considered at once, microaggregation is a way to implement k-anonymity (Samarati, 2001; Sweeney, 2002).

Algorithm 1.

```
Data: X: original data set, k: integer
Result: X : protected data set
begin
    while (|X| ⩾ 3 ∗ k) do
        Compute average record x̄ of all records in X;
        Consider the most distant record x_r to the average record x̄;
        Form a cluster around x_r. The cluster contains x_r together with the k − 1 closest records to x_r;
        Remove these records from data set X;
        Find the most distant record x_s from record x_r;
        Form a cluster around x_s. The cluster contains x_s together with the k − 1 closest records to x_s;
        Remove these records from data set X;
    if (|X| >= 2 ∗ k)
        Compute the average record x̄ of all records in X;
        Consider the most distant record x_r to the average record x̄;
        Form a cluster around x_r. The cluster contains x_r together with the k − 1 closest records to x_r;
        Remove these records from data set X;
    Form a cluster with the remaining records;
end
```

3.2. Distance and aggregation of query logs

To microaggregate user queries of the form shown in Fig. 2, we need to define a proper distance function for the partition step of the microaggregation and an aggregation operator to be used in the aggregation step.

We will denote the user query tree for a given user id_i as:

$$q(id_i) = (id_i, \varphi^i)$$

where $\varphi^i = (\varphi^i_1, \varphi^i_2, \varphi^i_3, \ldots)$ is the vector of queries for user id_i. That is, φ^i_j corresponds to the jth query for user id_i, and is composed of $\varphi^i_j = \left\{t^i_j, r^i_j, u^i_j, \phi^i_j\right\}$, where $\phi^i_j = (\mu_0, \mu_1, \mu_2, \ldots)$ is the query string (search terms used in the query). We will also use $|\varphi^i|$ as the number of queries for user id_i, and $\left|\phi^i_j\right|$ as the number of terms (words) in the query j of user id_i.

A previous step to the microaggregation is the normalization of the numeric data: timestamp, rank, number of queries per user, and number of terms per query. In general, given an attribute A with maximum value max(A) and minimum value min(A) in the original log, the normalization of x_i (the original values) and denormalization of x'_i (the protected values) for all $x_i \in A$, and $x'_i \in A'$ is given by:

$$\text{norm}(x_i) = \frac{x_i - \max(A)}{\max(A) - \min(A)}$$
$$\text{denorm}(x'_i) = (x'_i(\max(A) - \min(A)) + \min(A)$$

The normalized number of queries for user id_i is denoted as $\overline{|\varphi^i|}$, and the normalized number of terms in the query φ^i_j as $\overline{|\phi^i_j|}$.

3.2.1. User query distance

The distance is calculated as the aggregation of several distance functions for each pair of user queries. We define the distance functions used as:

- $d_{euclid}(x, y) = \sqrt{(x-y)^2}$: Euclidean distance will be used for the rank.
- $d_t(t_i, t_j)$: Distance between two timestamps t_1, t_2, as the Euclidean distance of the UNIX epoch representation of the timestamps.
- $d_u(u_i, u_j)$: Distance between two domain names (the clicked URL). Given two domain names: $X = x_n \ldots x_0$, and $Y = y_m \ldots y_0$, and assuming $m \geqslant n$, the distance is given by

$$d_u(X, Y) = \sum_{i=0}^{m} w_i \alpha_i$$

where $w_i = 2^{m-i}/(2^m - 1)$ and $\alpha_i = 0$ if $x_i = y_i$ (case-insensitive string equality) or 1 otherwise. That is, d_u is a weighted mean of α_i. Note that we consider most relevant the right-most part of the domain name.

- $d_{lev}(x, y)$: The normalized Levenshtein or edit distance between two strings x, y. The distance calculates the minimum number of edits (insertion, deletion, or substitution) needed to convert one string into the other. The value is then normalized by the maximum length of the strings.
- $d_\phi(\phi_i, \phi_j)$: Distance between two query strings (the terms introduced by the user). The distance is computed as:

$$d_\phi(\phi_i, \phi_j) = \frac{1}{3}\left(2 \cdot \sqrt{\left(\overline{|\phi_i|} - \overline{|\phi_j|}\right)^2} + d_{\mathscr{H}}(\phi_i, \phi_j)\right) \tag{1}$$

where $d_{\mathscr{H}}$ is the Hausdorff distance defined in the metric space (μ, d_{lev}), where μ is the set of all words μ_i. Each query is seen as a set of words, $\phi_i = \{\mu_1^i, \mu_2^i, \ldots\}$, and the edit distance is used to compare the words. Thus

$$d_{\mathscr{H}}(\phi_1, \phi_2) = \max(I_{\mathscr{H}}(\phi_1, \phi_2), I_{\mathscr{H}}(\phi_2, \phi_1)) \tag{2}$$

where

$$I_{\mathscr{H}}(\phi_1, \phi_2) = \max_{\mu_i \in \phi_1} \min_{\mu_j \in \phi_2} d_{lev}(\mu_i, \mu_j)$$

Note that d_ϕ considers the similarity of the words between the query strings relying in the edit distance, but also takes into account the size of the query in number of words, something that the Hausdorff distance does not measure.

- $d_\varphi(\varphi_i, \varphi_j)$: Distance between two single queries of the form $\varphi_i = (t_i, r_i, u_i, \phi_i)$, as a mean of the corresponding distances:

$$d_\varphi(\varphi_1, \varphi_2) = \frac{1}{6}\left(d_t(t_1, t_2), d_{euclid}(r_1, r_2), d_u(u_1, u_2), 3 \cdot d_\phi(\phi_1, \phi_2)\right) \tag{3}$$

Given the previous distance functions, the distance between two users is calculated as:

$$d(q(id_1), q(id_2)) = \frac{1}{2}\left(\sqrt{\left(\overline{|\varphi^1|} - \overline{|\varphi^2|}\right)^2} + d_{\mathscr{H}}(\varphi^1, \varphi^2)\right) \tag{4}$$

where $d_{\mathscr{H}}$ is the Hausdorff distance in the metric space (φ, d_φ), where φ is the set of all queries φ^i.

Note that we consider the number of queries of both users and the similarity of the set of queries between the users. The purpose of the distance is to form clusters of similar users. That is, users with a similar profile of search queries.

3.2.2. Comments on the distance function

We have chosen the distance functions attempting to provide the more straightforward measure for each part. Numeric values use Euclidean based distances, which are widely used and accepted in continuous data protection methods (especially in statistical disclosure control). The distance for domain names, clearly weights the most relevant part of the domain name. This distance was successfully applied to the anonymization of access logs from Web servers (Navarro-Arribas and Torra, 2010).

The query strings distance from Eq. (1) is more elaborated. It first computes the Hausdorff distance between the set of terms of each query using the Levenshtein distance between terms (strings) as shown in Eq. (2). Since the Hausdorff distance does not take into account the number of elements in each set we have also introduced a measure of the number of terms, computing the Euclidean distance between the normalized number of terms of each query. Note also that the Hausdorff distance has more weight than the distance between number of queries (double weight). This is so, because we consider more important the similarity of the string itself than the length of the query. Moreover, when we consider the distance between single queries in Eq. (3), the distance between the query strings has more weight, prevailing over the other parts.

A similar approach is used to compute the final distance between two users in Eq. (4), where we use the Hausdorff distance between the set of queries of each users and also consider the distance between the number of queries of each user. In this case both measures have the same weight. The number of queries is in this case more relevant due to the aggregation process that will be described in Section 3.2.3. Aggregating users with very different number of queries will result in higher information loss.

Finally, it is important to remark that the objective of the distance function is to group similar users, or users with the same search profile, together. Nevertheless a very relevant part is the distance between the terms of the queries which at the end relies in the Levenshtein or edit distance. This distance only takes into account syntactic similarities and does not actually consider the semantics of the queries. Thus, our method can be seen as a user anonymization and protection at syntactic level. A semantic approach could lead other interesting results which are to be explored.

3.2.3. User query aggregation

To find the centroid of a cluster of user queries, we compute their aggregation ($\mathbb{C}$) as the aggregation of each part of the user queries:

$$\mathbb{C}(q(id_1), \ldots, q(id_k)) = \left(id', \mathbb{C}_\varphi(\varphi^1, \ldots, \varphi^k)\right)$$

where id' is a temporary identifier for the centroid that will be replaced by the original user id in the protected dataset. And the aggregation of queries $\mathbb{C}_\varphi$ is defined as:

$$\mathbb{C}_\varphi(\varphi^1,\dots\varphi^k) = \mathbb{C}_\varphi\left(\left(\varphi_1^1,\dots,\varphi_{|\varphi^1|}^1\right),\dots,\left(\varphi_1^k,\dots,\varphi_{|\varphi^k|}^k\right)\right) = \varphi^* = \left(\varphi_1^*,\dots,\varphi_{|\varphi^*|}^*\right)$$

The centroid φ^* is composed of queries from the cluster queries φ^i for $i = 1 \dots k$. For each original query vector φ^i, that is, all queries from user i, we pick a sub-vector $\varphi^{*,i}$ of queries such that:

$$|\varphi^{*,i}| = \frac{|\varphi^*| \cdot |\varphi^i|}{\sum_{j=1}^k |\varphi^j|}$$

These queries, $\varphi^{*,i} = \left(\varphi_1^{*,i},\dots,\varphi_{|\varphi^{*,i}|}^{*,i}\right)$, are such that preserve the frequency of query strings from the original query φ^i. That is, in a more formal way, given a frequency function f on query strings, we require that,

$$f\left(\varphi_q^{*,i}\right) \simeq f(\varphi^i)$$

where

$$f\left(\varphi_q^{*,i}\right) = \frac{\left|\left\{\varphi | \varphi \doteq \varphi_q^{*,i} \quad \text{and} \quad \varphi \in \varphi^{*,i}\right\}\right|}{|\varphi^{*,i}|}$$

where $\varphi_i \doteq \varphi_j$ if the query string of both queries are equal, that is, if and only if $\phi_i = \phi_j$.

The other parts of the query are aggregated by using the arithmetic mean for the rank and the timestamp, and generalizing the URL to the right-most common part (sub-domain).

4. Evaluation

To evaluate our proposal we have tested the microaggregation of real data from the AOL logs released in 2006, which corresponds to the queries performed by 650 000 users over three months. For our tests, we randomly select 1000 users from the logs, which correspond to 55 666 lines of query logs.

In the following sections we measure the privacy achieved by our method, and the utility of the protected data. We also evaluate the utility of the protected data in data mining processes, and provide an analysis of the frequency of queries and words.

4.1. Profile exposure level

For each user id we have her original set of queries φ and the corresponding protected ones φ', which have been protected by means of our microaggregation method. Note that φ and φ' can be seen as two random variables, which can take so many values as different queries they have and with probability proportional to the number of repetitions.

In order to verify that our method protects the users' queries obtaining k-anonymity we have used the *Profile Exposure Level* (*PEL*) (Erola et al., submitted for publication) that is defined as follows:

$$PEL = \frac{I(\varphi,\varphi')}{H(\varphi)} \cdot 100$$

where $H(\varphi)$ is the entropy of the original set of queries, and $I(\varphi, \varphi')$ is the mutual information between φ and φ'.

PEL measures the percentage of the user information that is exposed when φ' is disclosed. Thus the user information is calculated as the entropy of φ, and the mutual information gives a measure of the information that φ' provides about φ, i.e. when φ' is known, how does it reduces the uncertainty about φ. So, if we divide the $I(\varphi, \varphi')$ by $H(\varphi)$ we obtain the percentage of information that attackers can deduce of φ by means of φ'. In order to protect the users' privacy, the percentage should be as low as possible.

We have microaggregated the 1000 users from the AOL logs for $k \in \{2,\dots,50\}$. Then we have computed the *PEL* for each user and value of k.

Fig. 3 shows for $k = \{2,\dots,10\}$ the theoretical level of privacy, i.e. the k-anonimity, and the average of the *PEL* obtained. Thus, we can see that our method offers k-anonymity, since the theoretical and the *PEL* obtained experimentally are very close.

4.2. Information loss

The first usability measure for categorical data was presented in Domingo-Ferrer et al. (2001). The authors proposed an entropy-based measure to evaluate the information loss in SDC. In the same line, Lixia and Jianmin (2009) proposed to measure the information loss as:

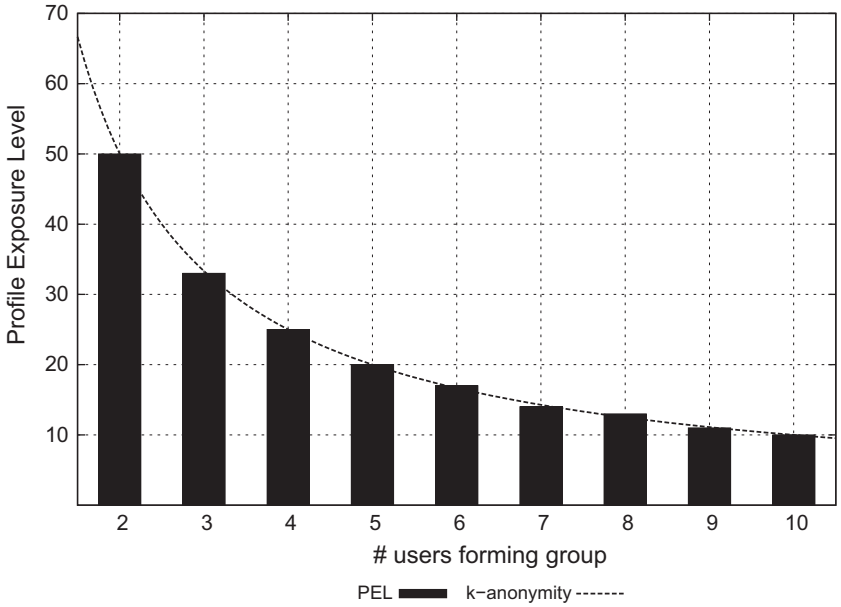

Fig. 3. The average *PEL* for $k \in \{2, \ldots, 10\}$.

$$ILR = \frac{|\text{original\_entropy} - \text{new\_entropy}|}{\text{original\_entropy}} \cdot 100$$

We have used the *ILR* (Information Loss Ratio) to evaluate the utility of our proposal with the same files obtained previously when we calculated the *PEL*. Thus, we have 1000 users and their original queries, and for every $k \in \{2, \ldots, 50\}$ the protected queries for every user. Fig. 4 shows the data utility broke down according to the parameter k of the microaggregation. It can be observed that when the k increases (thus, the number of user per cluster increases), the information loss of the user also grows. Note that minor fluctiations are due to small and expected perturbations in the microaggregation process.

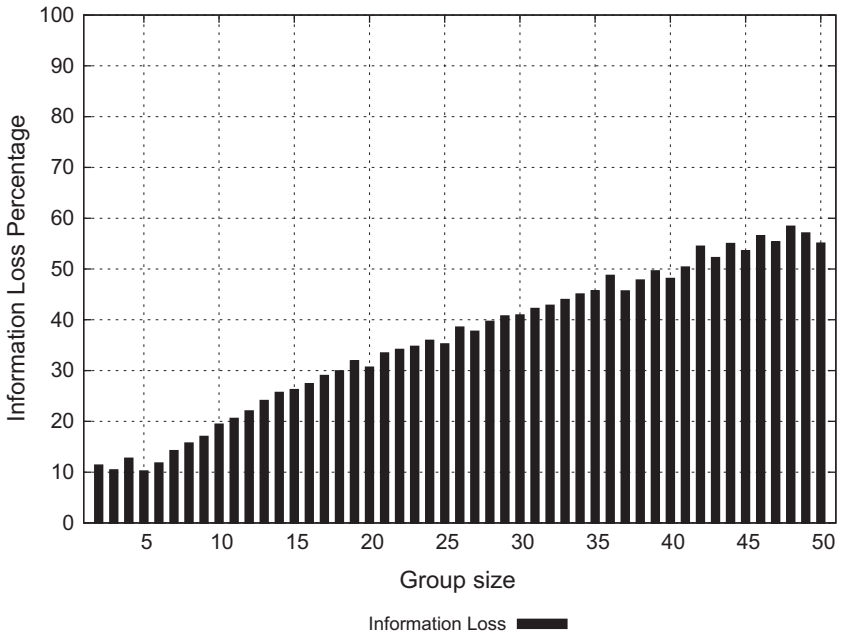

Fig. 4. The average *ILR* for $k \in \{2, \ldots, 50\}$.

4.3. On the relation of the PEL and IRL

The *PEL* and the *ILR* help us to establish a trade-off between privacy and usefulness of the data (information loss).

Using a greater k we have a high information loss (*ILR*), as we can see in Fig. 4. For $k = 35$ we loose the 50% of the users' information, and for $k = 3$ we loose only the 10%. The minimum *ILR* provides the most utility data, so we should select the smallest values of k.

Nonetheless, as it is shown in Fig. 3 a greater k offers more privacy to the users, i.e. their profiles are less exposed. For instance, when $k = 2$, 50% of the user's profile is exposed, and for $k = 10$ only the 10% is exposed. From the privacy point of view a greater k would be recommendable to protect the privacy of the users. However, we consider that a user's profile has enough protection if the *PEL* is at least of 40%, i.e. $k = 3$ (see Erola et al. (submitted for publication)).

Thus, we can conclude that the optimal k for the microaggregation is $k = 3$, because we obtain a reasonable privacy level (*PEL*) and a low information loss (*ILR*).

Note also that the number of records for each user is not very relevant, since all measures and operations are based on percentages. Obviously, users with extremely low number of queries (one or two) will lose a lot of information (close to 100%) in the protected version. Nevertheless this won't affect to the overal results in normal situations (recall that we are dealing with real data from the AOL search engine, where the number of queries per user is relatively large (Pass et al., 2006)).

4.4. Utility in data mining

Query logs are normally used in data mining processes for their analysis. To evaluate the utility of our protection method in data mining we have considered clustering as a generic data minig process. There are several data mining techniques, from which clustering is one of the most popular (Baeza-Yates et al., 2007; Beeferman and Berger, 2000; Beitzel et al., 2007). Although normally the clustering of query logs is performed with some customized and more elaborated clustering, we show our results on a simple clustering just to give a generic idea.

We have compared the clustering of protected data with the clustering of the original data. We have used the k-means algorithm to cluster user query logs relaying in the distance and aggregation functions described in Sections 3.2.1 and 3.2.3.

To compare the clusters obtained in the original data and the protected data, we have used two different well known indexes: the Jaccard index and the Rand index.

We denote the partition of the original data as Π, and the partition of the protected data as Π'. Let $\Pi = \{\pi_i, \ldots, \pi_n\}$ and $\Pi' = \{\pi'_1, \ldots, \pi'_n\}$ (both partitions have the same number of clusters). We define r, s, t, and u as the number of pairs of elements (a, b) such that:

- r: a and b are in the same cluster in Π and Π'.
- s: a and b are in the same cluster in Π but not in Π'.
- t: a and b are in the same cluster in Π' but not in Π.
- u: a and b are in different clusters in Π and Π'.

Then the indexes are defined as:

Jaccard index

$$JI(\Pi, \Pi') = \frac{r}{r + s + t}$$

Rand index

$$RI(\Pi, \Pi') = \frac{r + u}{r + s + t + u}$$

Fig. 5 shows the Jaccard and Rand indexes comparing the original data with the data protected with $k \in \{3, 5, 10, 20, 30, 50\}$, using the k-means algorithm with $\kappa = 2 \ldots 500$. We use κ to denote the k parameter of the k-means algorithm, which corresponds to the number of clusters. As the number of clusters increases, the indexes are closer to 1, meaning that both partitions are very similar. Regardless of the k used in the microaggregation process, we obtain similar results for both indexes.

More interesting is to see the differences between the indexes as the microaggregation k is incremented. As noted in Sections 4.1 and 4.2, as this k increments, we achieve more privacy but less utility. Given a protected dataset with $k = 3$ and another with $k = 50$, Fig. 6 shows the difference between the clusters of the two datasets, as the k-means κ increases. The straight line denotes de difference between the Jaccard index of both datasets, and the doted line the difference between the Rand index.

Although the values for the indexes in Fig. 5 seem similar. We can see in Fig. 6 that the differences between the same index but for the two different values of k (3 and 50) decrements as the k-means κ increases. This means that if we are going

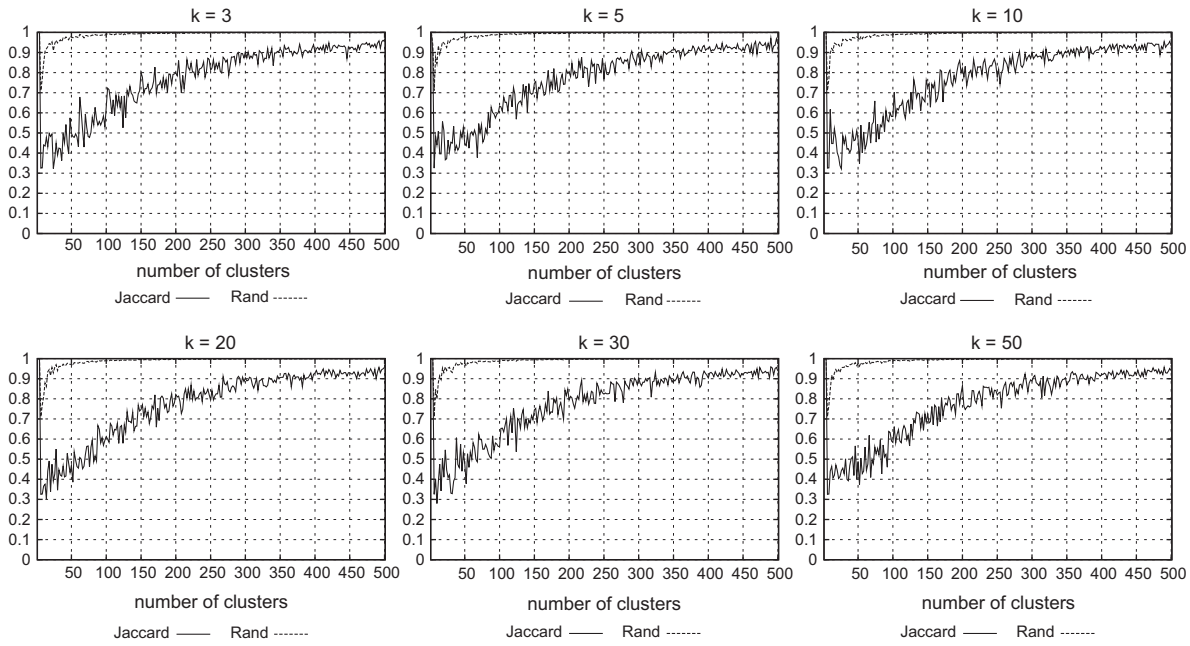

Fig. 5. Rand and Jaccard indexes for aggregated data with $k \in \{3, 5, 10, 20, 30, 50\}$.

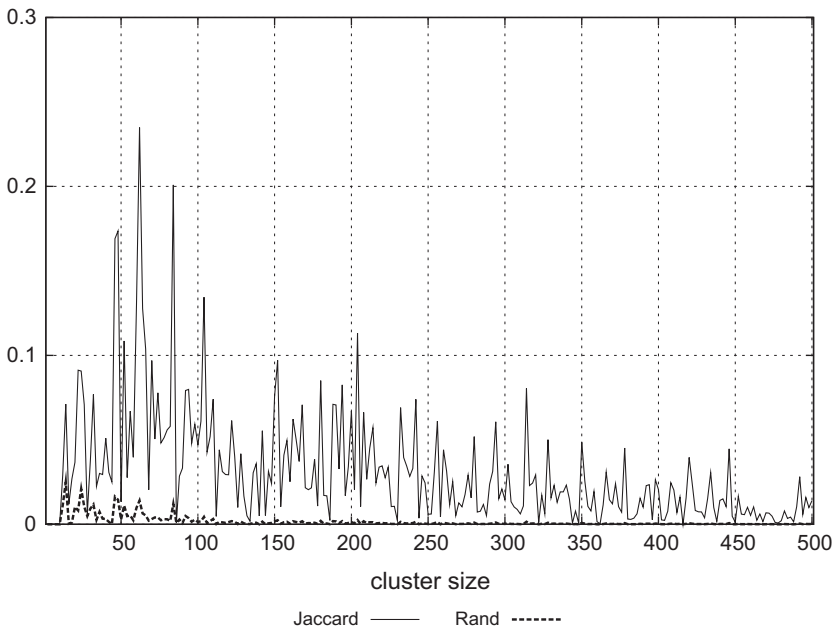

Fig. 6. Difference of the Jaccard and Rand indexes for data microaggregated with $k = 3$ and $k = 50$.

to cluster data in the data mining process with relatively big κ, we can use a relatively big k in the microaggregation. That is, we can provide high privacy protection to the data while preserving the partitions of the clustering.

Note that we are clustering users (or user profiles) using the same user distance used in the microaggregation process. This anticipates that the relatively good results were expected from how the protected log is generated. We think that this is the main reason that makes microaggregation a good protection technique for privacy-preserving data mining when a distance-based clustering is used in the data mining process. The example described here is just an exemplification of this statement. Some particular data mining application where our protection method will provide good results are in fact those making use of some clustering technique: categorization, classification, etc.

4.5. Frequency analysis

We have also analyzed the frequency of queries in the protected data. Fig. 7 shows the frequency of the 10 most popular queries in the original data, and their evolution in the protected data as the microaggregation k increases. Note that $k = 1$ in the figure corresponds to the original data.

Although there are variations in the frequency, they are very low. Most relevant is that in all the protected data the same 10 queries are the 10 most popular, with some minor exceptions.

If we take a look to the frequency of single words, excluding common stop words, we see a similar result. As shown in Fig. 8, frequency of words is relatively preserved. There are some concrete cases where a given word disappears with big values of k, but the overall result is quite good.

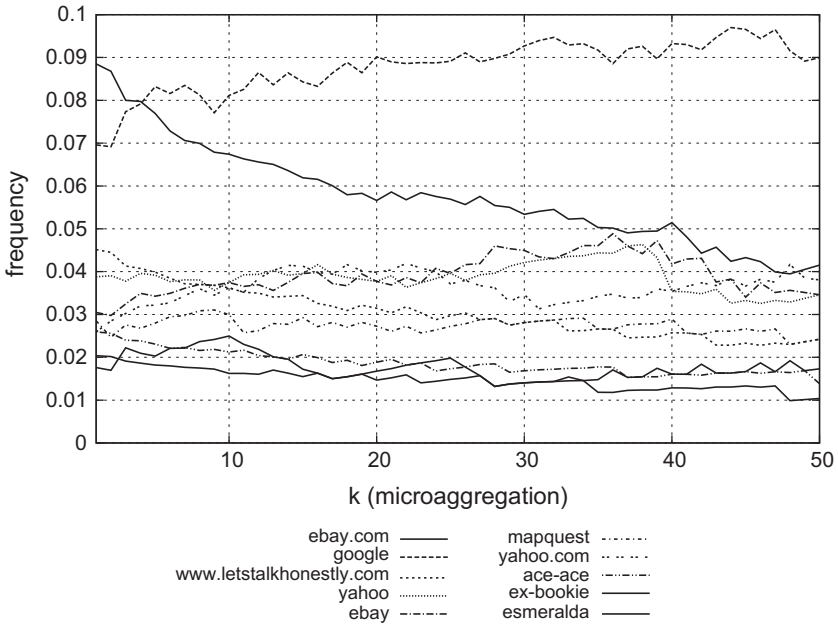

Fig. 7. Query frequency analysis.

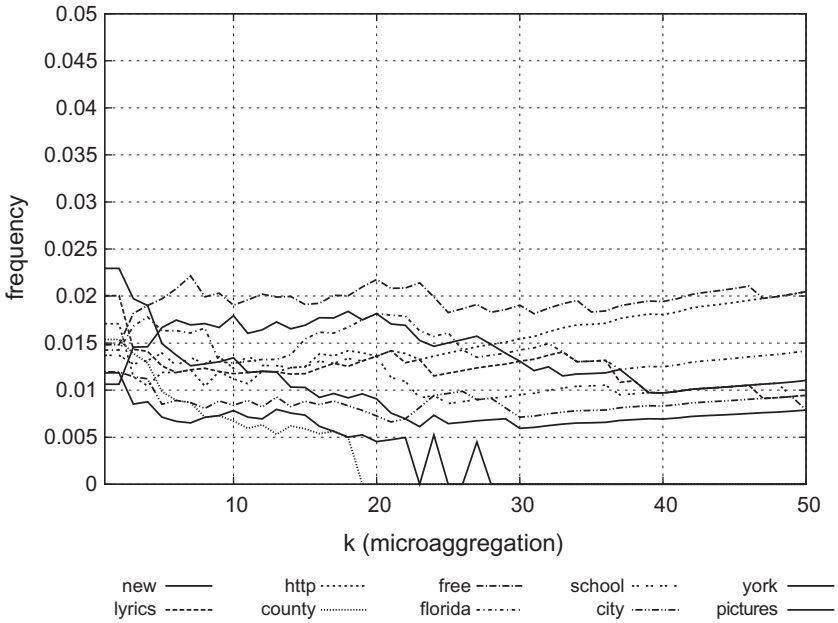

Fig. 8. Word frequency.

Normally this kind of studies are performed with much bigger datasets (note that the most frequent word appears less than a 2.5%). The fact that our method preserves the frequencies with relatively smaller sets, points to a even better preservation in bigger datasets.

5. Conclusions

In this paper we have provided a technique to ensure user k-anonymity in query logs by microaggregation. The query logs can be protected to prevent the disclousure of sensitive information, and the reidentification of users, in order to make them public. The main advantage of our proposal with respect to previous ones, is that it provides user k-anonymity. That is, from the protected data set, there are k indistinguishable users. Any attack which attempts to re-identify a user, will end up with a set of k potential users, so the probability of reidentification is directly related to the size of the parameter k used in the microaggregation.

To provide microaggregation at user level, we have defined a new user distance and aggregation operator. The user aggregation described in Section 3.2.3 was designed in order to be as computationally efficient as possible. Note that the most important part is the aggregation of the queries since it is the information that will be more valuable in future analysis. Note also that queries are aggregated separately. An alternative could be to actually mix the terms of queries from different users to end up with new queries that somehow summarize all the users' queries. We opted for the fist approach given the complexity that the second one imposes, and also because it already produced satisfactory as show in Section 4.

The other parts of the query are aggregated with the most common aggregation operators used in data privacy and statistical disclosure control. Other operators could easily be used, if required.

As always happens with statistical disclosure techniques, there is a trade-off between privacy and usability. We have shown that our proposal, besides providing k-anonymity, maintains to some extend the information of the original logs. Both, in terms of the information regarding the users, and in the use of the data for data mining. Our proposal can be seen as an efficient and relatively simple method to protect query logs, when compared to existing solutions, to ensure a high degree of anonymity and privacy.

Acknowledgments

Partial support by the Spanish MICINN (Projects eAEGIS TSI2007-65406-C03-02, TSI2007-65406-C03-01, ARES-CONSOLIDER INGENIO 2010 CSD2007-00004, Audit Transparency Voting Process PT-430000-2010-31, and N-KHRONOUS TIN2010-15764), the Spanish Ministry of Industry, Commerce and Tourism (Project TSI-020100-2009-720 and SeCloud TSI-020302-2010-153), and the Government of Catalonia (Grant 2009 SGR 1135) is acknowledged. G. Navarro-Arribas enjoys a Juan de la Cierva Grant (JCI-2008-3162) from the Spanish MICINN.

References

Adar, E. (2007). User 4xxxxx9: Anonymizing query logs. In *Query logs workshop in 16 international world wide web conference*.

Baeza-Yates, R., Hurtado, C., & Mendoza, M. (2007). Improving search engines by query clustering. *Journal of the American Society for Information Science and Technology, 58*(12), 1793–1804.

Barbaro, M., & Zeller, T. (2006). A face is exposed for AOL Searcher No. 4417749. *The New York Times*.

Beeferman, D., & Berger, A. (2000). Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 407–416).

Beitzel, S., Jensen, E., Chowdhury, A., Frieder, O., & Grossman, D. (2007). Temporal analysis of a very large topically categorized web query log. *Journal of the American Society for Information Science and Technology, 58*(2), 166–178.

Cooper, A. (2008). A survey of query log privacy-enhancing techniques from a policy perspective. *ACM Transactions on the Web, 2*(4), 1–27.

Defays, D., & Nanopoulos, P. (1993). Panels of enterprises and confidentiality: the small aggregates method. In *Proceedings of 92nd symposium on design and analysis of longitudinal surveys* (pp. 195–204). Statistics Canada.

Domingo-Ferrer, J., Mateo-Sanz, J., & Torra, V. (2001). Comparing sdc methods for microdata on the basis of information loss and disclosure. In *Proceedings of ETK-NTTS 2001* (pp. 807–826). Luxemburg: Eurostat.

Domingo-Ferrer, J., & Mateo-Sanz, J. (2002). Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and Data Engineering, 14*(1), 189–201.

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

EFF (2009). *AOL's massive data leak*. Electronic Frontier Foundation. <http://w2.eff.org/Privacy/AOL/>.

Erola, A., Castellà-Roca, J., Viejo, A., & Mateo-Sanz, J. (submitted for publication). Exploiting social networks to provide privacy in personalized web search.

Hong, Y., He, X., Vaidya, J., Adam, N., & Atluri, V. (2009). Effective anonymization of query logs. In *Proceeding of the 18th ACM conference on information and knowledge management (CIKM'09)* (pp. 1465–1468).

Jansen, B., & Spink, A. (2005). An analysis of web searching by European AlltheWeb.com users. *Information Processing & Management, 41*(2), 361–381.

Jansen, B., Spink, A., & Pedersen, J. (2005). A temporal comparison of AltaVista web searching: Research articles. *Journal of the American Society for Information Science and Technology, 56*(6), 559–570.

Jansen, B., Spink, A., & Saracevic, T. (2000). Real life, real users, and real needs: A study and analysis of user queries on the web. *Information Processing & Management, 36*(2), 207–227.

Jones, R., Kumar, R., Pang, B., & Tomkins, A. (2007). I know what you did last summer: Query logs and user privacy. In *Proceedings of the sixteenth ACM conference on conference on information and knowledge management* (pp. 909–914). ACM.

Korolova, A., Kenthapadi, K., Mishra, N., & Ntoulas, A. (2009). Releasing search queries and clicks privately. In *Proceedings of the 18th international conference on world wide web (WWW'09)* (pp. 171–180).

Kumar, R., Novak, J., Pang, B., & Tomkins, A. (2007). On anonymizing query logs via token-based hashing. In *16th International world wide web conference* (pp. 629–638).

Lixia, W., & Jianmin, H. (2009). Utility evaluation of *k*-anonymous data by microaggregation. In *International conference on communication system, networks and applications, 2009 ICCSNA* (Vol. 4, pp. 381–384).
Mills, E. (2006). AOL sued over web search data release. *CNET News* <http://news.cnet.com/8301-10784_3-6119218-7.html>.
Navarro-Arribas, G., & Torra, V. (2009). Tree-based microaggregation for the anonymization of search logs. In *Proceedings of the 2009 IEEE/WIC/ACM international joint conference on web intelligence and intelligent agent technology (WI-IAT'09)* (pp. 155–158).
Navarro-Arribas, G., & Torra, V. (2010). Privacy-preserving data-mining through micro-aggregation for web-based e-commerce. *Internet Research, 20*(3), 366–384.
Oganian, A., & Domingo-Ferrer, J. (2001). On the complexity of optimal microaggregation for statistical disclosure control. *Statistical Journal of the United Nations Economic Commision for Europe, 18*(4), 345–353.
Pass, G., Chowdhury, A., & Torgeson, C. (2006). A picture of search. In *Proceedings of the 1st international conference on scalable information systems* (p. 1).
Poblete, B., Spiliopoulou, M., & Baeza-Yates, R. (2010). Privacy-preserving query log mining for business confidentiality protection. *ACM Transactions on the Web (TWEB), 4*(3), 1–26.
Samarati, P. (2001). Protecting respondents identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering, 13*(6), 1010–1027.
Sweeney, L. (2002). *k*-Anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10*(5), 557–570.
Torra, V. (2004). Microaggregation for categorical variables: A median based approach. In *Proceedings of privacy in statistical databases (PSD 2004). Lecture notes in computer science* (Vol. 3050, pp. 162–174).
Torra, V. (2008). Constrained microaggregation: Adding constraints for data editing. *Transactions on Data Privacy, 1*(2), 86–104.
Xiong, L., & Agichtein, E. (2007). Towards privacy-preserving query log publishing. In *Query log analysis: Social and technological challenges. Workshop in 16th international world wide web conference*.