Arnau Erola Cañellas

CONTRIBUTIONS TO PRIVACY IN WEB SEARCH ENGINES

Ph.D. Dissertation

Directed by Dr. Jordi Castellà Roca

Departament d'Enginyeria Informàtica i Matemàtiques



Universitat Rovira i Virgili

Tarragona
2013



I STATE that the present study, entitled *Contributions to privacy in web search engines*, presented by Arnau Erola Cañellas for the degree of Doctor of Philosophy in Computer Science, has been carried out under my supervision at the Departament d'Enginyeria Informàtica i Matemàtiques of this university, and that it fulfills all the requirements to receive the European Doctorate Distinction.

Tarragona, June 10th, 2013

The doctoral thesis supervisor

Dr. Jordi Castellà Roca

Abstract

Web search engines (WSEs) collect and store information about their users in order to tailor their services better to their users' needs. Nevertheless, while receiving a personalized attention, users lose control over their own data. Search logs can disclose sensitive information and the real identities of users, thus creating serious risks of privacy breaches. Privacy preserving techniques seek to limit these risks by modifying the data. Although privacy is preserved, the data utility is reduced in a consequence of the data modifications. Achieving a good trade-off between privacy and utility can be a difficult task. In the present thesis we discuss the problem of limiting privacy disclosure risks in search logs while preserving enough data utility.

The first part of this thesis focuses on the methods to prevent the gathering of information by WSEs. Since search logs are convenient in order to receive an accurate service, the aim is to provide logs that are still suitable to provide personalization. To that end, we propose a protocol which uses a social network in order to hide the queries submitted by a user. Results shows that users achieve good levels of privacy, meanwhile the response time of the protocol is acceptable.

The second part of this thesis deals with the dissemination of search logs. We propose microaggregation techniques which allow the publication of search logs while providing a high privacy protection. Aimed at minimizing the information loss, the proposals are specifically developed to deal with query logs. Evaluation shows that our proposals achieve good trade-offs between privacy and utility, outperforming the previous methods.

Acknowledgements

I would like to thank my director Jordi, whose expertise made this thesis possible. I would also like to thank my father, my mother and the rest of my family for their unconditional support. I am very grateful to Pierangela Samarati for her thoughtful comments during my stay in Crema. And last but not least, to Patricia for her patience and support when there was no sign of light at the end of the tunnel.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Contributions . . . 4

2 Preliminaries **7**
- 2.1 Types of data . . . 7
- 2.2 Statistical Disclosure Control . . . 7
- 2.3 Microdata release . . . 8
- 2.4 Microaggregation . . . 10
- 2.5 MDAV . . . 11
- 2.6 k-anonymity . . . 12

3 State of the art **15**
- 3.1 Client-side obfuscation . . . 16
  - 3.1.1 Uncollaborative schemes . . . 16
  - 3.1.2 Collaborative schemes . . . 19
- 3.2 Server-side disclosure control techniques . . . 23

4 Client-side anonymization **27**
- 4.1 Introduction . . . 27
- 4.2 Contribution . . . 28
- 4.3 Protocol for protecting the privacy of the users . . . 29
  - 4.3.1 The protocol in detail . . . 31
  - 4.3.2 Sending probability P_s . . . 33
  - 4.3.3 Query forward function Ψ_f . . . 33
  - 4.3.4 User selection function Ψ . . . 34
  - 4.3.5 Selfishness function $\Upsilon(N)$ . . . 35
    - 4.3.5.1 Coprivacy . . . 36

4.4 Evaluation . . . 37
4.5 Profile Exposure Level . . . 38
4.5.1 Mutual Information . . . 39
4.6 Usability measure . . . 42
4.7 Simulations . . . 43
4.7.1 Tests . . . 44
4.7.2 Privacy . . . 44
4.7.2.1 Simulation results from scenarios without selfish users . . . 45
4.7.2.2 Simulation results from scenarios with selfish users . . . 48
4.7.3 Usability . . . 49
4.8 Conclusions . . . 51
4.8.1 Publications . . . 53

5 Server-side anonymization **55**
5.1 Search logs . . . 55
5.2 Search logs release . . . 56
5.3 Microaggregation of search logs . . . 58
5.3.1 Microaggregation of query logs . . . 59
5.3.1.1 Distance and aggregation of query logs . . . 59
5.3.1.2 User query distance . . . 60
5.3.1.3 Comments on the distance function . . . 62
5.3.1.4 User query aggregation . . . 63
5.3.2 Evaluation . . . 64
5.3.2.1 Profile Exposure Level . . . 65
5.3.2.2 Information loss . . . 66
5.3.2.3 On the relation of PEL and IRL . . . 67
5.3.2.4 Utility in data mining . . . 68
5.3.2.5 Frequency analysis . . . 72
5.3.3 Conclusions . . . 73
5.3.3.1 Publications . . . 74
5.4 Semantic microaggregation of search logs . . . 75
5.4.1 Towards a semantic interpretation of query logs . . . 76
5.4.1.1 ODP . . . 76
5.4.1.2 An ODP similarity measure . . . 77
5.4.2 ODP-based microaggregation of query logs . . . 78

5.4.3 Data preparation . . . 78
5.4.3.1 Partition . . . 79
5.4.3.2 Aggregation . . . 83
5.4.4 Evaluation . . . 84
5.4.4.1 Usefulness Evaluation Method . . . 85
5.4.4.2 Results . . . 86
5.4.4.3 ODP levels . . . 86
5.4.4.4 SRP vs. ODP-categories . . . 87
5.4.4.5 Computational cost vs. ODP-categories . . . 89
5.4.5 Conclusions . . . 90
5.4.5.1 Publications . . . 91
5.5 Semantic microaggregation optimization . . . 92
5.5.1 Query anonymization method . . . 92
5.5.1.1 Query processing and conceptual mapping . . . 93
Query processing . . . 94
Conceptual mapping . . . 95
Knowledge base . . . 95
5.5.1.2 Semantic data partition . . . 96
Comparing queries . . . 97
User query log comparison . . . 98
Centroid calculus . . . 100
5.5.1.3 Query log anonymization . . . 101
5.5.2 Evaluation . . . 104
5.5.2.1 Evaluation measures . . . 105
5.5.2.2 Comparison . . . 107
5.5.2.3 Results . . . 108
5.5.3 Conclusions . . . 121
5.5.3.1 Publications . . . 121

6 Conclusions **123**
6.1 Contributions . . . 123
6.2 Future work . . . 125
6.3 Publications . . . 126

Bibliography **129**

Chapter 1

Introduction

1.1 Motivation

In recent years, the problem of properly protecting personal information has received a lot of attention due to the vast amount of information that is collected by Internet companies. Thanks to the evolution of information technologiess, every transaction performed by an individual is stored, analyzed or even shared or disseminated. The queries submitted to a Web Search Engine (WSE) are an example.

WSEs have become one of the most successful services on Internet, responding to the demand for information facilities and services of the Information Society. By providing an easy way to access the Web, WSEs receive several hundred million queries each day. For example, during 2011, the WSE Google received near 5 000 million queries per day [42]. All these transactions are collected and stored as search or query logs.

The reasons why WSEs maintain the search logs [21] can be classified into the three following categories:

- **Personalization.** WSEs provide their users result pages related to

their searches (web pages containing links to the resulting data) in the web search process. From the huge amount of results, often thousands, only some links are relevant to the user's needs; meanwhile the other ones are irrelevant. The study presented in [49] states that a 68% of the users of WSEs click a search result within the first page of results. Even more relevant is the fact that a 92% of the users click a result within the first three pages of search results. Within WSEs business scenario, providing accurate results in order to further gain clients and developing loyalty is crucial. Hence, in order to satisfy the user's needs, links should be ranked accoding to their relevance to the user, i.e. relevant links to the user should be placed in the first positions of the returned results.

Nevertheless, ranking the search results according to the users' needs is a real challenge. Queries are usually composed by few words which cannot refer to specific information or can contain ambiguous terms (terms that have different meanings). For example, the word *Java* can refer to the Java programming language or to the island of Java. Without any other knowledge, it is impossible to know which meaning *Java* has in the query. Thus, WSEs have to *disambiguate* the queries by identifying the correct sense when they have different ones. This process requires the knowledge of: (i) the interests of the user; and (ii) the query context. For example, if a certain user is interested in computer science, the WSE will assume she is referring to the Java programming language. The disambiguation process is used by schemes of Personalized Search (PS) [81, 90, 94, 110].

By analyzing the stored logs, WSEs are able to extract these users' interests, which help to improve the accuracy of the search results and to personalize the advertisements. The latter is very important in the bussines model of the WSE as it provides high revenues. For instance, in 2012, Google had a revenue of 43 686 million dollars from advertisement [43].

- **Improving search.** Past searches are an invaluable resource to im-

prove the quality of search results. By knowing the frequencies of most formulated queries and most selected results, WSEs are able to improve the ranking algorithms [2] and to suggest reformulated queries that can add specificity to the user's initial query [56]. Moreoever, query logs provide information on how people employ language, which can be useful to improve features such as query spelling corrections or to recognize when a user is posing a question.

- **Sharing data.** Aside from the information retrieval role, WSEs can act as an information source for third parties. Query logs are of great interest for researchers [6] to study and test new Information Retrieval (IR) algorithms, to learn about users information needs and query formulation approaches [57], and also to investigate the use of language in queries [100]. Marketing companies can exploit query logs to characterize general user profiles, behavior and search habits [82, 12], to have competitive advantage over their counterparts, to improve keyword advertising campaigns [57] and to extract market tendencies and trending topics [45]. Moreover, governmental authorities can force the WSEs to disclose query logs of specific users or groups [111, 98].

Query logs cleary contain valuable information although in most cases, logs also contain personal information of the users [95]. These are some examples of this situation: (i) if a certain user has searched for a certain place, it can be inferred that she lives there; (ii) if she searches a certain disease, it can be deduced that she (or someone close to her) suffers that disease; and (iii) the user can make a vanity query in which she searches for her own name [59, 95]. Moreover, a query can itself contain information about several issues. For instance, a simple query as *Drug Clinic Portland* is probably disclosing that the user lives in *Portland* and has some problems with *drugs*.

Not surprisingly, privacy disclosure risks arise when the WSEs want to disseminate query logs. These risks can be classified into two categories: (i) identity disclosure when a user is re-identifyed; (ii) attribute disclosure when information about a user is retrieved. Hence, the main threat exposed by a

search log is to be able to link user queries with a real identity. In order to avoid these risks, WSEs should protect the data prior to their dissemination.

The protection of query logs implies some data modifications which limit the privacy disclosure risks although at the same time they reduce the data reliability. The problem is that achieving a desirable degree of privacy in search logs is not easy and presents an important trade-off between the privacy and utility of the data. While the utility is conditional to the ability of performing a latter analysis with the data, privacy is conditional to the ability of disclosing information about the users, although it is done with the help of external information. Several authors [1, 55, 82] have discussed about how the combination of the modified data can disclose enough information to re-identify some users. Hence, although query logs can be protected prior to their publication, there is no absolute guarantee of anonymity.

There is at least one well-known case of privacy disclosure in search logs which shows that achieving a desirable degree of privacy in search logs is not easy. AOL Research, in an attempt to help the information retrieval research community, released in 2006 around 21 million queries performed by 650 000 costumers over a period of 3 months. Although the records were previously de-identified, two reporters of the New York Times were able to locate AOL customer no. 4417749 as Thelma Arnold, a 62 years old widow living in Lilburn [7], from the query logs, and quite a lot of other sensitive information was exposed. The case ended up not only with an important damage to AOL users' privacy, but also with a major damage to AOL itself, with several class action suits and complaints against the company [34, 71, 89].

1.2 Contributions

The release of query logs from AOL with poor protection was a mistake normally regarded as a bad initiative taken by the company. Our objective

is to provide a stronger anonymization so query logs collected by search engine companies do not pose a risk to the privacy of their users.

In a standard WSE scenario, the protection of query logs limiting improper disclosures can be addressed from two different points:

- **Client-side.** WSEs have no interest to give users control over the collected data because: (i) query logs helps to improve the information retrieval service and therefore the users' satisfaction; (ii) the business model of the WSEs is based on advertisements, whose efficacy relies on their personalization. Moreover, once the personal data are gathered, users can do nothing to prevent the WSE from using them for commercial purposes, putting their privacy at risk. The AOL case is an example of this.

  Accordingly, users attempt to obfuscate the information that WSEs can gather about them by using some obfuscation mechanisms. In this way, users prevent that WSEs create a detailed profile about them, limiting the privacy breaches. However, the profile should contain enough reliable information if the user wants to obtain an acurate service. In addition, if the user anonymizes her own profile in a proper way, her privacy will be properly protected agains a data dissemination.

  In this sense, we propose a Peer-to-Peer (P2P) protocol that exploits social networks in order to protect the privacy of the users from the profiling mechanisms of the WSEs. In order to be usable in practice, the protocol should resist the presence of users who do not behave properly (*i.e.* adversaries) and should also offer a short response time. Due to the lack of a standard measure to evaluate the privacy protection achieved, we need to define a new one.

- **Server-side.** The WSE wants to share or outsource the collected query logs without putting the privacy of users at risk. For this reason, it anonymizes the query logs using techniques such as privacy preserving data mining (PPD) and statistical disclosure control (SDC).

> Disseminated query logs should preserve the privacy of the users, but at the same time they should be suitable to perform a posterior analysis. In order to do so, we propose to apply microaggregation techniques. As microaggregation offers k-anonymity privacy, then the objective is to minimize the information loss. To that end, appropriate similarity and aggregation functions to query logs should be proposed.

The rest of the present thesis is organized as follows. Chapter 2 gives some basic definitions. Chapter 3 reviews the state of the art on WSEs users. A proposal to prevent the information gathering by the WSE is presented in Chapter 4. The following chapter (Chapter 5) presents three new microaggregation techniques to disseminate query logs. Finally, conclusions and future work are drawn in Chapter 6.

Chapter 2

Preliminaries

2.1 Types of data

Data sets can contain information on individuals (microdata) of aggregated information (macrodata). For instance, a telephone survey which contains the answers of respondents is a microdata set, while a list of average wages in the European countries is a macrodata set.

2.2 Statistical Disclosure Control

Statistical Disclosure Control (SDC) techniques are needed to limit the risks of information inference in microdata. SDC techniques seek to disseminate statistical information in such a way that no confidential information about a specific individual can be inferred. To that end, data are modified to provide sufficient protection while trying to keep the information loss at minimum.

2.3 Microdata release

Internet services collect and store information about their users in order to tailor their services better to their users' needs. Nowadays, these data are usually released in form of microdata because they have the advantage to be more flexible than the aggregated macrodata.

A microadata set is usually represented in a tabular form, where each record contains attributes (data) of an individual respondent (user). These attributes can be either numerical or categorical, and they are usually classified in the following categories, which are not mutually exclusive, depending on their content:

- **Identifiers**. Attributes which unambiguously identify the respondent. For example, the passport number, the credit card number, the full name, etc. Such data are usually removed because they are not valuable for the advanced functionalities and constitute a high privacy threat.
- **Quasi-identifiers**. Attributes that cannot identify a user. However, in combination or with the help of external information (for instance the information of public databases), they can re-identify the respondent. The zipcode, age, first name, etc. are some examples. For instance, zipcode and age are probably not enough to identify the respondent, although she can be identified if her first name is unusual in this zone.
- **Confidential attributes**. The attributes contain sensitive information of users. Examples are religion, illnesses, investments, etc.
- **Non-confidential attributes** Attributes that do not contain sensitive information. Country names and favorite sports can be some examples. However, they can also constitute an identity disclosure risk because in combination they can form quasi-identifiers.

Privacy disclosure risks of microdata can be classified into two categories: identity disclosure and attribute disclosure. The identity disclosure is produced when an attacker detects the presence of an individual; meanwhile, the attribute disclosure is produced when an attacker retrieves sensitive information about a respondent.

In the literature we can find several proposals which anonymize/de-identify microdata in order to minimize the disclosure risks of the disseminated data [25]. They usually perform transformations over potentially identifying data, reducing the level of specificity (e.g. zip code can be transformed from 12345 to 123XX; specific ages can be grouped in a range, such as 15-25; a job name can be generalized from plastic surgeon to doctor, etc.) and/or making groups of individuals indistinguishable (e.g. the set of ages of several individuals 15, 17, 20, 21 are replaced by their average 18). These transformations distort input data, making them less specific or detailed, a dimension commonly referred and quantified as Information Loss (IL). Since the utility of anonymized data is closely related to the amount of information loss, anonymization methods should balance the trade-off between information loss and disclosure risk [25].

In order to achieve this goal, within the Statistical Disclosure Control (SDC) community, authors have proposed many techniques to anonymize structured databases, which consist of records with several univalued attributes, while minimizing the information loss [25, 54, 67, 69]. In these methods, identifier attributes are removed, and quasi-identifier attributes are anonymized by transforming the data. Confidential attributes can remain unaltered, thus enabling posterior analyses. Anonymization of quasi-identifier attributes is usually done so that the masked database fulfills the k-anonymity property. A database is k-anonymous if each record is indistinguishable from, at least $k-1$, other records with respect to their quasi-identifier attributes [91, 99].

2.4 Microaggregation

Microaggregation [23] is a family of statistical disclosure control techniques, 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 their quasi-identifier values by the values the centroid of the cluster has. The constant k is a parameter of the method that controls the level of privacy. The larger the k, the more privacy the protected data will have.

Microaggregation was originally defined for numerical attributes [23], but later extended to other domains; e.g., to categorical data in [104] (see also [33]) and in constrained domains in [106].

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.

From a formal point of view, microaggregation can be defined as an optimization problem with some constraints. We give a formalization below using u_{ij} to describe the partition of the records in the sensitive data set X. That is, $u_{ij} = 1$ if record 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:

$$
\begin{aligned}
\text{Minimize} \quad & SSE = \sum_{i=1}^{g} \sum_{j=1}^{n} u_{ij} (d(x_j, v_i))^2 \\
\text{Subject to} \quad & \sum_{i=1}^{g} u_{ij} = 1 \text{ for all } j = 1, \ldots, n \\
& 2k \geq \sum_{j=1}^{n} u_{ij} \geq k \text{ for all } i = 1, \ldots, g \\
& u_{ij} \in \{0, 1\}
\end{aligned}
$$

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., $v_i = \sum_{j=1}^{n} u_{ij} x_i / \sum_{j=1}^{n} u_{ij}$. However, it has been proved that finding the optimal data partition considering more than one variable at a time (multivariate microaggregation) is in general NP-Hard [78]. Hence, approximation algorithms to optimal microaggregation have been proposed.

2.5 MDAV

MDAV [30] (Maximum Distance to Average Vector) is an approximation algorithm to optimize the micoaggregation. MDAV stands out from other microaggregation methods since it is specifically designed to minimize the information loss [29, 66].

The behavior of the method is depicted in Algorithm 1. Data partition begins by calculating the centroid of the whole dataset and selecting the most distant record (x_r) to it. Then, a cluster is constructed with the k-1 least distant records to x_r. After that, the most distant record x_s to x_r is selected and a new cluster is constructed. The process is repeated until less than $2k$ records remain ungrouped. Remaining records are grouped together in a last cluster. As a result, all clusters will have k records, except for the last one, which may have from k to $2k-1$ records (considering that the input

dataset is not multiple of k). Finally, data anonymization is performed by replacing each record of each cluster by the centroid of the cluster.

MDAV, like most SDC methods, has been originally designed to deal with numerical data. As discussed in the introduction, numbers can be easily managed by means of arithmetical operators. In this case, the Euclidean distance and the arithmetic average have normally been used to compare and anonymize numerical data [33]. However, the application of MDAV to free textual data such as query logs is not straightforward, since semantically-grounded operators are needed to accurately compare and transform them. Even though some authors have applied the MDAV method to categorical data, most of them limited to terminological comparisons [33].

2.6 k-anonymity

The k-anonymity principle [91, 99] has been widely used to protect the identities of the respondents against re-identification attacks in relational databases. A protected dataset is said to satisfy k-anonymity if each combination of quasi-identifier attributes appears at least k times, i.e. each respondent is indistinguishable from at least other $k-1$ respondents.

k-anonymity bears some resemblance to the underlying principle of microaggregation. In fact, when all variables are considered at once, microaggregation is a way to implement k-anonymity.

Traditionally, k-anonymity is achieved by generalizing or suppressing quasi-identifier attributes, thus causing information loss owing to the reduction of details of the released data. While privacy protection is given by parameter k, information loss can be minimized by reducing the required generalizations and suppressions.

However, while k-anonymity offers protection against identity disclosure, attribute disclosure is still possible. As a consequence, several extensions

Algorithm 1 MDAV

Require: X: original data set, k: integer
Ensure: X': anonymized data set
$X = X'$

/*Data Partition*/
while $|X| \geq 3 \times k$ **do**
Compute centroid c_x of all records in X
Find the most distant record x_r to centroid c_x
Form a cluster in X' that contains x_r together with its $k-1$ least distant records
Remove these records from X
Find the most distant record x_s to x_r
Form a cluster in X' that contains x_s together with its $k-1$ least distant records
Remove these records from X
end while
if $|X| \geq 2 \times k$ **then**
Compute centroid c_x of all records in X
Find the most distant record x_r to centroid c_x
Form a cluster in X' that contains x_r together with its $k-1$ least distant records
Remove these records from X
end if
Form a cluster in X' with the remaining records

/*Data anonymization*/
for each cluster q in X' **do**
Compute centroid c_q of all records in q
Replace all records of q in X' by their centroid c_q
end for

of the k-anonymity principle have been proposed in the literature. Among them, l-diversity [65] and t-closeness [61] are the most important ones.

- l-diversity. When some sensitive attributes of a group of respondents which share the same quasi-identifier set of attributes are the same, this sensitive information is unprotected. To fix this drawback, l-diversity requires the presence of at least l different values for the sensitive attributes.
- t-closeness. The knowledge of the frequency distribution of sensitive attribute values can help an attacker to reduce her uncertainty about sensitive information of an individual. To fix this drawback, t-closeness requires that the sensitive value distribution of any set of respondents which share the same quasi-identifier attributes differ from the overall sensitive value distribution by a threshold t at most.

Although the extensions of k-anonymity minimize attribute disclosure, their privacy principle is the same. Hence, the proposed methods in this work address k-anonymity protection, yet they can be adapted to provide l-diversity and t-closeness.

Chapter 3

State of the art

Different ways in which WSEs retrieve the user's interests have been proposed by several authors. In [97], the authors use the browsing history. The use of click-through data is proposed in [85]. In [96, 86], two schemes which introduce the use of web communities for this purpose are presented. In [101], the authors present a client side application which stores the interests of the users. Nevertheless, the use of the queries previously submitted by users have been proved to be the best approach [96, 41]. This latter mechanism is very effective because it profiles users without their collaboration.

In this chapter we review the existing proposals to protect the privacy of the users of the WSEs. This proposals can be classified into two categories: client-side and server-side. We start with the client-side mechanisms which help users to obfuscate their profiles. Then, we present the server-side diclosure control techniques that search engines can apply in order to disseminate the data while limiting privacy disclosure risks.

3.1 Client-side obfuscation

Privacy concerns have a long history. They already existed in information retrieval from public databases. In this field, when a user submits a query, she is also exposing her interests to the database operator. Since the early 80s, several proposals to hide this personal information have emerged in order to address this situation.

Those proposals can be classified according to the existence of cooperation between the entities involved in the searching process: (i) the uncollaborative schemes where the user has got the cooperation of neither other users nor the WSE; and (ii) the collaborative schemes where the users has de coooperation of either other users or the WSE. We next summarize the different existing schemes of each type.

3.1.1 Uncollaborative schemes

Intuitively, it can be assumed that the privacy of users can be preserved by preventing the WSE from identifying the true source of the query. In this way, the WSE cannot link users with their queries and it cannot create their profiles. A trivial way to provide anonymity to users who use a WSE is to use dynamic IPs and a web browser without cookies. However, this approach has the following drawbacks:

- The renewal policy of the dynamic IP address is not controlled by the user but the network operator. This operator can always give the same IP address to the same Media Access Control (MAC) address. Nevertheless, certain users require static IP addresses.
- A browser without cookies loses its usability in a high number of web applications. This situation may not be affordable for certain users.

Another straightforward way to conceal the source of a query is using an anonymizing proxy. There are many public proxy servers available on the Internet. When a user wants to submit a query to a WSE, she sends the query to the proxy. Then, the proxy submits the query to the WSE, receives the response and sends it back to the user. The whole process is done anonymously, *i.e.* the true source of the query remains hidden. However, since all the queries are sent through the same proxy, they can be easily linked together by the proxy itself. An adversary with access to the logs of the proxy could identify the true source of the queries.

This problem can be solved by using a group of proxies instead of only one. In this way, Chaum proposed in [17] the use of an anonymity network which consists of several routers that act as anonymizers. The input and the output routers in the network are rotated among them. Therefore, logs from all the routers are necessary in order to link the queries which have been generated by the same user. There are many implementations of this scheme. Probably, the *Tor Project* [103] is the most renowned.

The main drawback of this approach is that user cannot receive a personalized search. Moreover, the process of submitting a query to the WSE and receiving the answer through an anonymous channel is very time-consuming. The authors in [90] used the anonymous network Tor with paths of length two (note that the default length is three) and submitting a query was, on average, 25 times slower than performing a direct search. In addition, anonymous channels are not sufficient to preserve the privacy of users. They only take care of the data transport, hence users should use specific programs to hide identifying information. This information can be obtained from the cookies, the HTTP headers or from active components of the websites.

This situation is usually solved by using the anonymity network in combination with an HTTP filter like Privoxy [84]. This tool attempts to delete all the unnecessary information that users submit to the WSE. As a result of this process, the disclosure of personal information is reduced but it does not improve the response time of a query submission. In addition,

Privoxy is a general-purpose filter and it does not remove active components like JavaScript or ActiveX. FoxTor [88] and TorButon [103] are two FireFox plug-ins that combine a Tor network with a Privoxy filter.

An alternative method for providing privacy is based on obfuscating the profiles by means of noise. Submitting false random queries is an intuitive way of achieving it. Schemes that follow this approach provide non-anonymous privacy in the sense that a certain user can be identified but her interests remain hidden. The authors in [94] state that this approach can be considered as a way to get k-anonymity. The author in [99] explains that a release provides k-anonymity protection if the information for each person contained in the release cannot be distinguished from at least $k-1$ individuals whose information also appears in the release. In a WSE scenario, a protection level comparable with k-anonymity is achieved if a query of a certain user cannot be distinguished from at least $k-1$ queries generated by other users. This means that k different queries have the same probability of being the real one.

The main difficulty in generating false random queries is selecting the query terms properly. Some proposals choose the false query terms outside the interests of the user [36, 35, 32]. These proposals try to confuse the WSE. On the other hand, schemes presented in [58, 48] select the false query terms from the topics which are relevant to the user. This latter approach focuses on the general areas of interest of the user. The specific ones are avoided.

In the literature, there are two main works based on these two approaches:

- TrackMeNot [48] uses the periods when the activity of the users is low to submit random queries to the WSEs. In this way, the system does not affect the normal work of the users. However, sending fake queries increases the network traffic and overloads the WSEs. As a result, this scheme protects the privacy of the users but it also reduces the network and WSEs performance. In addition, this behavior introduces a serious privacy threat: for each user, the WSE is able to divide all her

queries depending on whether or not they have been submitted during working hours (according to the time zone of the user). Probably, all the queries which have been submitted out of the working hours have been sent by TrackMeNot. The period of time between the submission of two different queries can also be used for the same purpose: it can be assumed that when users are working, they do not submit only one query but several in a short period.

- GooPIR [32] submits a bunch of queries that contain fake words together with the authentic term. The WSE cannot know which words are fake and which are not. In this way, the profile of the user is obfuscated and her privacy is protected. This proposal uses a Thesaurus in order to decide which words can be added to each search. Thus, GooPIR can only submit words. Full sentences are not addressed (note that sentences cannot be formed by random words). Moreover, the system assumes that the frequencies of keywords and phrases that can appear in a query are known and available: for maximum privacy, the frequencies of the target and the fake queries should be similar, so that the uncertainty h(k) of the search engine about the real target query is maximum.

3.1.2 Collaborative schemes

Private information retrieval (PIR) protocols were proposed to be used in scenerios where the database server cooperates. A PIR protocol allows a user to retrieve a certain item from a database without the latter learning what item is being acquired. Trivially, PIR can be achieved by sending a copy of the entire database to the user, but this is very inefficient and infeasible in practice.

The first PIR protocol was proposed in 1997 by Chor, Goldreich, Kushilevitz and Sudan [19, 20]. However, it requires the existence of at least two copies of the same database. Besides, those databases cannot communicate between

them. Accordingly, this proposal cannot work in a single server scenario like WSE.

Two years later, Kushileviz and Ostrovsky [60] presented the first single-database PIR scheme. Nevertheless, it requires the cooperation of the database. This fact represents a major drawback which disqualifies it in scenarios where the database is not willing to collaborate. WSEs are an example of this situation: they have no motivation to protect the privacy of users.

For the above reason, it seems that relaxing the PIR assumptions is needed for the sake of practicality. We next review some other methods where users collaborate between them in order to reach the same purpose.

Crowds [86, 87] is a system based on the concept of *users blending into a crowd*. In this scheme, a user tries to hide her actions within the actions of many others. Like Tor, it is based on the mixing system of Chaum [17], but in this case, the users also act as routers. This proposal works as follows: a certain user who wants to submit a query can send it directly to the WSE or she can forward it to one of her *neighbours* in the crowd. A neighbour who receives a query can submit it to the WSE or she can forward it again to one of her own neighbours. A query is forwarded between users until someone submits it to the WSE. There are some shortcomings in this framework proposal:

- Like in the anonymity networks, Crowds only protects the data transport. Users are responsible for hiding their private information.
- Personalization is only possible if the members of the crowd share the same interests.
- The authors argue that Crowds can scale without limits: the load on each user stays approximately constant as the crowd size grows, although this can not be guaranteed. Moreover, The structure of the crowd must be maintained and this task is costly.

- This proposal requires a central node which manages the crowd (users joining/leaving the crowd).
- In order to keep a certain user concealed, her queries have to be uniformly distributed among the rest of the users [99]. This scheme does not address this point.

With the same practical spirit, the system described in [27] cloaks a user in an anonymous peer-to-peer (P2P) user community. A user submits queries on behalf of her anonymous peers and conversely. Pairs of users in the P2P community share symmetric encryption keys which are used to establish confidential channel. In this way, the database still learns which item is being retrieved (which deviates from strict PIR), but it cannot obtain the real query histories of users, which become diffused among the peer users. The resulting relaxation is named user-private information retrieval (UPIR).

The same authors in [28] present an evolution of [27] which uses shared memory sectors to store and read the queries and their answers. In this way, there is no connection between users. The authors argue that a wiki-like collaborative environment can be used to implement a shared memory sector. This scheme has the following drawbacks: (i) there is no study about the memory-space requirements of this proposal; (ii) users must scan their shared memory sectors at regular intervals, and this introduces a significant overhead to the network; and (iii) this scheme achieves a response time of 5.84 seconds without considering the network time. Thus, the final response time is expected to be clearly above 5.84 seconds but the exact value is not specified by the authors.

A method that only exposes a part of the user profile is presented in [110]. This scheme extracts the interests of the user from her browsing history and emails. All the collected information is organized following a hierarchical tree where the leaves are the specific interests. Only the general interests are shown to the WSE. The authors argue that this behavior provides a fair search quality and a certain level of privacy to the user. Nevertheless, the

WSE can still create a user profile with her general interests. In addition, this proposal requires certain modifications at server side, hence it is not suitable for all the WSEs.

In [14], the Useless User Profile (UUP) protocol is proposed. In this scheme, every user who wants to submit a query will not send her own query but a query of another one instead. Users do not know which query belongs to each user, hence the privacy of users is preserved. Confidentiality is achieved by means of certain cryptographic tools (a threshold cryptosystem and a ciphertext re-masking operation). This scheme has been tested in a real environment and it provides an overhead of 5.2 seconds with a group of three users and a key length of 1024 bits. The shortcomings of this proposal are the following:

- This scheme requires a central node which creates the groups of users. The process of creating groups introduces a significant delay as it requires a large number of users in order to provide an acceptable response time.
- It does not consider the trade-off between the privacy level achieved and the quality of the service.

The scheme proposed in [108] uses the same principle as Crowds [86]: users submitting queries on behalf of other users. An attractive feature of this system is that an existing social network (*e.g* Facebook) can be used as a peer community. This fact makes its deployment quite straightforward. Another benefit from the use of social networks is that users who share the same group are intended to be friends in real life. This implies that they are likely to share similar interests, hence, the distorted profiles still allow users to get a proper service from the WSE [72]. Besides, this contribution improves former proposals in terms of query delay. Nevertheless, this scheme presents some drawbacks that are summarized below.

- This scheme relies on two functions in order to work properly. The first function estimates the profile exposure level and selects who is the user that must send a certain query in order to preserve the privacy of users. The second function evaluates the selfishness of the users and punishes the users who do not collaborate with the group. The most effective these functions are, the better the system behaves. However, the authors do not provide a deep study of these functions and they leave its design as future work. Therefore, achieving better implementations of both functions is of paramount importance.

- The authors have simulated their scheme and they have demonstrated its functionality in the environment proposed by them. Nevertheless, this scheme should be analyzed when dealing with real data in order to gauge the real privacy level achieved by this solution. In addition, some measurement functions are needed to evaluate the levels achieved by the proposed protocol in terms of: privacy, protection against selfish users, usability and quality. Without a standard measurement function, it is not possible to argue whether a certain scheme works properly or not.

3.2 Server-side disclosure control techniques

Once WSEs have gathered the users' queries, they assume the corporate social responsibility of preserving the privacy of their users. Accordingly, in order to make the most of the collected data, WSEs should anonymize the data taking care to minimize the information loss (IL).

In the literature, we can find several approaches dealing with the query log anonymization problem based on query removal or hashing. In [21] a survey of these naive methods is given. First, some systems simply remove old query sets assuming that user logs will not be large enough to enable identity disclosure [15]. This simple criterion hardly preserves privacy in front of highly identifying queries. A more appropriate approach suggests deleting

only infrequent queries [1], assuming that those are more likely to refer to identifying or quasi-identifying information. This is, however, challenging due to the difficulty of setting deletion thresholds and due to the fact that the vast majority of queries occur a small number of times [11], which may result in eliminating a substantial amount of non-identifying queries.

Other removal-based methods focus on removing identifying data (e.g. IP addresses) associated to queries and/or identifying/private information found within queries (e.g. SS numbers, credit cards, addresses, etc.) [16]. However, as discussed above, the combination of the remaining no-identifying data (i.e. quasi-identifiers) may end with disclosure identity, like in the case of the infamous AOL incident. In a different approach inspired in secret sharing methods, Adar [1] proposes to split user queries, assigning them to fictional user ids. This, however, limits the usability of results, thus invalidating the conclusions extracted by a horizontal analysis of queries (e.g. user profiling methods) [46]. Finally, other systems rely on the application of hashing functions to identifying information (e.g. IP address) and/or to queries themselves [59]. Even though this makes the identity disclosure difficult, it also requires from revealing hashing functions to parties willing to perform query analysis. Moreover, some works have also shown the ineffectiveness of hash-based schemes, such as token-based hashing [59].

More elaborated approaches [57, 82] remove only those queries that result in clicking common URLs, assuming that those may be dependent (i.e. quasi-identifiers). Poblete et al. [82] represent query logs as a graph in which nodes are queries and two nodes are connected by one edge if the intersection of their clicked URLs sets is not empty. In particular, they propose a graph disconnection heuristic focused on the elimination of queries (and the corresponding edges). The complete anonymization process is iterative and encompasses the removal of (i) all vulnerable queries, i.e. all queries that return less than k documents (over-restrictive queries) and queries that contain the target URL or at least the site of the URL as a keyword (well-target queries), (ii) those returning documents from less than K sites and (iii) all queries that contribute to a nonzero density of the query graph. The density

of the graph is the likelihood of finding an edge among any two nodes. The method stops when the value of density is zero. Korolova et al. [57] propose to generate a private query click graph. In a nutshell, for every user of the query log, it keeps only the first d queries posed by the user. This step limits users' activity. Then, for each remaining query q_i which appears n_i times, the method outputs the query q_i and n'_i, where n'_i is n_i plus a random variable drawn independently from the Laplace distribution with mean zero. The remaining queries are considered safe to publish. Although the method has several parameters, the authors provide some indications and lemmas to compute them according to the parameter d. Even though both approaches focus on the removal of the most informative queries, the obvious drawback is that their utility is completely lost [21].

More recent approaches rely on SDC methods to anonymize query logs while minimizing the amount of query removal [73, 47]. To do so, they group similar users together (according to the similarity of their query logs), and then their queries are replaced by a prototype query log, thus becoming indistinguishable. In this manner, users and queries are preserved, even though the latter are transformed to minimize the disclosure risk.

Semantics have been scarcely considered in related works. The pioneer work by Terrovitis et al. [102] deals with textual set-valued data proposing generalizations of input values according to ad hoc constructed Value Generalisation Hierarchies (VGH), which iteratively generalize input values up to a common node. Authors generalize groups of input queries to a common conceptual abstraction (e.g. *sailing* and *swimming* $\rightarrow$ *water sports*), until users who performed those queries become indistinguishable (i.e. k-anonymous).

Due to the dimensionality and unbounded nature of query logs, which makes the construction of ad hoc VGH unfeasible, authors in [46] propose to anonymize the set of queries made by a user by generalizing the queries using WordNet [70]. WordNet is a generic lexical database of the English language, where concepts are interlinked by means of conceptual-semantic and lexical relations. The problem of relying in WordNet when facing the

anonymization of query logs is that the query introduced by the user can be meaningless in a generic dictionary, despite the fact that they might not be in English. We think that better results can be obtained for query logs by gathering semantic information from the Open Directory Project (ODP), whose its main purpose is precisely to serve as a catalog of the Web by providing a content-based categorization or classification of Web pages. Nevertheless, we need to introduce novel approaches to make the information obtained from ODP useful. Unlike WordNet, which already has lots of published and tested distance functions, or aggregation operations, ODP lacks this extensive previous work.

Chapter 4

Client-side anonymization

4.1 Introduction

As we stated in Chapter 1, once personal data are gathered, users can do nothing to prevent the WSEs from using them for commercial purposes. Therefore, users should employ privacy-preserving mechanisms which prevent WSEs from profiling them in a detailed way. In addition, if users anonymize their own profiles in a proper way, their privacy will be properly preserved even if the data are disseminated.

However, remember that WSEs collect and store information about their users in order to tailor their services to their users' needs. Thus, there is a trade-off between the privacy level achieved and the quality of the service. If the user desires a high degree of privacy, she will probably receive a deficient service. If the user desires an accurate service, her privacy will probably be jeopardized. Our objective is to provide a mechanism which obfuscates users' profiles in a way that they remain useful to provide an accurate service while minimizing the disclosure risks.

The review of the current proposals in the literature shows that the work presented in [108] offers unique and very interesting features:

- It uses existing social networks in order to provide already-generated groups of users.
- These fixed groups are made of friends in real life. Therefore, the users of a certain group are very likely to share similar interests. As a result, [108] generates a distorted profile which is a trade-off between the privacy level achieved and the quality of the service.
- It outperforms the rest of proposals in the literature in terms of query delay. A system that provides a small query delay is more likely to be used by the users.

Nevertheless, the following research questions appear when considering this scheme:

- The privacy level achieved by the users of this proposal depends on the function that calculates the probability of submitting a query. Can this function be re-designed to improve the current results?
- Mechanisms to measure the privacy level achieved by the users are needed in order to compare different proposals. Is there a standard measure that can be used for this purpose?
- The simulations which are shown in [108] have been performed using synthetic queries (queries which are generated at random by a computer) and each user is always submitting the same one towards the WSE. Will the use of real queries (queries which are generated by humans) influence the behavior of this scheme in terms of privacy protection?

4.2 Contribution

A collaborative system to preserve the privacy of WSEs' users is proposed. Specifically, we improve the scheme presented in [108]. The main contribu-

tions are next summarized:

- The function used to decide which user must submit a certain query to the WSE has been studied and re-designed. As a result, the privacy level achieved by the users has improved.
- A new measure to estimate the privacy achieved by the users, the *Profile Exposure Level (PEL)*, is proposed.
- For the first time - to the best of our knowledge - the tests have been performed using real data extracted from the AOL file [4]. In this way, the correct behavior of the proposed system has been tested with queries which have been generated by real users.

These changes improve the privacy achieved by the users in the previous version while keeping its usability. The protocol submits standard queries to the WSE, so it requires neither changes at the server side nor the collaboration of the server with the users. The system is based on the two following assumptions: (i) due to the flat-rate broadband connection proliferation, users (their computers) are most of the day connected to the Internet; and (ii) users are organized in social networks.

The following sections are organized as follows: Section 4.3 presents our proposal in detail. The measurement methods which have been used are explained in Sections 4.4, 4.5 and 4.6. Section 4.7 presents the simulation results and they are compared with the results achieved by [108]. Finally, some conclusions are given in section 4.8.

4.3 Protocol for protecting the privacy of the users

The proliferation of flat-rate broadband connections enables users to be connected to the Internet most of the day. More specifically, the use of mobile devices with communication capabilities (*e.g.* iphone, htc, blackberry...)

allows users to be online most of the time, receiving the latest tweets or facebook messages among others. This fact paves the way to new applications like Peer-to-Peer (P2P) [3]. In a P2P environment, users collaborate between them to perform a particular service. Crowds [86] and P2P-PIR [27] are two examples where the user profile is obfuscated within a group profile.

These schemes do not overload the network but require the maintenance of the network structure. In order to solve this threat, existing structures are used: *Social Networks*. More specifically, the social network concept, which is explained in [24], is followed. This approach does not require the existence of a central node. Therefore, users are connected directly between them or by means of other users who act as intermediaries. Besides, these social networks are not open to examination (a user only knows her direct connections in the network).

The users of a social network are connected to similar users with common interests [72]. Accordingly, a group of users who are maintained by a social network can be used to create a usable user profile. Note that WSEs need usable profiles in order to provide a proper service. By using the proposed protocol, the WSE obtains a profile of each user but this profile is not detailed. In addition, our work is based on social networks that allow users to know the number of neighbours that a certain neighbour has (only the number of connections, not the identities behind them). This helps to equitably distribute all the queries across the network.

Next, the performance of the protocol is briefly described: a user U generates queries that she can either directly submit to the WSE or she can send to a neighbour (a neighbour is a direct relationship in the social network). A neighbour that receives a query can submit it directly to the WSE or she can forward it again to one of her own neighbours. This process is repeated until someone submits the query to the WSE. The profile of U is distorted by the queries that she submits to the WSE, but which are generated by other users of the social network.

Ideally, all users should behave properly. Nevertheless, this cannot be guar-

anteed in a real environment. Therefore, two types of users can be defined:

- *Selfish user.* A selfish user is a user who does not follow the proposed protocol. When a selfish user wants to submit a query to the WSE, she sends the query to a neighbour who submits it on her behalf. However, when a selfish user receives a query from another user, she systematically discards it and she does not answer.
- *Honest user.* An honest user is a user who follows the proposed protocol. When an honest user wants to submit a query to the WSE, she decides if she submits it directly or if she forwards it to a neighbour who submits it on her behalf. When the honest user receives a query from another user, she decides if she submits the query to the WSE or if she forwards the query to another user.

A selfish behavior prevents the queries from being distributed equitably among the group. This situation can jeopardize the privacy of the honest users. Thus, we propose a mechanism to prevent users from behaving in that way.

4.3.1 The protocol in detail

Supposing that a certain user U_i is a member of a social network and that she has k neighbours (direct connections) $\{N_1, \ldots, N_k\}$, U_i knows the number of connections of each of her neighbours. With all this information, U_i can calculate the *sending probability* P_s for each neighbour (see Section 4.3.2 for more details about this probability). Besides, U_i keeps a measure about the selfishness level of each neighbour (see Section 4.3.5 for more details about the function that evaluates the selfishness and its parameters).

When U_i wants to submit a query q to the WSE, she runs the following protocol:

1. U_i executes the *user selection function* $\Psi(U_i,N_1,\ldots,N_k)$, which returns a sorted vector ν of users belonging to the group $\{U_i,N_1,\ldots,N_k\}$. U_i will use ν to decide whether she submits q to the WSE or she sends q to a neighbour N_j (see Section 4.3.4 for details about the ordering of ν and the operations performed by Ψ).

   Let U_j be the j-th user belonging to vector ν. U_i can carry out two actions according to the value of U_j:

   (a) If $U_j = U_i$, then U_i submits q to the WSE.

   (b) If $U_j \in \{N_1,\ldots,N_k\}$, then U_i sends q to U_j. U_j can accept or reject q, hence there are two possible behaviors:

   - If U_j rejects q then U_i updates negatively the parameters that measure the selfishness of U_j (see Section 4.3.5).
   - If U_j accepts q then U_i initializes a timer. If the response from U_j arrives before the end of this timer, U_i updates positively the parameters that measure the selfishness of U_j. Otherwise, U_i updates negatively the parameters that measure the selfishness of U_j (see Section 4.3.5).

   This process is repeated until someone accepts q. In case that all the neighbours reject q, U_i submits her query by herself.

2. U_j executes the *selfishness function* $\Upsilon(U_i)$ in order to either accept q or not. Therefore, there are two possible situations:

   - If U_j accepts q, then U_j is the new responsible for submitting q. Thus, U_j repeats the first step of the protocol by replacing the function Ψ with Ψ_f (i.e. U_i executes Ψ_f). Let Ψ_f be the function that decides whether U_j has to either submit q to the WSE or not. If Ψ_f answers negatively to that question, U_j attempts to forward

q to a random neighbour. The aim of Ψ_f is to distribute the queries among U_j and her neighbours equitably (see Section 4.3.3 for more details about it). When U_j receives the answer to q (*i.e.* from the WSE or from a neighbour), she returns it to U_i.

- If U_j rejects q, then U_j ends her participation and U_i updates negatively the parameters that measure the selfishness of U_j.

4.3.2 Sending probability P_s

Each user assigns to each one of her neighbours a sending probability P_s in order to equitably distribute her queries throughout the network.

Let U_i be a user, $\{N_1,\ldots,N_k\}$ be her group of neighbours and $\{H_1,\ldots,H_k\}$ be the number of neighbours that each neighbour of U_i has (i.e. H_j is the number of neighbours of N_j). Note that k is the number of neighbours of U_i. U_i assigns a certain P_s to each neighbour, so that the probability that U_i sends a query q to her neighbour N_j is proportional to H_j:

$$P_s(N_j) = \frac{H_j}{\sum_{f=1}^{k} H_f + 1}$$

4.3.3 Query forward function Ψ_f

Let U_i be a user and $\{N_1,\ldots,N_k\}$ be her group of neighbours. Function Ψ_f determines whether U_i should submit the query q to the WSE or she should forward it to one of her neighbours. The selection of a certain user within $\{U_i,N_1,\ldots,N_k\}$ is equiprobable:

$$\Psi_f = \frac{1}{k+1}$$

4.3.4 User selection function Ψ

Considering a certain user U_i who wants to submit a query q. U_i has the following list of neighbours: $\{N_1,\ldots,N_k\}$. U_i executes Ψ in order to decide which user within $\{U_i,N_1,\ldots,N_k\}$ should submit q to the WSE.

U_i is perfectly concealed when all the members of $\{U_i,N_1,\ldots,N_k\}$ have submitted the same number of queries generated by U_i [108]. If U_i achieves this requirement, she will be hidden among the group $\{U_i,N_1,\ldots,N_k\}$. Thus, she will achieve a privacy level comparable to k-anonymity [99].

The system uses probability P_s to equitably distribute the queries in a path of length two, *i.e.* the source of the queries submits to the WSE the same number of queries as her neighbours and as the neighbours of her neighbours. This is possible because the number of neighbours and the number of neighbours of these neighbours are known. Since the complete topology of the social network is unknown, it is not possible to control the distribution of queries in paths longer than two. Therefore, more distant neighbours will submit a small quantity of queries to the WSE. As a result, the true source of the queries is hidden among a group with a path length of two.

Nevertheless, if the group is known, then, the WSE can obtain certain information. For example, consider a group of users $\{U_i,N_1,\ldots,N_k\}$, where each one has always submitted the same query. In this extreme situation, the WSE can know the structure of the group and it can be also capable of determining the direct and indirect (path of length two) connections that a certain user holds with the other members of the group. If the WSE gathers all this information, it straightforward finds the centroid of the users who have sent the same number of queries and can hence identify the source of the queries.

This weakness is solved by modifying P_s in order to obfuscate the users within $\{U_i,N_1,\ldots,N_k\}$ adding more distant neighbours. In this way, the system creates a vector which contains the users $\{U_i,N_1,\ldots,N_k\}$. Each user in this vector appears repeated as many times as the number of neighbours

she has. U_i appears only once. This vector is duplicated a random number of times. Finally, an interval of vector elements are deleted at random. The final vector which is obtained through this process is the new P_s.

Let θ be the interval of users which are pruned in order to create variance in the system. The variance prevents the WSE from calculating a group centroid. Let U_i be a certain user and $\{U_i,N_1,\ldots,N_k\}$ be the group of users which have submitted queries from U_i to the WSE. The group $\{U_i,N_1,\ldots,N_k\}$ is sorted in ascending order by the number of queries from U_i which they have submitted. If during several executions of the protocol, the position of U_i in this group remains the same, or very similar, the WSE could be able to identify the source of the query. Therefore, θ is randomly chosen between two percentages (these values have been calculated empirically and are justified in Section 4.7) and it can be recalculated as often as desired.

4.3.5 Selfishness function $\Upsilon(N)$

Let U be a user that receives a query q from her neighbour N. U executes Υ in order to decide whether to accept q or not. The aim of Υ is to punish the users who behave in a selfish manner.

Initially, U assigns to each neighbour a probability p of accepting queries from that source. Value p is set to 1.0 (100% of probability) for each neighbour. For notation purposes, $p_{U,N}$ represents the probability of U to accept a query from N.

Assuming that U receives a query q from user N, U accepts q with probability $p_{U,N}$:

- If U accepts q, the following actions are performed:
  - N increments her own probability of accepting queries from U:

$$p_{N,U} = p_{N,U} + (2 \cdot \zeta)$$

Where ζ is a constant defined in the system. Value ζ is calculated empirically and justified in Section 4.7.

- U decrements her probability of accepting queries from N:

$$p_{U,N} = p_{U,N} - \zeta$$

Both operations are done to incentivize users to accept queries from their neighbours. A certain user U who forwards several queries to the same neighbour N without accepting queries in exchange (selfish behavior) will finally get a $p_{N,U} = 0$. If the same situation happens with all her neighbours, U will be isolated and forced to submit all her queries to the WSE by her own. An isolated user is able to improve her situation by accepting queries from her neighbours. By decreasing ζ for a reject and increasing $2 \cdot \zeta$ for an accept, the protocol punishes the users that systematically reject all queries instead of the users that accept and reject queries.

- If U rejects q, the following happens:
  - N decrements her own probability of accepting queries from U:

$$p_{N,U} = p_{N,U} - \zeta$$

We have simulated our scheme for several values of ζ to check out which one better isolates the selfish users of the system. The best results were achieved with $\zeta = 0,03$ (see Section 4.7 for details about the choice of this value). Therefore, this is the value used hereinafter.

4.3.5.1 Coprivacy

The proposed scheme protects the privacy of the users in communities where most users are honest. If there is a large quantity of selfish users, the privacy of honest users would be jeopardized (see Section 4.7 for more details about

it). On the other hand, selfish users lose their privacy in both scenarios. Besides, an honest user alone cannot protect her own privacy. Therefore, honest users do not have any motivation to leave the network, as they need each other. Regarding the selfish users, if they leave the social network, the system will work better. If they do not leave the network, they will be isolated by honest users by using the selfishness function Υ.

The situation where users collaborate in a system to preserve their privacy is named *coprivacy* and it was defined in [26]. A protocol is *coprivate* if the best option for a player to preserve her privacy is to help another player in preserving her own privacy.

4.4 Evaluation

The review of the state of the art exposes the features that should be fulfilled by any system which provides privacy to WSEs' users. These features are: privacy, protection against selfish users, usability and quality. Evaluating a system is based on assessing these four characteristics. In our case, the optimal values for ζ and θ are the ones that offer better results for these features.

To the best of our knowledge, there is no standard method to evaluate these four attributes. Accordingly, we propose the following ones which are later used to analyze our system:

- *Privacy.* The user profile must be kept hidden from the WSE. We propose the Profile Exposure Level (PEL) which uses mutual information in order to measure the level of exposure of the user profile (see Section 4.5).
- *Protection against selfish users.* If a user behaves in a selfish way she must be penalized. In our simulations, we use different values of ζ in order to find out which one performs better against selfish users.

- *Usability.* This feature is crucial for the system. A completely secure scheme that suffers from high query delay will not be used by most users (see Section 4.6). In our work, the usability is evaluated by using the average response time of a query.
- *Quality.* The quality refers to the similarities between the responses which are gathered using the proposed system and the responses which are received submitting the queries directly to WSE. Our protocol submits original queries to the WSE (queries are not modified in any way). Thus, this feature is not considered in our evaluation.

4.5 Profile Exposure Level

Let Ω_X be the real queries of the user and Ω_Y be the queries sent to the WSE. Note that Ω_X and Ω_Y can be seen as two random variables X and Y respectively, which can take so many values as different queries they have and with probability proportional to the number of repetitions. Accordingly, we define the Profile Exposure Level (PEL) as follows:

$$PEL = \frac{I(X,Y)}{H(X)} \cdot 100$$

where H(X) is the entropy of the original set of queries and I(X,Y) is the mutual information between X and Y. PEL measures the percentage of the user information that is exposed when Y is disclosed. Thus, the user information is calculated as the entropy of X, and the mutual information gives a measure of the information that Y provides about X (*i.e.* if Y is known, how much does this reduce the uncertainty about X?).

4.5.1 Mutual Information

Given two random discrete variables X and Y, that have sample spaces Ω_X and Ω_Y respectively, it can be considered that:

1. The probability function of the variable X is defined by $p(x)$ when, for all $x \in \Omega_X$, $p(x) = P(X = x)$.
2. The probability function of the variable Y is defined by $p(y)$ when, for all $y \in \Omega_Y$, $p(y) = P(Y = y)$.
3. The joint probability function of the variables X and Y are defined by $p(x, y)$ when, for all $x \in \Omega_X$ and $y \in \Omega_Y$, $p(x, y) = P(X = x, Y = y)$.
4. The probability function of the variable X conditioned on the variable Y is defined by $p(x/y)$ when, for all $x \in \Omega_X$ and $y \in \Omega_Y$, $p(x/y) = P(X = x/Y = y)$.

The mutual information, $I(X, Y)$, of two random variables X and Y is a measure that allows us to evaluate the information that each variable provides about the other. $I(X, Y)$ shows the amount of uncertainty in X which is removed by knowing Y. Mathematically, this is expressed as:

$$I(X, Y) = H(X) - H(X/Y)$$

where $H(X)$ is the entropy of the variable X and $H(X/Y)$ is the conditional entropy of the variable X, given the variable Y. $H(X/Y)$ is defined as the uncertainty about X which still remains after Y is known. These entropies are expressed as:

$$H(X) = -\sum_x p(x) \cdot \log_2 p(x)$$

$$H(X/Y) = -\sum_{x,y} p(x,y) \cdot \log_2 p(x/y)$$

From the previous expressions we can obtain the following one:

$$I(X,Y) = -\sum_{x} p(x) \cdot \log_2 p(x) + \sum_{x,y} p(x,y) \cdot \log_2 p(x/y) = \sum_{x,y} p(x,y) \cdot \log_2 \frac{p(x/y)}{p(x)}$$

Being *p(x,y) = p(x/y)· p(y)*, the former expression develops to:

$$I(X,Y) = \sum_{x,y} p(x/y) \cdot p(y) \cdot \log_2 \frac{p(x/y)}{p(x)}$$

In our scheme, X represents the queries which are originally generated by the user, *i.e.* the queries she wants to submit. The variable Y represents the real queries that the user finally submits to the WSE.

Notation

$\Omega_X = \{x_i\}_{i=1}^{m}$, set with the elements of X (without repetitions).

$\Omega_Y = \{y_i\}_{i=1}^{r}$, set with the elements of Y (without repetitions) .

$C_X = \{c_{x_i}\}_{i=1}^{m}$, set with the cardinal of each element of X.

$C_Y = \{c_{y_i}\}_{i=1}^{r}$, set with the cardinal of each element of Y.

$M = \sum_{i=1}^{m} c_{x_i}$, total number of elements of the set X, counting repetitions.

$R = \sum_{i=1}^{r} c_{y_i}$, total number of elements of the set Y, counting repetitions.

Calculation of *p(x)* and *p(y)*

Let us suppose that the probability of each element of X and Y is proportional to its cardinal. Then, *p(x)* and *p(y)* are computed as follows:

$$P(X = x_i) = \frac{c_{x_i}}{M},\ 1 \leq i \leq m.$$

$$P(Y = y_i) = \frac{c_{y_i}}{N},\ 1 \leq i \leq r.$$

Calculation of *p(x/y)*

$P(X = x_i/Y = y_j)$ is calculated for each pair x_i, y_j where $1 \leq i \leq m$ and $1 \leq j \leq r$.

Fixing $y_j \in Y$, the possible situations and their calculations are:

1. $y_j \notin X$. There is no information, *i.e.* the probability is randomly assigned among the elements of X proportionally to the cardinal.

$$P(X = x_i/Y = y_j) = \frac{c_{x_i}}{M},\ 1 \leq and \leq m.$$

2. $y_j \in X$. Then there exists a $x_k \in X$ so that $x_k = y_j$.

   (a) If $c_{y_j} \leq c_{x_k}$, it is assumed that y_j *comes* from x_k and not from any other $x \in \Omega_x$.

$$P(X = x_k/Y = y_j) = 1$$

$$P(X = x_{k'}/Y = y_j) = 0,\ 1 \leq k' \leq m, k \neq k'$$

(b) If $c_{y_j} > c_{x_k}$, one of the following statements is assumed: (i) y_j *comes* from x_k and not from any other $x \in \Omega_x$, with the probability proportional to the cardinal of x_k; (ii) there is no information with probability proportional to the difference of the cardinals.

$$P(X = x_k / Y = y_j) = \frac{c_{x_k}}{c_{y_j}} + \frac{c_{y_j} - c_{x_k}}{c_{y_j}} \cdot \frac{c_{x_k}}{M}$$

$$P(X = x_{k'} / Y = y_j) = \frac{c_{y_j} - c_{x_k}}{c_{y_j}} \cdot \frac{c_{x_{k'}}}{M},\ 1 \leq k' \leq m, k \neq k'$$

4.6 Usability measure

The international standard ISO/IEC 9126 (Software engineering - Product quality) [50], defines usability as the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use. More specifically, we focus on whether users can easily accomplish intended tasks at their desired speed or not.

In the proposed scheme, a task is considered to be the process of submitting a query and receiving the answer. The period of time needed to execute this process is named *response time*. Systems based on Tor networks obtain large response times. For instance, the authors in [90] got an average response time of 10 seconds. We argue that this time is too long for a web search system to be usable. Therefore, the objective is to reduce it as much as possible.

The period of time between the submission of a query to the WSE and the reception of the answer can be decomposed as follows:

1. The query goes from the user to the WSE.

2. The WSE processes the query and generates an answer.
3. The answer goes from the WSE to the user.

In our scheme, there is an extra step in which the message is forwarded between a certain number of users before reaching the WSE. The answer is returned through the reverse path. Therefore, the average response time of a query is:

$$\text{Response time} = (2 \cdot \#hops \cdot latency) + time_{WSE}$$

Where:

- $time_{WSE}$ is the time needed by the WSE to answer the query. In average, this time is 400 ms [108].
- *latency* is the round-trip time (RTT) between two peers. The study presented in [18] determines that the average latency between two random users in a worldwide P2P network is 530 ms.
- *hops* is the average number of hops that a query performs before reaching the WSE. This value has been obtained from the simulations.

4.7 Simulations

The evaluation of the proposed system has been done by simulating various social networks between 1.000 and 10.000.000 users. In these networks, each user is directly connected with a number of users between 1 and 10 following a power-law distribution [62].

4.7.1 Tests

The evaluation process includes two different types of tests:

- The first type checks the equitable distribution of messages around the network. Each user generates a unique query and sends it many times. This is the worst possible case because all the queries which are submitted by the same user are equal and different from the queries sent by other users. As a result, these queries can be easily linked together. Nevertheless, if the system works correctly, the user who has generated all these queries remains hidden among the set of users who have submitted them. This kind of tests use synthetic queries (queries generated at random by a computer). Besides, the results of these tests provide the optimal values for ζ (see Section 4.3.5).
- The second type of tests use real queries (queries generated by humans) in order to evaluate the privacy level achieved by the users. These queries were extracted from the AOL file [4]. This file shows which queries were submitted by each AOL user (note that the real identity of each AOL user is not disclosed, only her queries). In this way, in our tests, a certain simulated user gets the personality of a certain AOL user. Therefore, the simulated user only sends the queries which were generated by her assigned AOL user. Note that each user submits a different number of queries. Depending on this number, their privacy might vary.

4.7.2 Privacy

Before evaluating the privacy offered by our proposal, we need to determine the interval θ of users that are pruned in order to introduce variance to the system. We run 1.000 tests using social networks of 1.000 users. Table 4.1 shows the average position, variance and deviation of the users with 10 neighbours for different intervals of θ. The highest variance is obtained by

θ interval	Average position	Variance	Deviation
0% - 0%	5,81	8,34	2,89
80% - 40%	4,36	11,54	3,39
60% - 40%	4,46	11,37	3,37
80% - 20%	4,51	10,71	3,27
80% - 60%	4,51	12,5	3,53
40% - 20%	4,49	10,13	3,18
30% - 10%	4,42	10,12	3,18
30% - 20%	4,46	10,21	3,19
20% - 10%	4,3	9,67	3,11

TABLE 4.1: Average position, variance and deviation obtained with different θ intervals.

the 80%-60% interval. This is the interval which has been used in the rest of the simulations.

Figure 4.1 shows the average position and the deviation of the users according to the number of neighbours that they have. For a good comparison, we have included the ideal average position. It can be observed that the average position does not differ much while the position of the users within the group obtains a large deviation.

4.7.2.1 Simulation results from scenarios without selfish users

The optimal system behavior is achieved when all users follow the protocol. We performed tests with 1.000 users from the AOL files [4] to determine the level of privacy that is achieved in this situation. The Mutual Information (see Section 4.5.1) returns the entropy reduction in bits. This information is not useful without the initial uncertainty, hence, we use the percentage of information that involves the Mutual Information about the initial entropy as a measure. Hereafter, we consider that a user is exposed when this percentage is less than 40%, *i.e.* above 60% of the initial entropy.

Table 4.2 shows the average number of exposed users according to the num-

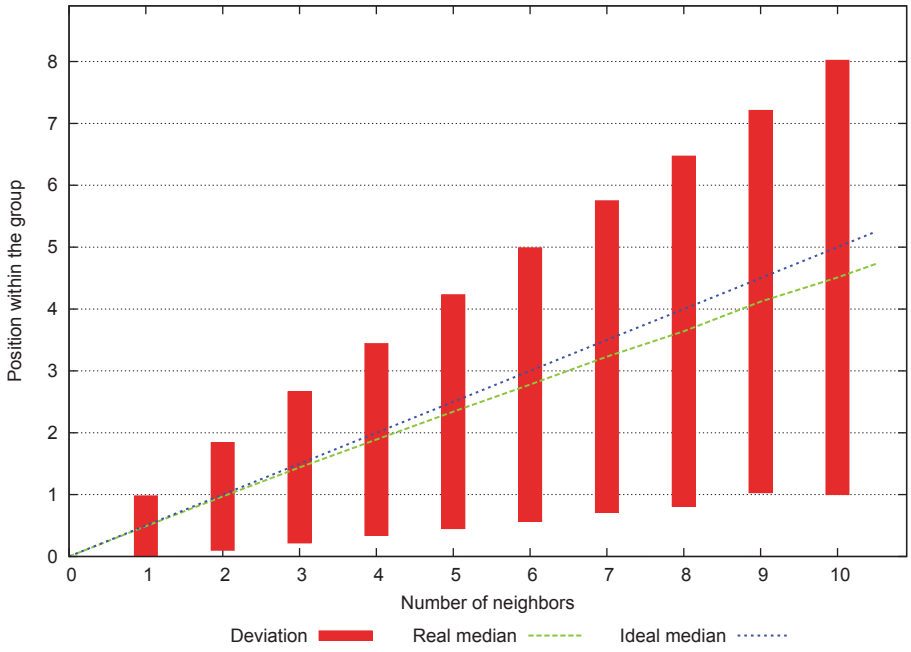


FIGURE 4.1: The ideal and simulated average position and the deviation for each number of neighbours.

ber of neighbours they have. We have completed the table with the number of users who have an uncertainty percentage above 70% and 80%.

Generally, the users with fewer connections are the most exposed ones. However, there are some users with many connections who also expose their profile to the WSE. This case can occur when the number of queries of these users is small or when their neighbours have sent them a small number of queries.

In table 4.3 it can be seen that nearly 90% of the users expose less than 20% of their profile. Moreover, taking into account that a user is exposed when at least 60% of her profile has been revealed, more than 95% of the users preserve their privacy.

# Neighbours	# Exposed 60%	# Exposed 70%	# Exposed 80%
1	15,4	15,1	4,7
2	12,6	12	4,7
3	5,2	4	1,2
4	4,67	4	2
5	1	1	1
6	1,3	0	0
7	1,2	0	0
8	0	0	0
9	0	0	0
10	0	0	0
	41,37	36,1	13,6

TABLE 4.2: Number of exposed users according to various percentages of exposure.

% Similarity	# Users	% Users
10	813,1	81,31
20	82	8,2
30	22,2	2,22
40	20,1	2,01
50	11,3	1,13
60	11	1,1
70	14,7	1,47
80	12,9	1,29
90	10,1	1,01
100	2,6	0,26

TABLE 4.3: Percentage of similarity between the mutual information and the initial entropy for the users.

ζ	**% Exposed honest users**	**% Exposed selfish users**
0,00	2,4	2
0,01	2,3	5
0,02	2,2	93
0,03	2,2	100
0,04	2,4	100
0,06	2,56	100
0,08	2,89	100
0,10	2,89	100

TABLE 4.4: Percentage of exposed users for different ζ values.

4.7.2.2 Simulation results from scenarios with selfish users

Before evaluating the privacy offered by the system in front of selfish users, we have to determine the ζ value which must be applied. The optimal value is important in order to prevent honest users from jeopardizing their privacy. We have run 100 tests over social networks of 1.000 users. 10% of these users were set to behave selfishly. Each user submits her own query 100 times.

Table 4.4 shows the percentage of honest users and selfish users that expose their profile for several ζ values. It can be observed that the number of honest users who are exposed grows when ζ is increased. Thus, the optimal penalty value is $0,03$. This is the value used hereinafter.

In order to evaluate the privacy offered by the system in environments with selfish users, we run 100 test over various social networks of 1.000 users using real data.

Figure 4.2 presents the percentage of exposed users for different percentages of selfish users. Note that, when there are more selfish users, the number of exposed honest users also grows.

With the above tests we have also calculated the number of queries that a selfish user submits to the WSE in a system's execution with real data or

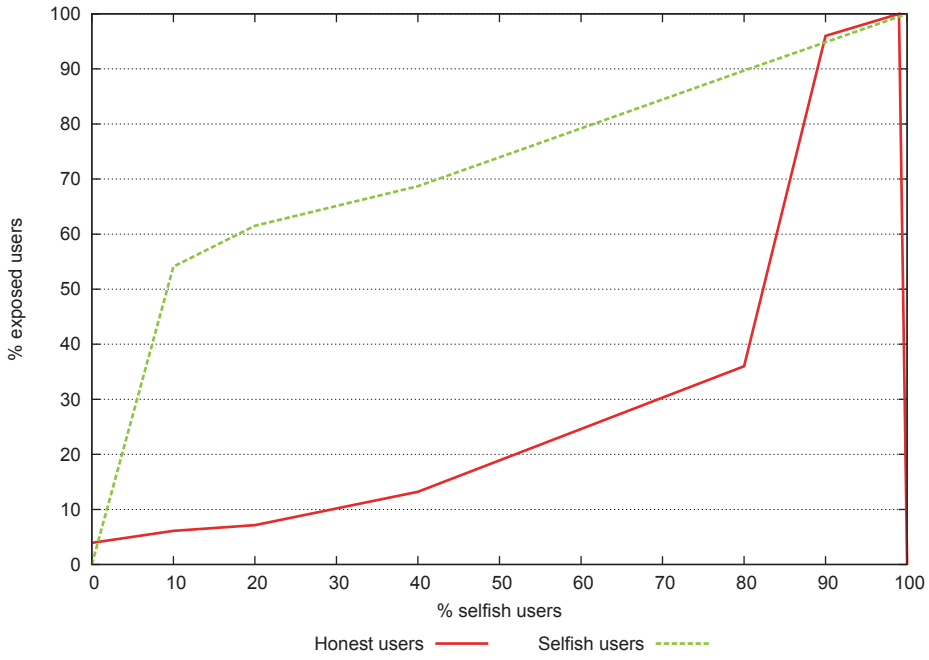


FIGURE 4.2: Percentage of exposed users for different percentages of selfish users.

synthetic data. Figure 4.3 shows the results achieved according the number of neighbours.

4.7.3 Usability

In Section 4.6 we have defined the term *usability*. In this section, our target is to obtain the average number of hops. We have performed several simulations with social networks of 10.000 users. In each simulation, each user generates 1.000 queries. The results show that the average number of hops is $3,59$. Therefore, the average response time of a query is $(2 \cdot 3,59 \cdot 530 \text{ ms}) + 400 \text{ ms} = 4205,4 \text{ ms}$.

This time is slightly worse than the 3914 ms obtained by [108]. However, the privacy level achieved by our scheme is significantly better. The controlled

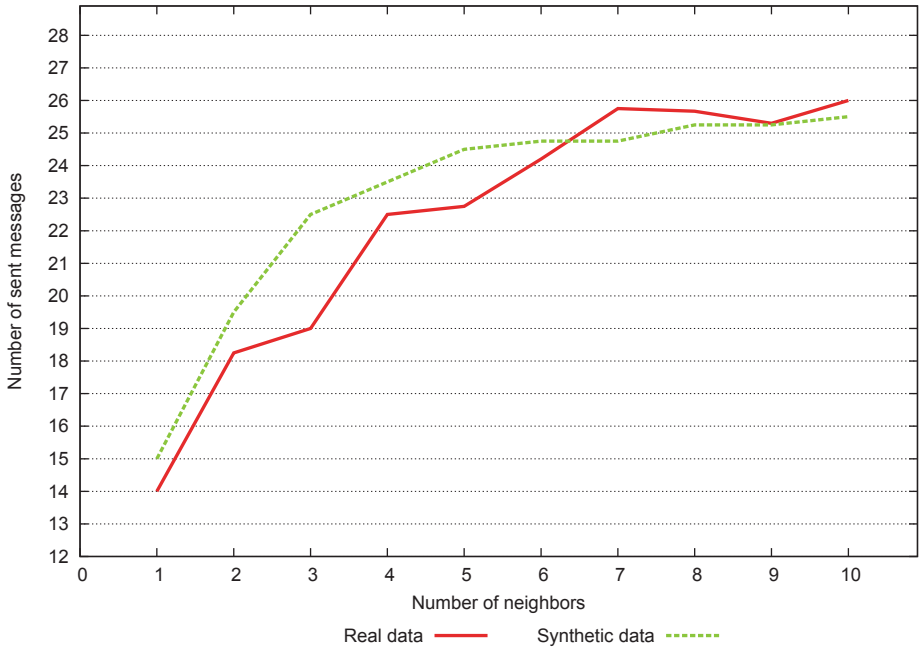


FIGURE 4.3: Average number of queries which are submitted by selfish users.

distribution of queries allows the system to hide the source user among her social network group with a path of length two.

In table 4.5, it can be observed that the new system achieves larger deviations and average positions closer to the ideal than the previous one.

As we have mentioned before, proposals based on Tor have an average response time of 10000 ms with paths of length two. The UUP [14] achieves a response time of 5200 ms. This is a 20 % worse than the time attained by our proposal. In addition, our scheme improves the quality of the search results because it considers the trade-off between the privacy level achieved and the quality of the service.

Regarding the response time of a direct WSE search, note that this search method does not protect the privacy of the users. The privacy-protection

Category	Average position	Previous average position	Deviation	Previous deviation
1 neighbour	0,49	1,01	0,49	0,01
2 neighbours	0,97	1,08	0,87	0,08
3 neighbours	1,44	1,68	1,22	0,72
4 neighbours	1,89	2,11	1,55	0,95
5 neighbours	2,34	3,23	1,89	1,21
6 neighbours	2,78	3,95	2,21	1,55
7 neighbours	3,23	5,15	2,52	2,15
8 neighbours	3,64	6,32	2,83	2,41
9 neighbours	4,12	7,30	3,09	2,40
10 neighbours	4,51	7,00	3,51	2,91

TABLE 4.5: Average position and deviation obtained with our system and its previous version [108].

process represents a cost for the users. They should consider whether their privacy deserves this cost or not.

4.8 Conclusions

We have described a new version of the protocol presented in [108], which preserves the privacy of the users by distorting their profiles. The new version improves the privacy achieved by the users, maintaining its usability.

In [108], the privacy of a certain user U who is generating certain queries is obtained by concealing her into a group Y of k users. This group is formed by the users who have submitted the queries generated by U. The users in Y are ordered according to the number of queries that each one has submitted to the WSE.

The authors of this work state that the WSE can identify U if her position in the ordered group Y is always the same. According to that, U should ideally be situated in the middle of Y, but the deviation of this position

should be high. A high deviation implies that it is difficult for the WSE to ascertain the exact position of U in Y. Thus, the WSE cannot identify U.

Therefore, in order to compare both protocols we have calculated the average position and deviation achieved by our system and we have compared those results with the results attained by the former protocol [108]. Results show that the privacy level achieved by our scheme is significantly better.

In addition to that, as there is a lack of standard measures to evaluate the privacy achieved by the users in this type of systems, we have proposed a new measure: the Profile Exposure Level (PEL). PEL measures the percentage of the personal information that is exposed when a certain user submits her queries to the WSE. This measure can be used by any entity that knows which queries have been generated by the user and which queries have been submitted by the user. Therefore, each user can compute her own PEL because she knows this information.

Also, for the first time - to the best of our knowledge - the tests have been performed using real queries (queries generated by humans) which were extracted from AOL's files. Former proposals in the literature used synthetic queries (queries generated by computers) to test their behavior. From the validity point of view, our approach is more accurate and hence the results of our simulations are trustworthy.

We have simulated the performance of the proposal and the results show that it can be successfully deployed in real environments because: (i) it provides an acceptable query delay; (ii) it offers privacy to users who have a reasonable number of direct connections; and (iii) it works properly in scenarios with users who do not follow the protocol.

Nevertheless, our proposal has only been tested on simulated social networks. Real social networks (*e.g.* Facebook, Windows live messenger...) have not been considered in our evaluation process. Therefore, we cannot assume that the results would be exactly the same in a real network. However, we consider than the behavior should be quite similar since the

simulation process has been realized trying to replicate a real environment.

4.8.1 Publications

- A. Erola, J. Castellà-Roca, A. Viejo and J.M. Mateo-Sanz, *Exploting Social Networks to Provide Privacy in Personalized Web Search*, Journal of Systems and Software, Vol. 84, no. 10, pp. 1734-17445, Oct 2011, ISSN: 0164-1212

Chapter 5

Server-side anonymization

5.1 Search logs

A query or search 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 position in the results ranking of the clicked URL. This format corresponds to logs released by AOL in 2006. Figure 5.1 shows some real logs from AOL data. The information provided in these logs is the same as the logs from AllTheWeb [51], and it closely resembles other released data from Excite [53] or AltaVista [52]. 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 using basic techniques. Thus, the clicked URL is truncated to the domain name before publishing it, as a minor privacy measure. We can also assume that private information from the query terms such as social security numbers

```
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
```

FIGURE 5.1: Example of search query log.

has been removed [109].

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 [7]. Moreover, hashing techniques, applied to the query terms, are vulnerable to frequency analysis [59].

5.2 Search logs release

The release of search logs is related to the microdata release (see Section 2.3) yet quite different. Unlike relational databases, query logs do not constitute well-defined sets of attributes, since several subsets of queries could play the role of quasi-identifiers. Moreover, query logs have variable length and high dimensionality, compared to the relatively few attributes and values in relational records. Finally, query logs are expressed in free text, so they can almost take any possible value of any domain, whereas attributes in relational databases are usually either numerical or categorical.

This unbounded nature of queries make it difficult to detect the potentially identifying information. Several authors [1, 55, 82] have discussed this issue. Hence, although query logs can be submitted to an anonymization

process prior to their publication, there is no absolute guarantee of privacy protection.

We propose the application of microaggregation to anonymize search logs. This approach ensures a high degree of privacy, providing k-anonymity at user level, and preserves some of the data usefulness. To optimize the microaggregation, partition and aggregation operators that minimize the information loss must be defined.

In the next three sections we propose three different microaggregation methods specifically desinged to minimize the information loss of released search logs. For each method, evaluation, results and conclusions are presented.

5.3 Microaggregation of search logs

In this section 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). If there are k indistinguishable users, it is not feasible to re-identify users as in the case of the released logs from AOL [7] for some given k.

Each user of a search log can be represented by a simple ordered tree, grouping all its queries. Figure 5.2 shows the representation of the users from the query logs of Table 5.1. All requests in the query log belonging to the same user are treated as a single record in the protection process.

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

TABLE 5.1: Example of queries.

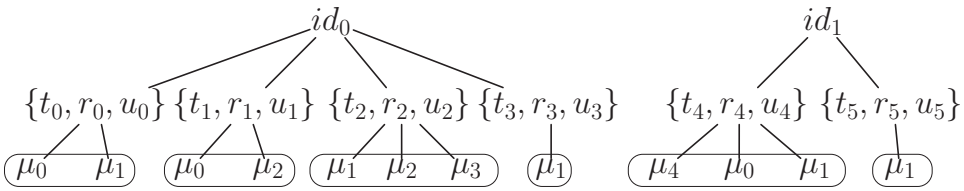


FIGURE 5.2: Example of user query trees.

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.

The rest of the section is organized as follows. In Section 5.3.1 we present the microaggregation of the query logs. Section 5.3.2 provides the evaluation of our proposal both in terms of privacy and usability, and finally, Section 5.4.5 discusses the key findings.

5.3.1 Microaggregation of query logs

In order to reduce excessive information loss resulting from query removal,our proposal will be based on data microaggregation (see Section 2.4). To maximize the utility of anonymized data, similar records should be clustered together, so that the information loss resulting from the replacement by their centroid can be minimized. However, remember that finding the optimal data partition is NP-hard [78] in general. Hence, we propose to use the approximantion algorithm MDAV(see Section 2.5).

The application of MDAV to query logs of the form shown in Figure 5.2 is not straighforward, since 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.

5.3.1.1 Distance and aggregation of query logs

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 $\varphi^i_j = \{t^i_j, r^i_j, u^i_j, \phi^i_j\}$, 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 $|\phi^i_j|$ 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 $\overline{|\phi^i_j|}$.

5.3.1.2 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: $X = x_n.....x_0$, and $Y = y_m.....y_0$, and assuming $m \geq n$, 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 that the right-most part of the domain name is more relevant.

- $d_{lev}(x, y)$: the normalized Levenshtein or edit distance between two strings x, y. The distance calculates the minimum number of edits (insertion, deleteion, 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:

$$d_\phi(\phi_i, \phi_j) = \frac{1}{3}\left(2 \cdot \sqrt{(\overline{|\phi_i|} - \overline{|\phi_j|})^2} + d_{\mathcal{H}}(\phi_i, \phi_j)\right) \tag{5.1}$$

  where $d_{\mathcal{H}}$ 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

$$d_{\mathcal{H}}(\phi_1, \phi_2) = \max(I_{\mathcal{H}}(\phi_1, \phi_2), I_{\mathcal{H}}(\phi_2, \phi_1)) \tag{5.2}$$

  where

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

  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:

$$d_\varphi(\varphi_1, \varphi_2) = \frac{1}{6}(d_t(t_1, t_2), d_{euclid}(r_1, r_2), d_u(u_1, u_2), 3 \cdot d_\phi(\phi_1, \phi_2)) \tag{5.3}$$

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

$$d(q(id_1), q(id_2)) = \frac{1}{2}\left(\sqrt{(\overline{|\varphi^1|} - \overline{|\varphi^2|})^2} + d_{\mathcal{H}}(\varphi^1, \varphi^2)\right) \tag{5.4}$$

where $d_{\mathcal{H}}$ 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.

5.3.1.3 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 [74].

The query strings distance from Eq. (5.1) 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. (5.2). 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. Also note that the Hausdorff distance has more weight than the distance between number of queries (double weight). As we consider that the similarity of the string itself is more important than the length of the query. Moreover, when we consider the distance between single queries in Eq. (5.3), the distance between the query strings has more weight, thus prevailing over the the other parts.

A similar approach is used to compute the final distance between two users in Eq. (5.4), where we use the Hausdorff distance between the set of queries of each user 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 5.3.1.4. 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 on 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 user anonymization and protection at syntactic level. A semantic approach could lead to other interesting results which are to be explored.

5.3.1.4 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:

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

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:

$$\begin{aligned}\mathbb{C}_\varphi(\varphi^1, \ldots \varphi^k) &= \mathbb{C}_\varphi\left((\varphi_1^1, \ldots, \varphi_{|\varphi^1|}^1), \ldots, (\varphi_1^k, \ldots, \varphi_{|\varphi^k|}^k)\right) \\ &= \varphi^* \\ &= (\varphi_1^*, \ldots, \varphi_{|\varphi^*|}^*)\end{aligned}$$

The centroid φ^* is composed of queries from the cluster queries φ^i for $i = \{1 \ldots k\}$. 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, $\varphi^{*,i} = (\varphi_1^{*,i}, \ldots, \varphi_{|\varphi^{*,i}|}^{*,i})$, 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,

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

where,

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

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 rightmost common part (sub-domain).

5.3.2 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.

5.3.2.1 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.

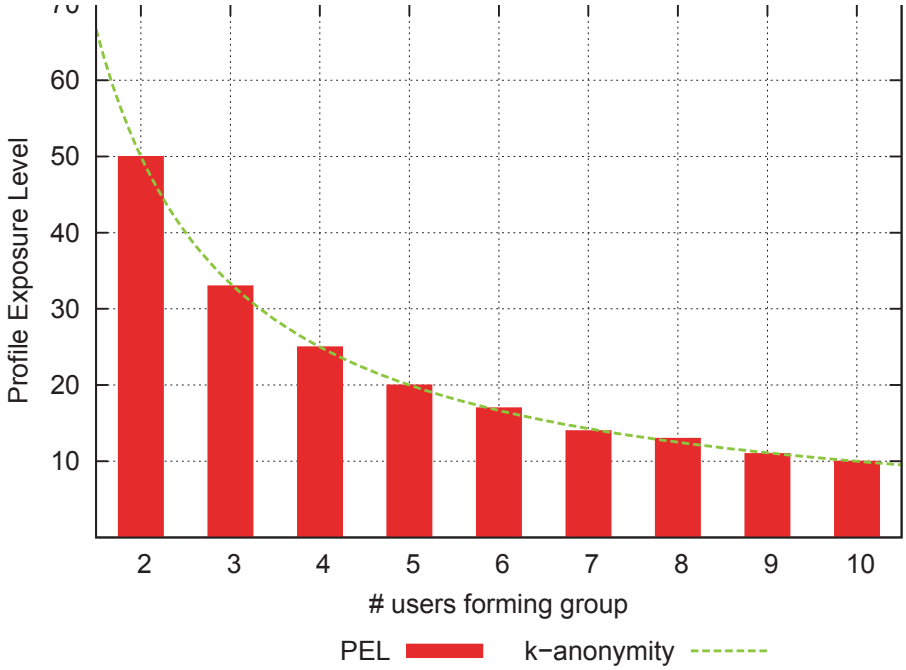


FIGURE 5.3: The average PEL for $k \in \{2, \ldots, 10\}$.

In order to verify that our method protects the users' queries obtaining k-anonymity, we have used the *Profile Exposure Level* (see Section 4.5).

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 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 $k \in \{2, \ldots, 50\}$. Then we have computed PEL for each user and value of k.

Figure 5.3 shows the theoretical level of privacy for $k = \{2, \ldots, 10\}$, 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.

5.3.2.2 Information loss

The first usability measure for categorical data was presented in [31]. The authors proposed an entropy-based measure to evaluate the information loss in SDC. In the same line, [63] proposed to measure the information loss as:

$$ILR = \frac{|original_entropy - new_entropy|}{original_entropy} \cdot 100$$

We have used the ILR (Information Loss Ratio) to evaluate the utility of our proposal with the same files obtained previously when we calculated PEL. Thus, we have 1000 users and their original queries, and for every $k \in \{2, \ldots, 50\}$ the protected queries for every user. Figure 5.4 shows that the data utility broke down according to the parameter k of the microaggregation. It can be observed that when k increases (the number of user per cluster thus increases), the information loss of the user also grows. Note that minor fluctiations are due to small and expected perturbations in the microaggregation process.

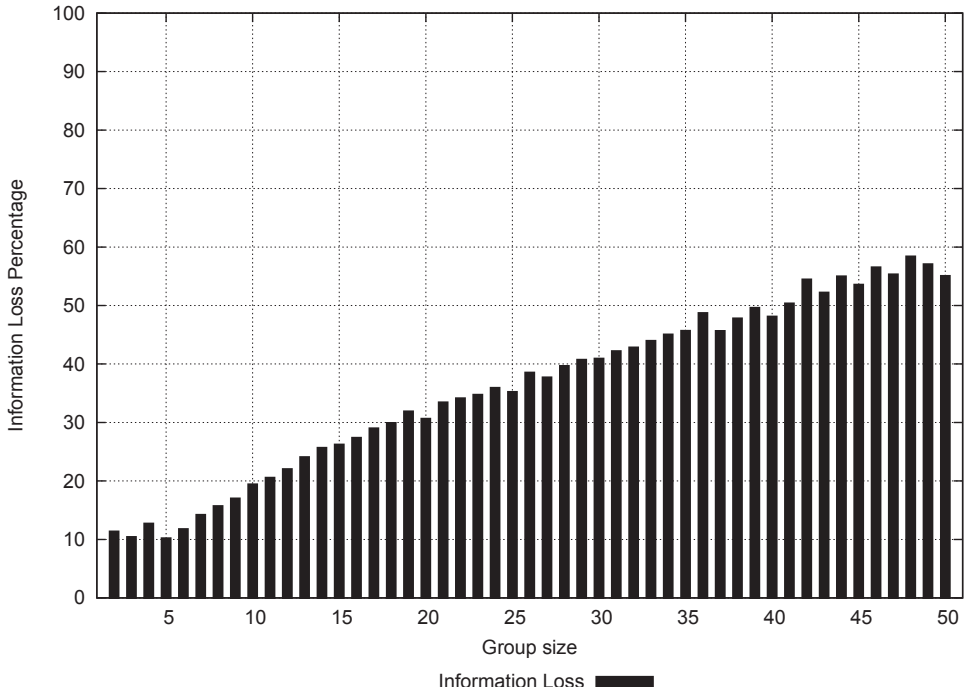


FIGURE 5.4: The average ILR for $k \in \{2, \ldots, 50\}$.

5.3.2.3 On the relation of PEL and IRL

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

By using a greater k, we have a high information loss (ILR), as we can see in Figure 5.4. For $k = 35$ we lose the 50% of the users' information, and for $k = 3$ we lose only the 10%. The minimum ILR provides the most data utility, so we should select the smallest values of k.

Nonetheless, as it is shown in Figure 5.3, 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, greater k would be recommendable to protect the privacy of the users. However, we consider that a user's profile has enough protection if PEL is at least of 40%, i.e. $k = 3$ (see [39]).

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 an 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 the overall 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 [80].

5.3.2.4 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 [9, 10, 5]. Although the clustering of query logs is normally performed with some customized and more elaborated clustering, we show our results in 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 relying on the distance and aggregation functions described in Sections 5.3.1.2 and 5.3.1.4.

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

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

Figure 5.5 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 $\kappa = \{2 \dots 500\}$. 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 5.3.2.1, and 5.3.2.2, as this k increments, we achieve more privacy but less utility. Given a protected dataset with $k = 3$ and another with $k = 50$, Figure 5.6 shows the difference between the clusters of the two datasets, as the k-means κ increases. The straigh line denotes de difference between the Jaccard index of both datasets, and the doted line denotes the difference between the Rand index.

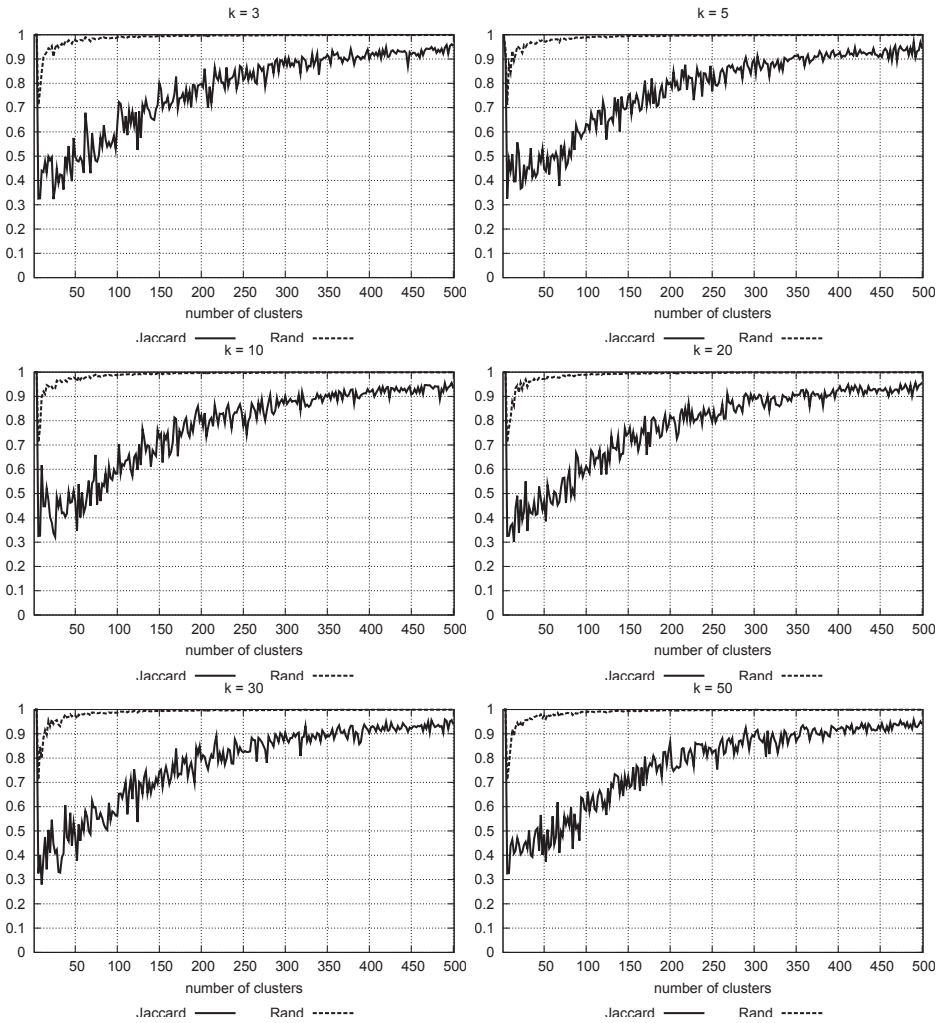


FIGURE 5.5: Rand and Jaccard indexes for aggregated data with $k \in \{3, 5, 10, 20, 30, 50\}$.

However, the values for the indexes in Figure 5.5 seem similar. We can see in Figure 5.6 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 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.

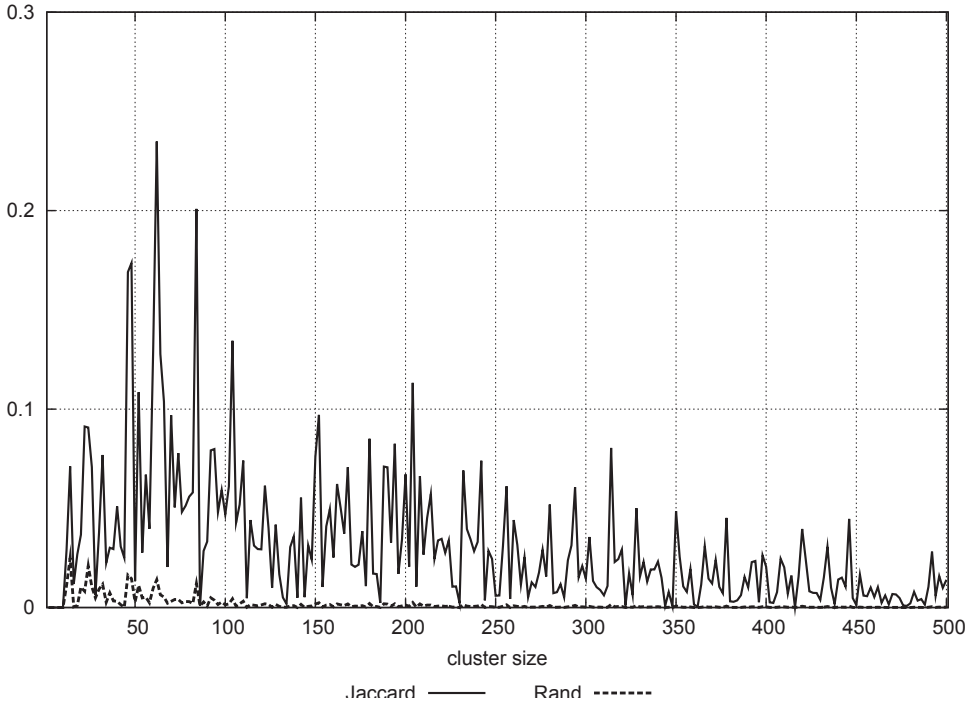


FIGURE 5.6: Difference of the Jaccard and Rand indexes for data microaggregated with $k = 3$ and $k = 50$.

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 min-

ing applications where our protection method will provide good results are in fact those making use of some clustering technique: categorization, classification, ...

5.3.2.5 Frequency analysis

We have also analyzed the frequency of queries in the protected data. Figure 5.7 shows the frequency of the ten 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.

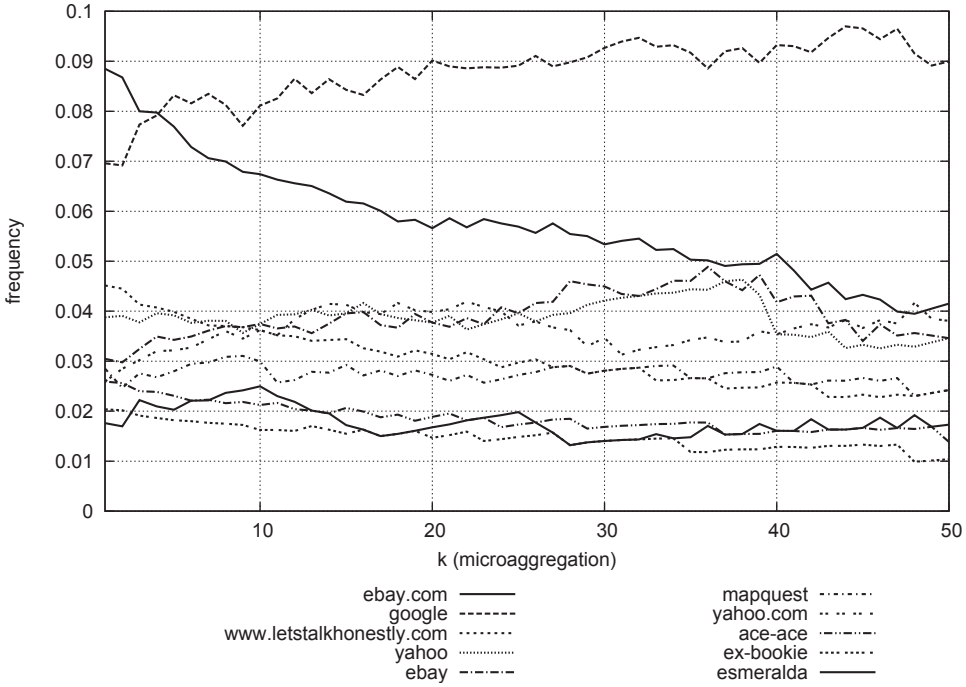


FIGURE 5.7: Query frequency analysis.

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

If we take a look at the frequency of single words, excluding common stop words, we see a similar result. As shown in Figure 5.8, frequency of words

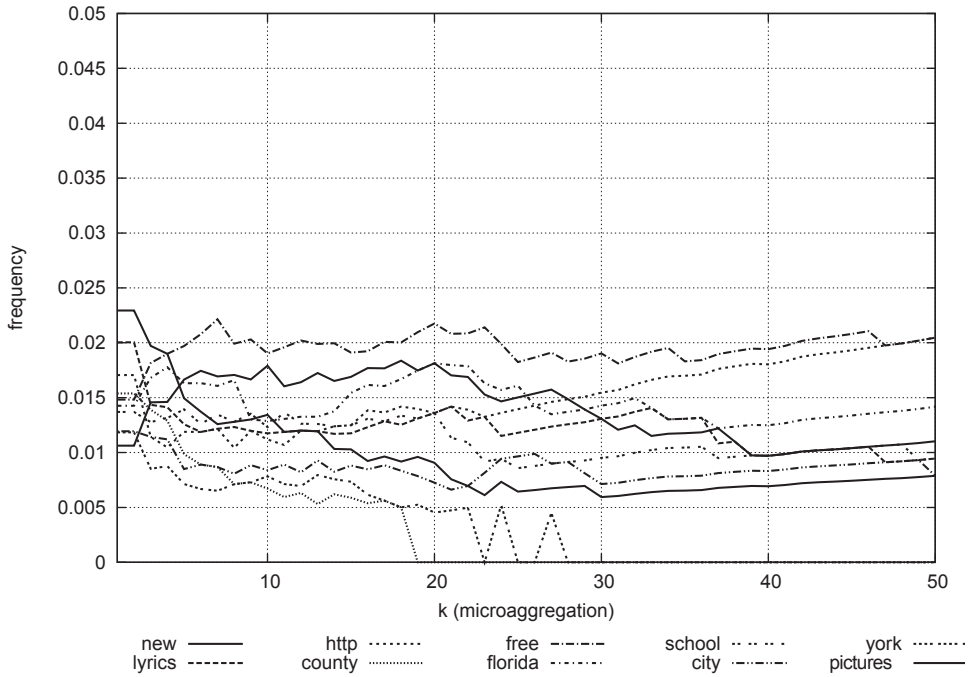


FIGURE 5.8: Word frequency.

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.

This kind of studies are normally 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 an even better preservation in bigger datasets.

5.3.3 Conclusions

To provide microaggregation at a user level, we have defined a new user distance and aggregation operator. The user aggregation described in Section 5.3.1.4 was designed in order to be as computationally efficient as pos-

sible. 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 first approach given the complexity that the second one imposes, and also because it already produced satisfactory results as show in Section 5.3.2.

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 it 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 the information of the original logs to some extent.

5.3.3.1 Publications

- G. Navarro-Arribas, V. Torra, A. Erola and J. Castellà-Roca, *User k-anonymity for privacy preserving data mining of query logs*, Information Processing and Management, Vol. 48, no. 3, pp. 476-487, May 2012, ISSN: 0306-4573.

5.4 Semantic microaggregation of search logs

Section 5.3 presented a method to provide k-anonymity in search logs by microaggregation. While preventing the identity disclosure, the microaggregation based on the aggregation of several distance functions minimizes the information loss. However, the system is hampered by the evaluation of query similarities, which is solely based on spelling and syntactical matching.

As we mention before, query logs have variable length and high dimensionality, compared to the relatively few attributes and values in relational records. Datasets with these characteristics are usually referred as set-valued data [102]. The management of set-valued data with textual values, arises new challenges that are not considered by methods focused on numerical or categorical data [66, 107]. Contrary to numerical data, which can be compared and transformed by means of mathematical operators (so that the anonymization can preserve statistical properties of the dataset), textual data require operators of aggregation and comparison that take the semantics of the text into account. As acknowledged by several authors [67, 107], it is crucial to preserve the semantics of textual data during the anonymization process in order to maintain their utility.

As a example, consider the concepts *flu, golf, hypertension, football* and a microaggregation method with $k = 2$. If the similarity function takes into account the semantics of the concepts, the two clusters *[flu, hypertension]* and *[golf, football]* can be created. On the contrary, if the similarity function takes into account the length of the concepts, the two clusters *[flu, golf]* and *[hypertension, football]* can be created. The first pair can be generalized by *illnesses* and *sports*, while the second one cannot be generalized without providing useless results, i.e. the resulting data is too distorted.

5.4.1 Towards a semantic interpretation of query logs

None of the mentioned methods in section 3.2 consider the semantics of queries during the anonymization process: they execute random or distribution-based transformations over data. In order to maximize the utility of anonymized logs, we need to pay special attention to the preservation of their semantics [67, 107]. To that end, we first need to map queries with their formal semantics. However, because semantics is an inherent human feature, the interpretation of textual data requires the exploitation of some sort of human-tailored machine-readable knowledge source. This allows mapping between words and their conceptual abstractions, analyzing the latter according to their semantic interrelations. Taxonomies, folksonomies and more general ontologies [44] have proven their usefulness as structured knowledge sources, thus enabling the interpretation of text at semantic level.

We propose a semantic microaggregation method that maps queries with their formal semantics using the Open Directory Project (ODP) [77] as knowledge base. The rationale behind our proposal is simple: the aggregation of users (users' logs) with more common interests minimizes the information loss.

5.4.1.1 ODP

ODP is the most widely distributed database of Web content classified by humans. ODP data powers the core directory services for some of the most popular portals and search engines on the Web, including AOL Search, Netscape Search, Google, Lycos, and HotBot, and hundreds of others. Thus, a query result using them is hardly influenced by the ODP classification.

ODP uses a hierarchical ontology structure to classify sites according to their themes. For example, when we search for *Barcelona FC*, ODP returns a list of categories where the query belongs to (Figure 5.9) to. Each result starts with a root category followed by deeper categories in the ODP tree.

```
Open Directory Categories  (1-5 of 5)

   1. Sports: Soccer: UEFA: Spain: Clubs: Barcelona   (11 matches)
   2. World: Polski: Sport: Sporty pilki i siatki: Pilka nozna: Kluby:  (...)
   3. World: Espanol: Regional: Europa: Espaa: Deportes y tiempo libre:  (...)
   4. World: Deutsch: Sport: Ballsport: Fuball: Vereine: Spanien   (3)
   5. World: Francais: Sports: Balles et ballons: Football: Rgional:  (...)
```

FIGURE 5.9: Example of ODP query result.

5.4.1.2 An ODP similarity measure

In order to be able to microaggregate users from the query logs, we have to define a distance or similarity measure between users. We introduce a similarity coefficient based on the common categories shared between queries from each user. We also introduce some notation here to formalize the process.

We consider the set of n users $U = \{u_1, \ldots, u_n\}$ from the query log, and their respective set of queries $Q = \{Q_1, \ldots, Q_n\}$, where $Q_i = \{q_1^i, \ldots, q_{m_i}^i\}$ are the queries of the user u_i. Each query q_j^i has several terms $q_j^i = \{t_1, \ldots, t_{r_j}\}$.

Given a term t_s, we can obtain its classification in ODP at a given depth. When querying ODP, the returned categories can be divided in depth levels. Let l be the parameter that identifies the depth level in the ODP hierarchy. For example, if we have the classification $Sports : Soccer : UEFA : Spain : Clubs : Barcelona$ and $l = 1$, we only work with the root category $Sports$; when $l = 2$ we work with $Sports : Soccer$; and so on. We will consider a maximum depth L to restrict the search space, so $l \in \{1, \ldots, L\}$.

We denote as $C_l = \{c_1^l, \ldots, c_{p_l}^l\}$ the set of possible categories at level l in the ODP. Given a user u_i we can obtain all the categories at level l from all queries of the user. We denote the set of categories for user u_i at level l as $C_l(u_i)$. Note that considering all the queries of user u_i, $Q_i = \{q_1^i, \ldots, q_{m_i}^i\}$, and their respective sets of terms $q_j^i = \{t_1, \ldots, t_{r_j}\}$ for $j = 1 \ldots m_i$, the number of categories for user u_i at level l is given by $|C_l(u_i)| = r_1 + \ldots + r_{m_i}$.

We can then define a similarity coefficient ODP_{sim} between two given users u_i and u_j as:

$$OPD_{sim}(u_i, u_j) = \sum_{l=1}^{L} \{|c_l| : c_l \in \{C_l(u_i) \cap C_l(u_j)\}\} \tag{5.5}$$

This similarity coefficient between two users computes the common categories between them for all considered levels, that is, levels up to L. Note that OPD_{sim} is symmetric and ranges from 0 (there is no similarity between the users) to $\sum_{l=1}^{L} |C_l|$ (maximum similarity between two users).

5.4.2 ODP-based microaggregation of query logs

The method we propose to protect query logs is a microaggregation that follows the outline of Section 5.4.1 with an extra step of data preparation. Our approach consists of the following steps:

1. Data preparation.
2. Partition.
3. Aggregation.

These steps are described in detail in the following sections.

5.4.3 Data preparation

To ease the computation of the protected data, the data are prepared by pre-querying ODP to classify the user queries. Following the notation introduced in Section 5.4.1.2, for every term t_s, we can obtain its classification for all levels $l \in \{1, \ldots, L\}$ using ODP. This allows us to obtain all the categories associated to all the users in all levels, that is, $C_l(u_i)$ for all user $u_i \in U$, and all considered levels. Next, we create a *classification matrix* that contains

the number of queries for each user and category at level l, $M_{U\times C_l}$. Note that we obtain one matrix for every level $l \in \{1,\ldots,L\}$. So, $M_{U\times C_l}(i,j)$ is the number of times that category c_j^l is found in the queries of user u_i.

Finally, we use the $M_{U\times C_l}$ matrices in order to compute the *incidence matrix* that contains the semantic similiarity of the users $M_{U\times U}$. Given the incidence matrix $M_{U\times U}$, $M_{U\times U}(i,j)$ is the number of common categories between users u_i, and u_j for all depth levels $l \in \{1,\ldots,L\}$. Moreover, note that the incidence matrix corresponds to the similarity coefficient described in Section 5.4.1.2, that is, $M_{U\times U}(i,j) = ODP_{sim}(u_i,u_j)$.

The process works as follows:

1. Obtain the classification matrices $M_{U\times C_l}$ using Algorithm 2.
2. Obtain the incidence matrix $M_{U\times U}$ using Algorithm 3, i.e. the similarity coefficient between users.

5.4.3.1 Partition

The partition step creates groups of k users with similar interests using Algorithm 4.

Let us assume that u_i and u_ρ are the most similar users in the set. We calculate the users' similarity ODP_{sim} using the incidence matrix $M_{U\times U}$, (see Section 5.4.3). The most similar users are those that have the highest similarity coefficient in the matrix. Next, we include u_i and u_ρ to the cluster. If the group size k is two, we delete u_i and u_ρ records from the incidence matrix and we repeat the process to obtain a new cluster. When the group size is bigger than two, we merge the columns and rows of u_i and u_ρ creating a new user u'. u' is the addition of both users, u_i and u_ρ. Let us assume, that u_ξ is the most similar user to u'. Next, we include u_ξ to the cluster with u_i and u_ρ. The method executes this process $k-2$ times.

Algorithm 2 Algorithm for computing the classification matrices $M^L_{U\times C}$ where $L = \{1, \ldots, l\}$

Require: the maximum depth L for the ODP categories
Require: the set of users $U = \{u_i, \ldots, u_n\}$
Require: the set of queries $Q_i = \{q^i_1, \ldots, q^i_{m_i}\}$ of each user u_i
Require: the set of terms $\{t_1, \ldots, t_{r_j}\}$ of each query q_j
Ensure: $\{M_{U\times C_1}, \ldots, M_{U\times C_L}\}$, i.e. for every level l, the matrix $M_{U\times C_l}$ with the number of queries for each category and user in the depth l

```
for l ∈ {1, ..., L} do
  for u_i ∈ {u_1, ..., u_n} do
    for q^i_j ∈ Q_i = {q^i_1, ..., q^i_{m_i}} do
      for t_s ∈ q^i_j = {t_1, ..., t_{r_j}} do
        obtain the categories c_t at depth l for the term t_s using ODP;
        for each c_t do
          if c_t ∈ M_{U×C_l} then
            M_{U×C_l}(u_i, c_t) = M_{U×C_l}(u_i, c_t) + 1;
          else
            add the column c_t to M_{U×C_l};
            M_{U×C_l}(u_i, c_t) = 1;
          end if
        end for
      end for
    end for
  end for
end for
return {M_{U×C_1}, ..., M_{U×C_L}}.
```

Algorithm 3 Algorithm for computing the incidence matrix $M_{U\times U}$

Require: the calssification matrices $\{M_{U\times C_1}, \ldots, M_{U\times C_L}\}$
Ensure: $M_{U\times U}$
  Initialize $M_{U\times U}(i,j) \leftarrow 0$ for all $i,j = 1 \ldots n$;
  for $M_{U\times C_l} \in \{M_{U\times C_1}, \ldots, M_{U\times C_L}\}$ **do**
    for each column $c_j \in M_{U\times C_l}$ **do**
      for each row $u_i \in M_{U\times C_l}$ **do**
        for each row $u_\rho \in M_{U\times C_l}$ **do**
          $M_{U\times U}(u_i, u_\rho) \leftarrow M_{U\times U}(u_i, u_\rho) +$
          $\min(M_{U\times C_l}(u_i, c_j), M_{U\times C_l}(u_\rho, c_j))$;
        end for
      end for
    end for
  end for
  return $M_{U\times U}$.

Algorithm 4 Algorithm for computing the clusters $Z = \{z_1, \ldots, z_\gamma\}$ of users

Require: the set of users $U = \{u_1, \ldots, u_n\}$
Require: the incidence matrix $M_{U\times U}$
Require: the clusters size k
Ensure: the clusters $Z = \{z_1, \ldots, z_\gamma\}$ of users for $\gamma = \lceil n/k \rceil$
  $M'_{U\times U} \leftarrow M_{U\times U}$;
  $U' \leftarrow U$;
  while $|U'| \leq k$ **do**
    obtain the cluster z of k users using the Algorithm 5 and $M'_{U\times U}$;
    remove the users $u_i \in z$ form U';
    remove the columns and the rows of the users $u_i \in z$ form $M'_{U\times U}$;
    add z to the set Z;
  end while
  return $Z = \{z_1, \ldots, z_\gamma\}$.

Algorithm 5 Algorithm for computing a cluster z of k users

Require: a incidence matrix $M'_{U\times U}$
Require: the clusters size k
Ensure: a cluster z of k users
  $z \leftarrow \emptyset$;
  obtain the two most similar users (u_i, u_ρ), i.e. the cell of $M'_{U\times U}$ with the highest value;
  add (u_i, u_ρ) to the set z;
  while $(|z| < k)$ **and** $(columns(M'_{U\times U}) > 0)$ **do**
    for each column $c_s \in M'_{U\times U}$ **do**
      $M'_{U\times U}(c_s, u_\rho) = M'_{U\times U}(c_s, u_\rho) + M'_{U\times U}(c_s, u_i)$;
    end for
    for each row $r_s \in M'_{U\times U}$ **do**
      $M'_{U\times U}(u_i, r_s) = M'_{U\times U}(u_i, r_s) + M'_{U\times U}(u_\rho, r_s)$;
    end for
    delete the column u_ρ of matrix $M'_{U\times U}$;
    delete the row u_ρ of matrix $M'_{U\times U}$;
    obtain the new u_i's most similar user u_ρ, i.e. the cell of the user u_i with the highest value;
    add u_ρ to the set z;
  end while
  return z.

5.4.3.2 Aggregation

For every cluster z_j formed in the partition step, we compute its aggregation by selecting specific queries from each user in the group. That is, given the cluster of users $z_j = \{u_1, \ldots, u_k\}$, we obtain a new user u_{z_j} as the representative (or centroid) of the cluster, which summarizes the queries of all the users of the cluster. The selection of queries is based on the following principles:

1. We give priority to queries semantically close between them.
2. The number of queries that a user contributes to the cluster representative is proportional to the number of queries of the user.

The first principle is considered in the partition step described in Section 5.4.3.1, since clusters are composed of users with semantically similar queries. The second principle is formalized defining some indexes as described below.

First, the number of queries of the centroid is the average of the number of queries of each user u_i of the cluster z_j. Then, the contribution of a user u_i ($Contrib_i$) to the centroid of a cluster with k users depends on her number of queries $|Qi|$. This contribution is carried out follows:

$$Contrib_i = \frac{|Qi|}{\sum_{i=1}^{k} |Qi|} \tag{5.6}$$

Thus, the quota of each user u_i in the new centroid u_{z_j} can be computed as:

$$Quota_i = \frac{|Qi|}{k} \tag{5.7}$$

More formally, the aggregation method runs Algorithm 6 for each cluster. First, it sorts logs from all users descending by query repetitions. Then, for each user u_i of the cluster and while not reaching $Quota_i$:

1. Add the first query of her sorted list with a probability $Contrib_i \times \#q_j_repetitions$. For example, if u_i has a query which is repeated 3 times, and $Contrib_i$ is 0.4, as $3 \cdot 0.4 = 1.2$, the method adds one query to the new log and then randomly chooses whether to add it again or not according to the presence probability 0.2.

2. Delete the first query of the list.

Algorithm 6 Algorithm to aggregate the k users of the cluster z

Require: a cluster z of k users
Require: the quota $Quota_i$ of each user of the cluster z
Require: the contribution $Contrib_i$ of each user of the cluster z
Require: the set of queries Q_i of each user of the cluster z
Require: the queries list SL
Require: the microagregged log ML
Ensure: the centroid of the cluster z
  $ML \leftarrow \emptyset$
  for each user $u_i \in z$ **do**
    $SL \leftarrow$ sort $Q_i = \{q_1^i, \ldots, q_{m_i}^i\}$ by query repetitions.
    while not reach $Quota_i$ **do**
      Add the first query q_1^i with a probability $Contrib_i \times \#q_1^i_repetitions$ to ML.
      Delete q_1^i of SL.
    end while
  end for
  return ML.

5.4.4 Evaluation

We have tested our microaggregation method using real data from the AOL logs released in 2006, which correspond to the queries performed by 650 000 users over three months. We randomly selected 1 000 users, which correspond to 55 666 lines of query logs. The usefulness evaluation and the results are presented below.

5.4.4.1 Usefulness Evaluation Method

For each user we have her original set of queries and the corresponding protected ones by means of our microaggregation method. All queries can be classified into categories, i.e., each query is classified into the L first depth levels of the ODP.

In order to verify that our method preserves the usefulness of the data (i.e. it does not introduce too much perturbation), we count the number of queries of each category for a given level l, which are in the original log as well as in the centroid, ρ. This number is divided by the number of original queries in l, χ, thus obtaining a *semantic remain percentage* (SRP) in the level.

$$SRP = \frac{\rho}{\chi} \tag{5.8}$$

To summarize, our evaluation method does not only match two equal terms in both logs, but also a term in the protected log that replaces one with closest semantic in the original log. By using a random partition algorithm, users of each cluster might not be semantically close.

Consider, as an example of the worst case, a cluster of k users $\{u_1, \ldots, u_k\}$ with respective queries $Q = \{Q_1, \ldots, Q_k\}$, such that $Q_i \cap Q_j = \emptyset$ for all $i \neq j$. Thus, only the queries of a single user in a specific topic will appear in the centroid.

In this case, the number of queries of u_i that appear in the centroid can be calculated using formula 5.7 and it is known that the sum of all quotas is χ. Therefore, in the worst case, i.e., when no common interests between users exists, we can calculate the average SRP as:

$$\frac{\sum_{i=1}^{k} \frac{|Q_i|}{\chi}}{k} = \frac{1}{k} \tag{5.9}$$

5.4.4.2 Results

As discussed in Section 5.4.1.1, ODP returns a list of categories for every term (or query), and each category is composed of various hierarchical levels. In our method, one or all categories can be used and, for each category, either all hierarchical levels or some of them can be considered. Intuitively, the more categories and levels (deeper levels) that are used, the higher the computational cost should be, and, perhaps, a better SRP can be achieved. Thus, we want to study how these parameters influence the SRP and the computational cost:

- ODP levels: every term has a categorization up to a hierarchical level, and the deepest level can be different for every term. The deeper the level is, the less terms that have information in this level there will be. We want to know the deepest level that gives information for a majority of terms.
- SRP vs. ODP-categories: we want to know the SRP value when we use more or less categories, i.e., if we use more categories, the SRP can be either higher, or have approximately the same SRP.
- Computational cost vs. ODP-categories: Supposing that more categories are used, the higher the computational cost will be, but the extra cost should be known. If the extra cost is not significant and a better SRP is obtained, more categories can be used.

5.4.4.3 ODP levels

In ODP, not all terms rank up to a certain level. For example, our working set of queries has terms with two levels (minimum) and others with twelve levels (maximum). In the study of the above mentioned relations (SRP vs. ODP-categories and computational-cost vs. ODP-levels), levels that do not have a ranking for the majority of terms can be ignored because such levels

only give information to improve the SRP for a reduced number of terms. Thus, we consider a level if it has information for, at least, the 50% of the terms (queries).

In this sense, we have calculated for every level the percentage of terms that have a result for the level, and Figure 5.10 shows the percentage of queries (our working set of queries) that can be classified up to a certain depth level in the ODP tree. It can be observed that only 57% of queries can be classified up to level 5. So, we only run tests up to this level.

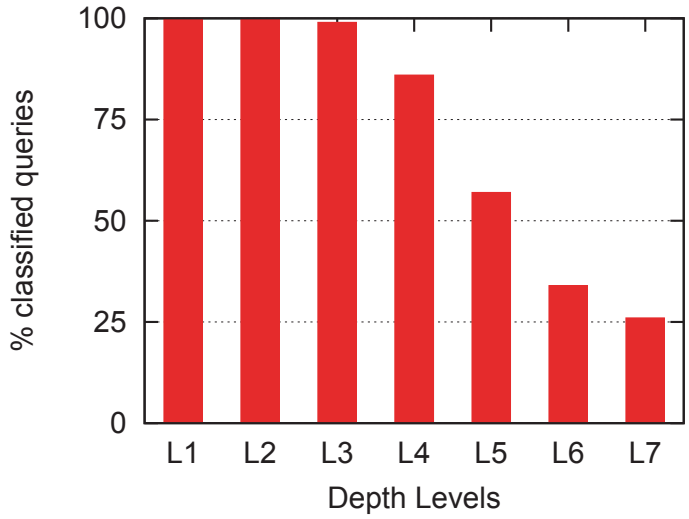


FIGURE 5.10: Percentage of queries that can be classified up to a certain level in ODP

5.4.4.4 SRP vs. ODP-categories

Besides some initial tests [38], we have calculated the percentage of semantically similar queries as the accumulation of levels, i.e., add the coincidences of level 1 and 2 to calculate the percentage of semantically similar queries at level 2. In this current work, we have changed the evaluation method because we think that evaluating each level separately is better to understand the remaining similarity of the queries in that level.

We have compared the results obtained (SRP) by either using the first five categories returned by the ODP or using only the first one. The range is sufficient in order to evaluate the SRP behavior when we use more categories. Note that the first category that gives ODP is the most significative for the introduced term. Figure 5.11 shows, for cluster sizes 2, 3, 4 and 5, the average SRP that users obtain for various levels L. The dark color represents the obtained results using the first category returned by the ODP, and the light color represents the obtained results using the first five categories.

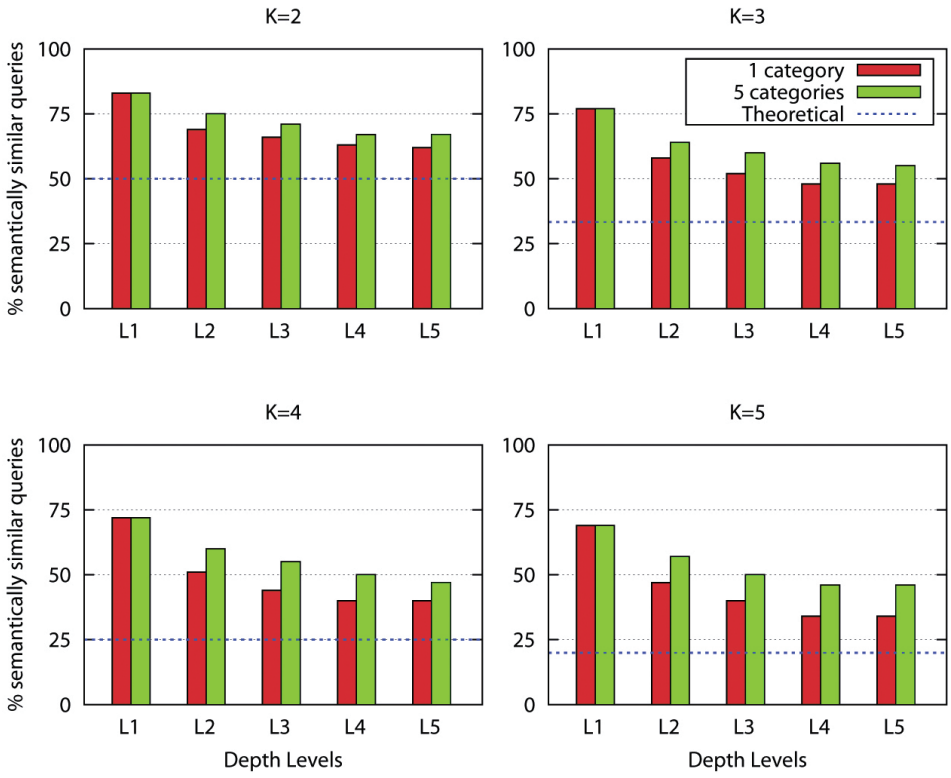


FIGURE 5.11: Semantic similarity percentage of microaggregated logs using either the first category or the five first categories returned by the ODP.

It can be observed that both tests improve the theoretical SRP (see Section 5.4.4) with all depth levels. By using more categories in the ODP classification, we achieve less similarity loss for deeper levels and larger clus-

ter sizes. For instance, when $L = 1$, the same gain is obtained in all cases, but when $L = 5$ and $k = 5$, the difference gain is approximately 10% using the first five categories instead of only the first one.

5.4.4.5 Computational cost vs. ODP-categories

The computation cost is larger when more categories are used. Figure 5.12 shows the average time required to microaggregate logs for cluster sizes $k = \{2, \ldots, 5\}$ for various levels. It can be determined that by using the first five ODP categories, the average time is three times larger than using only the first one.

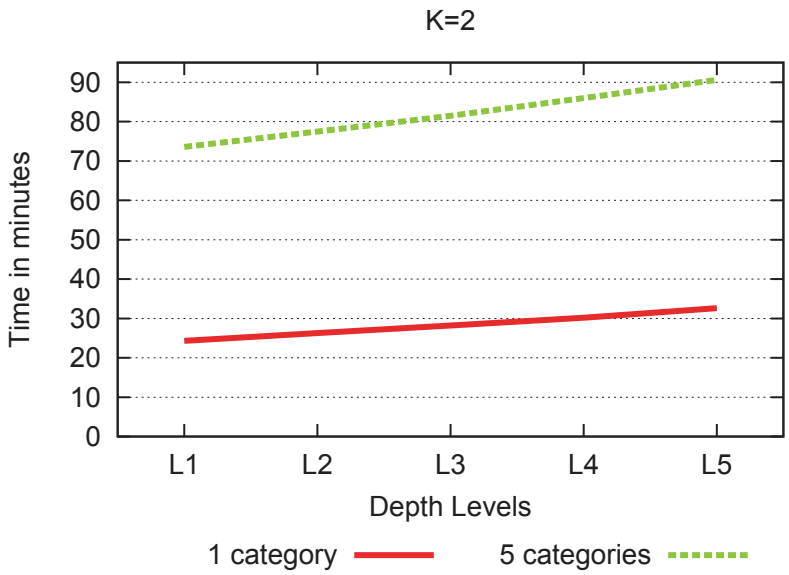


FIGURE 5.12: Average required time to microaggregate logs using our method for various ODP levels.

Tests were run on a Pentium Core 2 Duo 2.2Ghz without source code parallelization. Figure 5.12 demonstrates that the required time increases linearly with the number of user queries. Nonetheless, the program could be parallelized as follows:

- **Data preparation:** as each user has her queries, the classification matrices $M_{U \times C}$ can be computed simultaneously. Then, each cell of the incidence matrix $M_{U \times U}$ can be calculated independently, since the classification matrix of each user is available.
- **Partition:** the partition process is linear and cannot be parallelized, but it is a negligible part of the whole process. The time required for its calculation represents less than one percent of the total time.
- **Aggregation:** as users are divided into k groups, the logs' aggregation of each group can be run simultaneously.

Thus, the program parallelization could make the proposal scalable for very-large systems.

5.4.5 Conclusions

The existing microaggregation techniques for query logs do not usually take the semantic proximity between users into account, which is negatively reflected in the usefulness of the resulting data. We have presented a new microaggregation method for query logs, based on a semantic clustering algorithm. To that end, we use ODP as knowledge base to interpret the queries' terms and its hierarchy as metric space to define a distance operator. Aggregation is performed selecting randomly queries inside the same cluster.

We have tested our proposal using real query logs from AOL. As we have seen, we obtain good results, both in terms of information loss and in terms of protection, which is guaranteed because our method ensures k-anonymity at user level.

It also should be taken into consideration that we are working with a set of $1,000$ users, randomly selected from the AOL files. We expect to achieve

greater SRP values working with a larger set, because more similar users may be grouped.

The utility of the data is calculated according to a discrete function which evaluates the relation between two queries in our metric space. The utility obtained using one category per query or five categories per query, while closer in some levels, shows that the latter minimizes information loss. We assume that it is the effect of a biased queries' interpretation, and as much meanings we have for a query, the easier to find relations is.

5.4.5.1 Publications

- A. Erola, J. Castellà-Roca, G. Navarro-Arribas and V. Torra, *Semantic Microaggregation for the Anonymization of Query Logs*, Lecture Notes in Computer Science, Vol. 6344 (Privacy in Statistical Databases-PSD 2010), pp. 127-137, Sep 2010, ISSN: 0302-9743.
- A. Erola, J. Castellà-Roca, G. Navarro-Arribas and V. Torra, *Semantic microaggregation for the anonymization of query logs using the open directory project*, SORT-Statistics and Operations Research Transactions, Vol. 0, Special issue, pp. 41-58, Sep 2011, ISSN: 1696-2281

5.5 Semantic microaggregation optimization

Section 5.4 presented a semantic aggregation method that uses the Open Directory Project (ODP) as knowledge source [38, 37]. To that end, we look for each word of each query in ODP and retrieve the categories to which they belong (word's semantics). However, even though this method compares queries at a conceptual level (i.e. according to their categories), it fails to retain the meaning of the *complex queries* with several words or noun phrases, inasmuch queries are processed word by word (e.g. *water sports* was mapped to the concepts *water* and *sport*). In [46], even thought the author uses WordNet as knowledge base, similar problem occurs. The omission of this relation between query's words can results in a loss of semantics [68].

However, *complex queries* cannot be directly mapped to concepts in a knowledge base. Hence, in order to reduce the information loss resulting from query removal, we propose to apply several linguistic analyses to query logs which improve the recall and accuracy of the conceptual mapping.

5.5.1 Query anonymization method

In this section, we present a query log anonymization method which carefully considers the semantics of all kind of queries (i.e. from one-word to multiple noun phrases containing proper nouns). To that end, we adapt the MDAV microaggregation algorithm in order to coherently deal with set-valued datasets like query logs. We take special care in minimizing the disclosure risk while retaining their utility of the anonymized query logs.

Given a set of records, each one corresponding to the set of queries performed by each user, the basic steps of the method are:

1. *Query processing and conceptual mapping*: in order to semantically interpret textual queries, these are processed so that syntactical con-

structions (i.e. noun phrases) can be mapped to their conceptual abstractions modeled in a knowledge base.

2. *Semantic data partition*: clusters of query logs of at least k-users are created (fulfilling k-anonymity) by means of the MDAV microaggregation algorithm. The cluster construction process and the centroid calculus method, on which the MDAV method relies, have been adapted in order to consider query semantics and the distributional properties of set-valued data.

3. *Semantic query anonymization*: clustered query logs are replaced by a synthetic set of queries that represent both their meaning and their distribution. Synthetic query logs are constructed minimizing the information loss and the disclosure risk owing to the replacement.

5.5.1.1 Query processing and conceptual mapping

In order to semantically interpret users' queries, we need to map them to their formal semantics (i.e. conceptual abstractions) in a background knowledge base. Users' queries, being free text strings, could be problematic to manage. They may directly correspond to an individual concept (e.g. *computers*), to a specialization (e.g. *Apple computer*), but also to a concatenation of several concepts, within a syntactically coherent sentence (e.g. *stores offering bargain Apple computers*), or even a raw list of unconnected terms (e.g. *computers Apple store bargain*). These last examples of *complex queries*, which are usually performed by users of web search engines [64], cannot be directly mapped to concepts in a knowledge base.

Works dealing with query logs have not deeply addressed the analysis of *complex queries* [93]. Some of them [110] simply avoid queries than cannot be directly found in the knowledge base. This reduces the posterior analysis and anonymization to queries with single terms (e.g. *computer*) or simple noun phrases (e.g. *cell phone*). Other works [46, 37] simply extract individual words from complex queries, mapping each one to their corresponding

concept. These methods fail to properly interpret noun phrases composed by several words (e.g. the meaning of *water sports* is different to *water* + *sports*).

Accordingly, we propose to apply several linguistic analyses to query logs in order to improve the recall and accuracy of the conceptual mapping. This will help to better characterize the users in the posterior anonymization and, hence, to better retain data utility.

Query processing Let $U = \{u_1, \ldots, u_m\}$ be the set of users represented by their query logs, and let $u_r = \{q_1, \ldots, q_p\}$ be the queries extracted from the query log of the user u_r.

Each query, q_j, is morpho-syntactically analyzed to extract semantic units. A semantic unit is a piece of text that refers to a unique concept. We focus our action on Noun Phrases (NPs) which consist of a set of words in which at least one of them (i.e. the one most on the right) is a noun. This noun, which corresponds to a concept (e.g. *sports*), can be specialized by adding other nouns or adjectives on the left (e.g. *water sports*). To extract NPs, we apply several natural language processing methods [79]: sentence detection, tokenization (i.e. word detection, separating contractions, for example), part-of-speech (POS) tagging and syntactic parsing (i.e. POS-tagged words are put together according to their role, such as Noun phrase or Verb phrase).

As a result of this process, a query q_j is split into several ones $q_j = \{q_{j1}, \ldots, q_{jl}\}$, each one corresponding to a NP. In this manner, user queries with several NPs are treated as several individual queries for anonymization purposes $u'_r = \{q_{11}, \ldots, q_{j1}, \ldots, q_{pl}\}$, considering that each one contributes to the semantic characterization of the user.

Example 1. Let the query log of user u_r be q_1 =*diving in the Mediterranean*, and q_2 =*windsurfing in the Mediterranean*. The first query is split into two individual queries: q_{11} =*diving* and q_{12} =*Mediterranean*; the sec-

ond one is also split into two individual queries: q_{21} =*windsurfing* and q_{22} =*Mediterranean*.

Conceptual mapping In order to map individual queries to their conceptual abstractions in a knowledge base, we look for query-concept label matchings. Since words and NPs could be expressed (both in the queries and in the knowledge base) with different linguistic/morphological variations (e.g. *water sport*, *water sports*, *this water sports*, etc.), we apply additional analyses to detect equivalent formulations of the same concept.

1. Domain-independent words with very general meanings like determinants, prepositions and adverbs, called stop words, are removed from NPs (e.g. *this water sports* = *water sports*).
2. Both queries and concept labels in the knowledge based are stemmed [83] to remove derivational affixes of the same root word (e.g. plurals), identifying equivalent terms (e.g. *water sports* = *water sport*).
3. When a query composed by several words is not found in the knowledge base, we look for simpler query forms by progressively removing adjectives/nouns starting from the one most on the left (e.g. *exciting water sports* → *water sports*). The fact that NPs incorporate qualifiers is quite common in texts but these are scarcely covered in knowledge structures which try to model concepts in a general way. With this strategy, we improve the recall of the conceptual mapping while maintaining, up to a degree, the core semantics of the query.

Knowledge base Semantically-grounded related works either use WordNet [46] or ad-hoc Value Generalization Hierarchies (VGHs) [102] in order to map query terms to concepts. Both present limitations. The former has a low coverage of proper nouns/named entities, which are very common in query logs [93]. In the latter, the construction of ad-hoc VGHs is costly and unfeasible.

Consequently, we used the Open Directory Project (ODP) as the knowledge base in this work (see Section 5.4.1). The advantage is its large size and high recall, with more than one million categories covering up-to-date recently minted terms and named entities. ODP data files can be downloaded in SQL format and categories can be consulted off-line efficiently. Hence, users are mapped to categories, using ODP.

Let $\zeta_u = \{C_{u_1}, \ldots, C_{u_r}, \ldots, C_{u_m}\}$ be the set of users represented by categories, and let $C_{u_r} = \{< c_1, w_1 >, \ldots, < c_j, w_j >, \ldots, < c_n, w_n >\}$ be the categories obtained from the query log of user $u'_r = \{q_{11}, \ldots, q_{j1}, \ldots, q_{pl}\}$, where $< c_i, w_i >$ define a value tuple in which c_i is each distinct category obtained from the set of queries of user u'_r, and w_i is its number of repetitions. Note that different queries could be mapped to the same category.

Finally, we define $T(c_i) = \{c_j \in ODP |\ c_j\ generalizes\ c_i\} \cup \{c_i\}$ as the taxonomic generalizations of category c_i in ODP, including c_i, which summarizes the meaning of c_i.

Example 2. Following *Example 1*, individual queries q_{11} =*diving*, q_{12} =*Mediterranean*, q_{21} =*windsurfing* and q_{22} =*Mediterranean* are mapped to ODP obtaining C_{u_r} ={<*Swimming and Diving*, 1>, <*Mediterranean*, 2>, <*Windsurfing*, 1>} and the taxonomic generalizations: T(*Swimming and Diving*)= {*Sports*, *Water Sports*, *Swimming* and *Diving*}, T(*Mediterranean*)= {*Regional, Europe, Regions, Mediterranean*}, T(*Windsurfing*)= {*Sports, Water Sports, Windsurfing*}.

5.5.1.2 Semantic data partition

Data partition pursues to create clusters of query logs so that each cluster contains, at least, k users. The MDAV microaggregation method has been used to achieve this goal. MDAV relies on two basic functions that depend on the type of data to be processed: a comparison operator that measures the distance between records to add new ones in a cluster, and an averaging function to calculate the centroid used to create clusters.

Due to the characteristics of query logs, the adaptation of MDAV to this kind of data is not trivial. First, contrary to numerical data that can be compared, averaged and transformed by means of mathematical functions, textual queries require operators that consider their semantics [66]. Moreover, as queries define set-valued datasets with variable length and possible value repetitions, the coherent comparison/aggregation of query logs with different cardinalities and value distributions is also challenging.

In this section, we propose semantically-grounded comparison and averaging operators that are able to consider the distributional characteristics of set-valued data. Our goal is to make the MDAV-based data partition to capture both the meaning and the distribution of query logs, so that the information loss resulting from the posterior anonymization can be minimized.

Comparing queries As a result of the query processing and the conceptual mapping, the query log of each user is represented by a set of categories with their corresponding taxonomical generalizations. Hence, we propose a measure that computes the semantic distance between categories, according to their taxonomical trees.

Classical ontology-based methods estimate the distance between terms, according to the number of taxonomical generalizations/specializations that are needed to go from one term to another. This is equivalent to computing the length of the minimum taxonomical path defined between the pair of terms. However, due to their simplicity, they omit much of the taxonomical knowledge explicitly modeled in the knowledge base, achieving a relatively low accuracy [92]. More recent works [8, 92] significantly improve these basic methods by evaluating all the taxonomical ancestors of the compared terms: they measure the distance between terms as a function of the amount of their shared and non-shared taxonomical generalizations.

Considering that the ontological scheme of ODP provides detailed taxonomical structures, our distance measure is designed based on the same principles. Given a pair of categories c_1, c_2 (each one representing a query),

we evaluate their distance $\delta_s(c_1, c_2)$ according to the amount of non-shared taxonomical generalizations in ODP. Moreover, we can also suppose that category pairs that have more generalizations in common are less distant than those sharing a fewer generalizations. Hence, the semantic distance is computed as the ratio between the amount of non-shared categories and the sum of shared and non-shared categories 5.10.

Definition (Semantic distance between categories (δ_s)): The semantic distance between a pair of categories c_1, c_2 is defined as:

$$\delta_s(c_1, c_2) = \frac{|T(c_1) \cup T(c_2)| - |T(c_1) \cap T(c_2)|}{|T(c_1) \cup T(c_2)|} \tag{5.10}$$

Note that, by including the compared categories in T, we are able to distinguish different categories that have all their generalizations in common from two identical categories.

Example 3. Given the pair of categories: c_1 =*Swimming and Diving* and c_2 =*Windsurfing*; we have, as shown in *example 2*, that $T(c_1)$ ={*Sports, Water Sports, Swimming and Diving*} and $T(c_2)$ ={*Sports, Water Sports, Windsurfing*}. Hence, the semantic distance between categories (equation 5.10) is computed as: $\delta_s(c_1, c_2) = \frac{4-2}{4} = 0.5$.

User query log comparison The proposed measure (equation 5.10) has to be extended to semantically compare pairs of user query logs (instead of individual queries). Then, logs can be clustered by means of the MDAV algorithm. However, since query logs of different users may have different lengths and value distributions, it is necessary to integrate distance values between sets of different cardinalities.

The coherent integration is based on the fact that psychological studies show that people pay more attention to the similar features between entities rather than to their differences [40]. Considering categories of user queries as user features, given a category c_i of u_1, we compare it against all categories of u_2,

taking the minimum distance value as the result of this comparison, because it states the highest evidence of similarity between users with respect to feature c_i.

Note that multiple occurrences of the same category may appear in a query log, either because a user repeats a query or because several queries are mapped to the same category (i.e. they represent the same concept, such as *Varicella* and *Chicken Pox*). The distribution of queries/categories in a query log is also an important feature of the user and, together with category semantics, should be considered and preserved by anonymization methods [25]. In order to consider the category distribution, distance measurements for each different category are multiplied by its number of repetitions (Equation 5.11).

$$\Delta_s(c_i, C_{u_2}) = w_i \times \min_{j=1}^{|C_{u_2}|}(\delta_s(c_i, c_j)) \tag{5.11}$$

where $< c_i, w_i > \in C_{u_1}$ and the cardinality $|C_{u_2}|$ is the number of different categories of user u_2.

By repeating the process and adding the distance value between each c_i of u_1 against u_2, we obtain the aggregated distance from u_1 to u_2. Note that this distance may be different when evaluating it from u_2 to u_1. Hence, the final distance between u_1 and u_2 will be the sum between the distances computed from u_1 to u_2 and from u_2 to u_1 (Equation 5.12).

$$D_s(C_{u_1}, C_{u_2}) = \sum_{i=1}^{|C_{u_1}|} w_i \times \min_{j=1}^{|C_{u_2}|}(\delta_s(c_i, c_j)) + \sum_{j=1}^{|C_{u_2}|} w_j \times \min_{i=1}^{|C_{u_1}|}(\delta_s(c_i, c_j)) \tag{5.12}$$

To obtain normalized distance values between query logs (in the $[0, 1]$ interval) so that different user pairs can be compared regardless the cardinality of their logs, we divide it by the number of categories (including repetitions)

of both users.

Definition (Semantic distance between users (D_s)): The semantic distance between users (i.e. the set of categories C_{u_1} and C_{u_2} obtained from the queries of user u_1 and u_2, respectively) is defined in Equation 5.13.

$$D_s(C_{u_1}, C_{u_2}) = \frac{\sum_{i=1}^{|C_{u_1}|} w_i \times \min_{j=1}^{|C_{u_2}|}(\delta_s(c_i, c_j)) + \sum_{j=1}^{|C_{u_2}|} w_j \times \min_{i=1}^{|C_{u_1}|}(\delta_s(c_i, c_j))}{\sum_{i=1}^{|C_{u_1}|} w_i + \sum_{j=1}^{|C_{u_2}|} w_j} \tag{5.13}$$

Example 4. Given the categories of users query logs C_{u_1} ={<*Swimming and Diving*,1>, <*Mediterranean*,1>} and C_{u_2} ={<*Windsurfing*,1>, <*Mediterranean*, 2>}; and considering the taxonomic generalizations T(*Swimming and Diving*)= {*Sports, Water Sports, Swimming and Diving*}, T(*Windsurfing*)={*Sports, Water Sports, Windsurfing*} and T(*Mediterranean*)={*Regional, Europe, Regions, Mediterranean*}; the distance between users is computed as:

$$D_s(C_{u_1}, C_{u_2}) = \frac{(1 \times 0.5) + (1 \times 0) + (1 \times 0.5) + (2 \times 0)}{2 + 3} = \frac{1}{5}$$

Centroid calculus The centroid is understood as the value (or value set in our case) that minimizes the distance against all records in the dataset. When dealing with continuous-scale numerical data, the centroid can be accurately computed by averaging values. However, for textual data, the centroid must necessarily be discretized. In this case, some authors [33] select the centroid of textual/categorical datasets by picking up those which appear more frequently (i.e. the mode). However, this approximation omits the semantics of data.

Given that the distance measure presented above evaluates both the semantic and distributional features of query logs, we use it to compute the dataset

centroid, which is selected as the user query log that minimizes the sum of distances to all other query logs in the dataset.

Definition (Centroid): Given the distance function D_s, the centroid of a set of users represented by categories $\zeta_u = \{C_{u_1}, \ldots, C_{u_r}, \ldots, C_{u_m}\}$ is defined in equation 5.14.

$$centroid(C_{u_1}, \ldots, C_{u_r}, \ldots, C_{u_m}) = argmin_{C_{u_r} \in \zeta_u} \left\{ \sum_{i=1}^{m} D_s(C_{u_r}, C_{u_i}) \right\} \tag{5.14}$$

where *argmin* stands for argument of minimum, i.e. the values of C_{u_r} for which $\sum_{i=1}^{m} D_s(C_{u_r}, C_{u_i})$ attains its minimum value.

As a result of applying the MDAV algorithm, users' query logs will be grouped into $d = \lfloor \frac{m}{k} \rfloor$ clusters of, at least, k users. We define $P = p_1, \ldots, p_d$ as the partition of clusters obtained.

5.5.1.3 Query log anonymization

To fulfill the k-anonymity property, the aggregation step requires to replace all query logs of each cluster by a representative query set. Ideally, this representative should be the most similar (or least distant) to all elements in the cluster, so that the information loss resulting from that replacement can be minimized.

In a general microaggregation scenario, this representative is usually calculated as the centroid of the cluster [30, 22]. However, in the query log anonymization context, selecting a representative as given by equation 5.14 may lead to undesirable consequences. First, the fact that the centroid corresponds to the query log of a concrete user may excessively expose her, especially if an attacker has partial knowledge (e.g. some user queries are known) [46]. Moreover, since the representative can only be picked from the set of original users, it may not accurately represent the average distributional features of the represented group.

To palliate some of these problems, in some works [102] the representative is synthetically built by replacing queries in clusters with concepts that generalize all/some of them, according to a background taxonomy. Hence, anonymized query logs would be composed by sets of concepts rather than real queries. This fact hampers the utility of the anonymized logs in some environments in which queries (instead of their conceptual abstraction) are needed, such as query formulation analysis [6, 109].

In this work, we have also opted for creating synthetic query log representatives that do not correspond to the log of any concrete user to minimize her disclosure risk. However, we maintain individual queries untouched while retaining, as much as possible, the semantic and distributional characteristics of the represented cluster. The creation of synthetic query log representatives is formalized as follows.

Let $C_{p_t} = \bigcup_{C_{u_r} \in p_t} C_{u_r}$ be the set of users (represented by their categories) belonging to a cluster p_t. The construction of the representative for that cluster, that is $\bar{p}_t$, follows these steps:

1. First, we select a sole category that semantically represents the cluster p_t. This category, z_t, is the one that minimizes the sum of distances δ_s to all other categories corresponding to the users in the cluster, that is, C_{p_t}. Hence, z_t corresponds to the centroid category of the cluster p_t. In order to consider both semantic and distributional features in the selection of z_t, semantic distances, δ_s, are weighted by the number of repetitions of each category in the cluster.

   Definition (Centroid Category (z_t)). The centroid category z_t of a cluster p_t is calculated as:

$$z_t = argmin_{c_i \in C_{p_t}} \left\{ \sum_{j=1}^{|C_{p_t}|} w_j \times \delta_s(c_i, c_j) \right\}$$

2. Then, the representative $\bar{p}_t$ of a cluster p_t is built from the subset of categories in the cluster that are most similar to the centroid category z_t. This is done from two perspectives:

   (a) First, the categories of each user are sorted according to its distance with the above-selected centroid category z_t (equation 5.15).

   $$\delta_s(z_t, c_i) = w_i \times \delta_s(z_t, c_i) \tag{5.15}$$

   (b) Second, to construct $\bar{p}_t$ in a way that it also reflects the distribution of categories in the cluster, we compute the contribution *(quota)* that each user u_r in p_t should have in the representative. This *quota* states the number of semantically similar categories from C_{u_r} (with respect to z_t, according to equation 5.15) that a user u_r in p_t will contribute to $\bar{p}_t$. The quota of each user u_r is computed as the ratio between her number of categories (including repetitions) in C_{u_r} and the number of users in the cluster p_t.

   Definition (Quota(u_r)). The *Quota(u_r)* of a user u_r is calculated in Equation 5.16.

   $$Quota(u_r) = \frac{\sum_{i=1}^{|C_{u_r}|} w_i}{|p_t|} \tag{5.16}$$

According to the above criteria, we build the representative $\bar{p}_t$ by picking up *Quota(u_r)* categories (considering their number of repetitions as shown in equation 5.16) from the sorted list of categories (with respect to the centroid category z_t, according to equation 5.16) of each user u_r in the cluster. In this manner, a proportional number of user categories which are the most semantically similar to the centroid category z_t will be incorporated to the representative. The number of categories picked up for each user in the representative will reflect the original distribution of queries in the input dataset, that is, users with many queries will contribute more than those with a few of them.

3. The final step consists in replacing categories in $\bar{p}_t$ by appropriate queries picked up from the original dataset, so that anonymized data can still be useful for query-analysis tasks [6, 105]. Specifically, each category in $\bar{p}_t$ is replaced by a query taken from the whole original dataset corresponding to that category. These queries are randomly picked up from the set of suitable ones. Thanks to this random criterion, we minimize the chance that the cluster representative contains exact subsequences of queries of individual users, a circumstance that may compromise her anonymity [46]. At the same time, since query categories match, we also retain semantics in anonymized data.

As a result of the above process, user query logs of each cluster are replaced with a synthetic query log ($\bar{p}_t$) that contains a representative distribution of those queries found in the original dataset, which are also the most semantically similar to all users logs in the cluster. Hence, the final result is a set of anoymized user query logs $U^A = \{u_{1^A}, \ldots, u_{m^A}\}$.

Notice that since the computational complexity of MDAV is $O(m^2)$ and because the proposed semantic adaptation (using semantic similarity distances to compare query logs) does not affect the overall complexity of the algorithm, our method scales quadratic with respect to the dataset size, making it comparable to other aggregation-based anonymization methods in terms of scalability [38, 75].

5.5.2 Evaluation

In this section, the evaluation of the proposed method is detailed and compared against related works. First, we present the evaluation measures used to quantify the disclosure risk and utility of anonymized query logs (Section 5.5.2.1). Section 5.5.2.2 introduces other query anonymization strategies implemented to compare our results. And we finally present and discuss the results for the evaluated methods in Section 5.5.2.3.

5.5.2.1 Evaluation measures

As stated in the introduction, anonymization methods should maintain a trade-off between two opposite dimensions: data utility, as an inverse function of information loss, and disclosure risk, that is, the chance of an intruder to disclosure the identity of an individual or de-identification. In this section, the measures used to evaluate these dimensions are detailed.

From a general perspective, the utility of anonymized data is retained if the same conclusions can be extracted from the analysis of the original and anonymized datasets. To evaluate up to which point a query anonymization method retains the utility of data (or minimizes the information loss) in an objective and practical way, we rely on data mining techniques. Data mining aims at extracting useful information by characterizing user profiles or preferences. To do so, data mining techniques, and clustering methods in particular, are used to create groups of homogeneous users.

We propose to compute the information loss *(IL)* of an anonymization method by comparing the partitions resulting from original and masked query logs when applying a semantic hierarchical clustering using ODP as the knowledge source, i.e. the higher the distance between cluster sets, the lower the retained utility. Thus, in order to select the most adequate clustering for the given data we use the well-known Calinski-Harabasz index [13]. Differences are quantified according to a well-known distance measure between partitions [22]. Formally, being P_A a partition of the original data, and P_B a partition of the anonymized one, this distance is defined in Equation 5.17.

$$d_{Part}(P_A, P_B) = \frac{2 \times I(P_A \cap P_B) - I(P_A = -I(P_B)}{I(P_A \cap P_B)} \tag{5.17}$$

, where $I(P_A)$ is the average information of P_A which measures the randomness of the distribution of elements over the set of classes of the partition

(similarly for and $I(P_B)$), and $I(PA \cap PB)$ is the mutual average information of the intersection of two partitions [22].

Notice that the distance values obtained are normalized in the [0..1] interval, where 0 indicates identical clusters and 1 maximally different ones. Also, note that the scale is logarithmic, so that it grows according to the amount of differences observed.

Definition (Information Loss (IL)). The percentage of *IL* of masked query logs is quantified as the distance between the obtained partitions for the original (P_A) and anonymized data (P_B), in Equation 5.18.

$$IL = d_{Part}(P_A, P_B) \times 100 \tag{5.18}$$

Another way to evaluate data utility is to analyze the queries' distribution. Information loss can be defined in terms of these differences. Naturally, the larger the difference, the larger the loss. Queries that appear more times are considered to be the most important ones in the log. Thus, we propose to calculate the frequency preservation of the 10 most frequent queries of each log, and the preservation of the 10 most frequent categories of each log as well.

To measure the *Disclosure Risk* (DR) of anonymized query logs, we rely on one of the most common measures: *Record Linkage*(RL) [76]. It quantifies the amount of records (i.e. query logs) that can be correctly matched between the original dataset and the anonymized one. It assumes that a potential attacker would match the least distant original and anonymized records by comparing their queries and picking up the pair or pairs (if several result in the same value) with the highest amount of queries in common. Formally, being u_r an original record (i.e., the original query log of a user), u_{r^A} its anonymized version, and $P_{rl}(u_{r^A})$ the record linkage probability of an anonymized record to be disclosed, the percentage of RL is calculated as follows.

$$RL = \frac{\sum_{r=1}^{m} P_{rl}(u_{r^A})}{m} \times 100$$

$$where\ P_{rl}(u_{r^A}) = \begin{cases} 0 & \text{if } u_r \notin G \\ \frac{1}{|G|} & \text{if } u_r \in G \end{cases}$$

, where G is the set of original records that have been linked to u_{r^A}.

Thus, each u_{r^A} is compared to all records of the original dataset, picking up the pair or pairs with the highest amount of queries in common with u_{r^A}, obtaining the G set of matched records. If u_r is in G, then the probability of record linkage is computed as the probability of finding u_r in G. Otherwise, the record linkage probability is 0. Moreover, we also calculate RL using ODP categories instead of queries, in order to observe the amount of records that can be correctly matched, only knowing the interests (categories) of the users.

In a similar fashion, section 4.5 has proposed the Profile Exposure Level (PEL) measure, which quantifies the uncertainty reduction in the original query logs when an attacker obtains the corresponding anonymized logs. Unlike RL, which is based on query matches, PEL calculates the information that a protected log provides about an original log (mutual information) in terms of Shannon entropy. Hence, we propose to evaluate the privacy of the anonimized logs, using PEL with queries and categories.

5.5.2.2 Comparison

In order to evaluate the contribution of our proposal against related works on query anonymization, we have implemented and tested the methods (introduced in section 3.2) that, as ours, are based on microaggregation, and most recent/elaborated methods based on query removal:

- Korolova et al. [57]: a method based on the removal of scarcest queries (i.e. a priori, the most identifying queries) from the dataset.
- Poblete et al. [82]: another method based on query removal, in this case, relying on a graph-based representation of queries.
- Navarro et al.(Section 5.3): a method based on query microaggregation, proposing several syntactical measures to compare queries. In addition, other query features (like clicked URLs or timestamps) are considered to better differentiate and compare query logs.
- Erola et al.(Section 5.4): a method that compares queries at a conceptual level by using ODP categories. Even though a degree of semantics is considered during query aggregation, prototypes are randomly generated.

In addition to the above methods, we have also implemented a simplified version of our proposal in which *no semantics* are considered at all. In this case, neither query processing nor ODP are used. Queries are treated as simple strings and compared according to their equality/inequality. In the aggregation step, the representative query log is built only considering the distribution of queries. This method aims at evaluating the degree of data utility that can be retained when the query aggregation is solely focused on data distribution.

5.5.2.3 Results

The evaluation has been done using real query logs extracted from the AOL logs. From these, the query logs of 1 000 users have been randomly taken. They contain about 56 000 individual queries, of which around the 61% can be considered complex queries.

The data have been anonymized by means of our method and those introduced in section 5.5.2.2. For methods based on the k-anonymity model

(our proposal, the non-semantic version introduced above, and the ones of Navarro et al. and Erola et al.), k-values between 2 and 7, which resulted in 500 to 142 clusters, have been tested. Note that, due to the high heterogeneity and unbounded nature of query log data (i.e., all query logs define a unique set of queries), even the lowest k-value results in a modification of *all original* query logs after the aggregation process. Hence, most changes are observed for the tested k-value range. Other methods based on query removal (Korolova et al. and Poblete et al.) have been tested varying their corresponding anonymization parameters in reasonable margins (a d value ranging from 1 to 20 for Korolova et. al and a Kp value ranging from 2 to 40 for Poblete et al.). Anonymized datasets for the different approaches and anonymization degrees have been evaluated and compared according to their *information loss* (Figure 5.13) and *record linkage* (Figure 5.16), as detailed in section 5.1. Since k-anonymity values and those of the d and Kp parameters used in Korolova et al. and Poblete et al. approaches are not directly comparable, results are shown in different figures. To measure information loss as a function of the distance between data clusterization results, partitions with 80 clusters (according to the Calinski-Harabasz index) have been created.

After the analysis of information loss figures, we immediately observe a very noticeable difference between the Korolova et al. method and the other ones. The former results in high information loss figures, which state that the conclusions of the analysis of anonymized data will significantly differ from those obtained for the analysis of original query logs. As a result, the utility of masked data is severely hampered. The high degree of query removal performed by Korolova et al. (from a 88% of query removal for d=1 to a 90% for d=20) is, in fact, the least desirable solution from the data analysis perspective, since the semantics provided by the large amount of removed queries are completely lost in the anonymized dataset. Since it is based on the removal of scarcest queries of the dataset and, considering that most queries appeared once, even for the more relaxed anonymization parameter (d), it resulted in the complete removal of all queries for a considerable

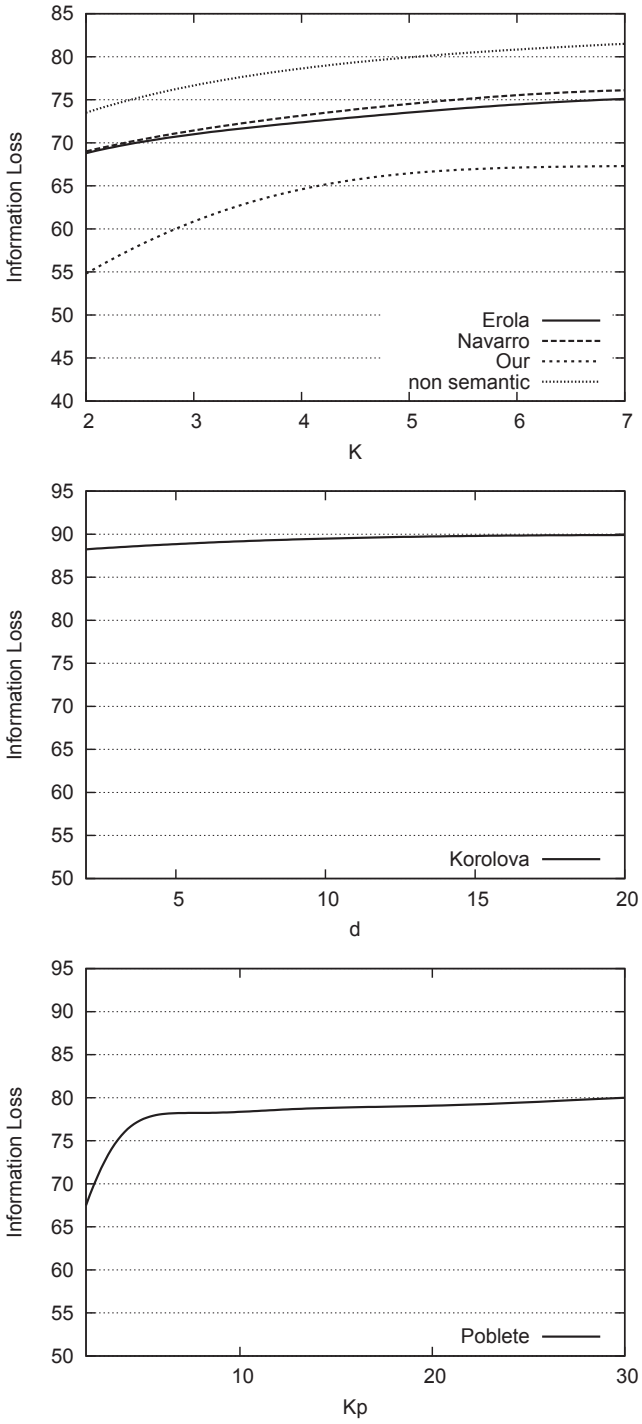


FIGURE 5.13: Information Loss (IL).

amount of users (i.e., for d=1, about a 46% of users had all their queries removed, whilst this figure increases to 65% for d=5).

In comparison, the method by Poblete et al. results in lower information loss figures, since its removal strategy is more focused on queries producing too little results, when queried in a search engine (parameter Kp), than on their distribution in the dataset. As a result, it removes a significantly lower amount of queries (i.e., for Kp=30 only a 10% of users had all their queries removed).

In general, methods based on query log aggregation better retain the utility of query logs. Moreover, since all of them are based on the k-anonymity model, their results can be compared under the same conditions. Among them, the worst was the one based solely on boolean comparisons between queries (i.e., the non-semantic implementation of our proposal). Since queries can only be evaluated as identical or different, and, since most queries in the dataset are unique, few evidences of similarity that can guide the partition and aggregation processes can be gathered. In this case, only query distribution (i.e. their number of repetitions) is considered. Better results are obtained for the method by Navarro et al. Even though it is also based on terminological comparisons, it enables a more fuzzy evaluation of query similarity thanks to the use of the Edit distance to compare Strings. This, combined with the evaluation of other query features such as the timestamps or clicked URLs, enables a finer grained comparison between queries and, hence, a more accurate aggregation.

However, these approaches do not consider query semantics in an explicit way. Even though terminological resemblance is an evidence of semantic similarity, it poorly captures and evaluates the meaning of queries. The approach by Erola et al. exploits ODP to retrieve categories to which queries refer, and evaluates the number of terminologically identical categories between query logs as a measure of similarity. Since categories define a more constrained set of modalities than query logs, the chance of discovering terminological matchings increases and so do the evidences of similarity.

Moreover, since categories are conceptualizations of textual queries, this approach enables a semantically-coherent partition of query logs. However, no semantic evidences are used during the aggregation process, which is solely focused on the preserving of the distribution of queries.

Finally, our method provides the lowest information loss for all k-values. Also note that since the dPart(PA,PB) (see Equation 5.17) function used to compute information loss has a logarithmic scale, the absolute differences in partitions for high information loss values are greater than for lower ones. The obtained improvement is the result of considering both the semantics of queries and their distribution in all stages of the query anonymization process (i.e. comparison between queries, centroid selection, cluster construction and data aggregation), relying on a semantically-coherent distance measure and ODP categories. Also note that the query processing stage also contributes to interpret semantics of complex queries (i.e. those with several word or noun phrases) more coherently, in comparison with methods that treat queries word by word [37, 75].

Figure 5.14 shows the frequency preservation of the queries. As before, we immediately observe that Korolova et al. obtains the worst results due to the excessive query removal. The aggregation based method by Erola et al., the non-semantic method and our method achieve close results. On the contrary, Navarro et al. obtains nearly 5% of better results, since it aggregates the logs taking special care to maintain the frequencies of the most important queries. However, greater results are obtained by Poblete et al., as it slightly modifies the query logs.

If we compare this figure with Figure 5.15, which shows the frequency preservation of the categories, we can observe that Korolova et al. and Poblete et al. are still the best and the worst, respectively. Within the aggregation based methods, we can observe that the non-semantic method achieves the worst results. The methods by Navarro et al. and Erola et al. better maintain the frequencies, as a result of considering some similarity resemblance during the clustering. Finally, our method attains the best results, as an

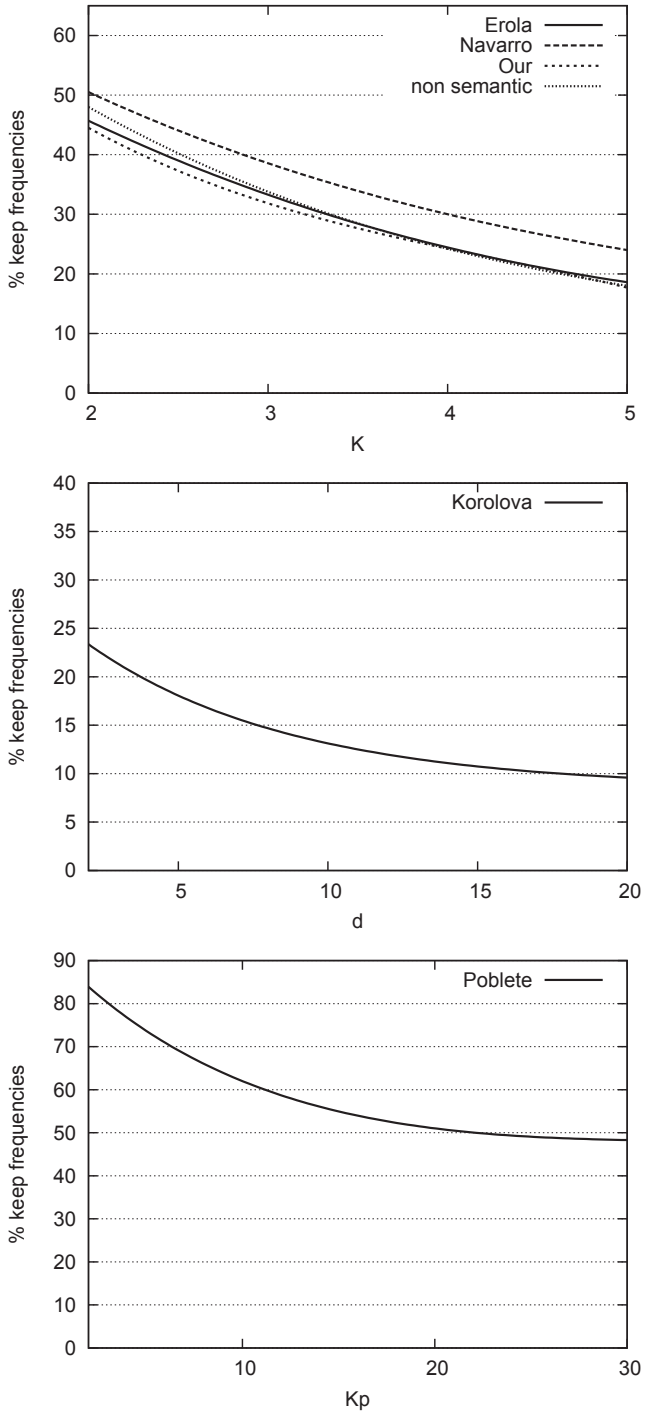


FIGURE 5.14: Frequency preservation of most common queries.

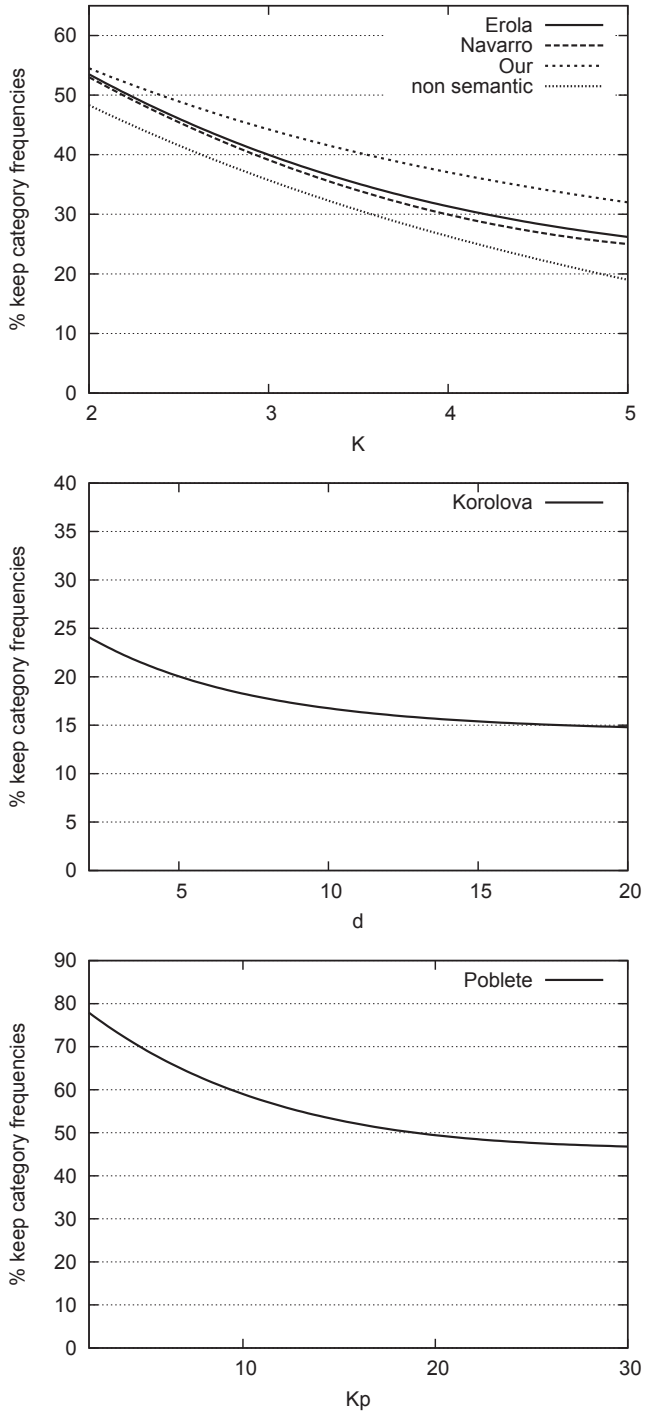


FIGURE 5.15: Frequency preservation of most common categories.

effect of considering query semantics as much as their distributions during anonymization.

Disclosure risk evaluates the chance by an intruder to match original and anonymized records, and directly depends on the degree of distortion introduced in the anonymized dataset. Since disclosure risk is based on the degree of overlapping between query logs, the more different they are with respect to original ones, the more private the results will be. After analyzing disclosure risk figures, we observe that the method by Korolova et al. results in the lowest figures (between 25% of matched records for d=1 to almost 0% for d=20). As stated above, this is caused by the great amount of user logs for which all queries have been removed. Obviously, if no queries are available, no linkage is possible, but it also becomes useless for statistical and data analysis due to the high information loss.

The method by Poblete et al. results in higher figures (around 63% of linkages for Kp>5) (Figure 5.17). This method removes a significantly lower amount of query logs (i.e., for Kp=30 only a 10% of query logs have been completely removed). These results, in combination with the fact that non-removed queries appear *as it is* in the masked dataset (i.e. no transformation, swapping or replacement is done), increase the chance of discovering correct linkages with original logs.

Within the aggregation based methods, the one solely based on query distribution (i.e., the non-semantic version of our proposal) results in the highest amount of linkages. In this case, the fact that queries can only be evaluated in a Boolean fashion, and the fact that query logs are aggregated according to their distributional properties, limit the amount of distortion introduced in the anonymized data and increase the chance of linkage.

Our method is able to minimize the information loss of anonymized data and at the same time it is also able to maintain the amount of linkages as low as the methods based on query aggregation do. In this case, even though semantics of anonymized data are better retained, the fact that the aggregation is made by randomly rearranging queries of different users (while

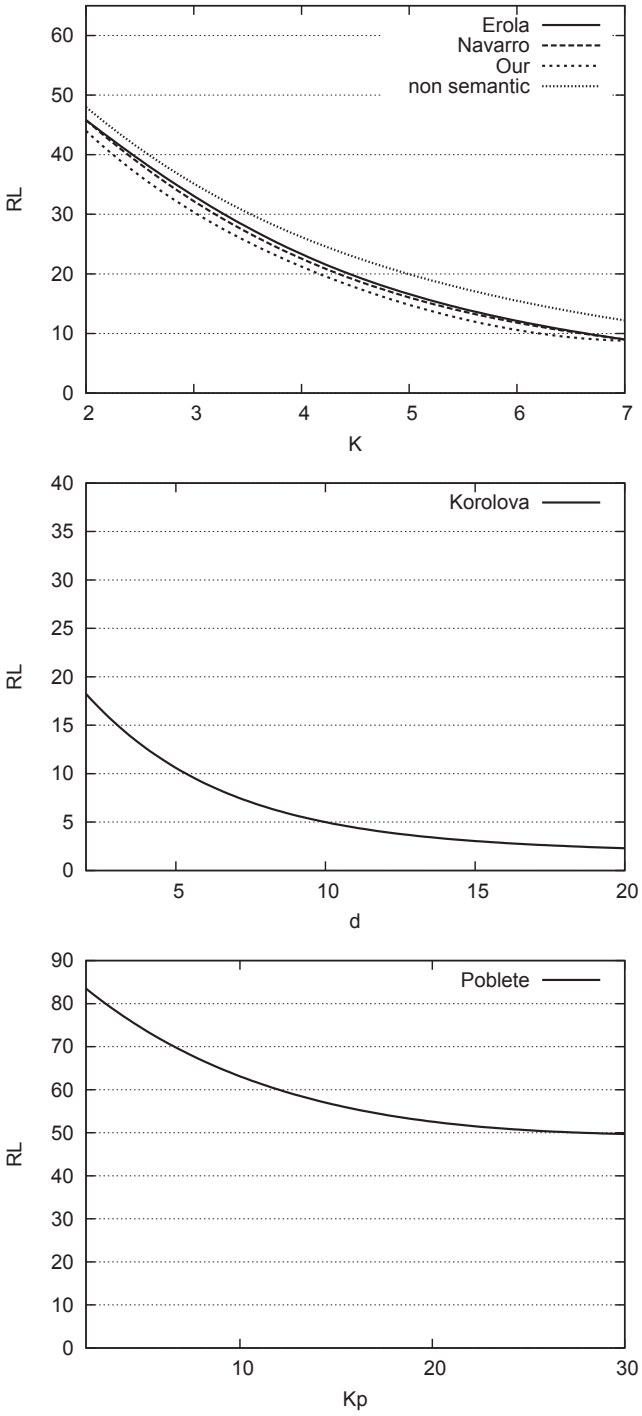


FIGURE 5.16: Record Linkage (RL).

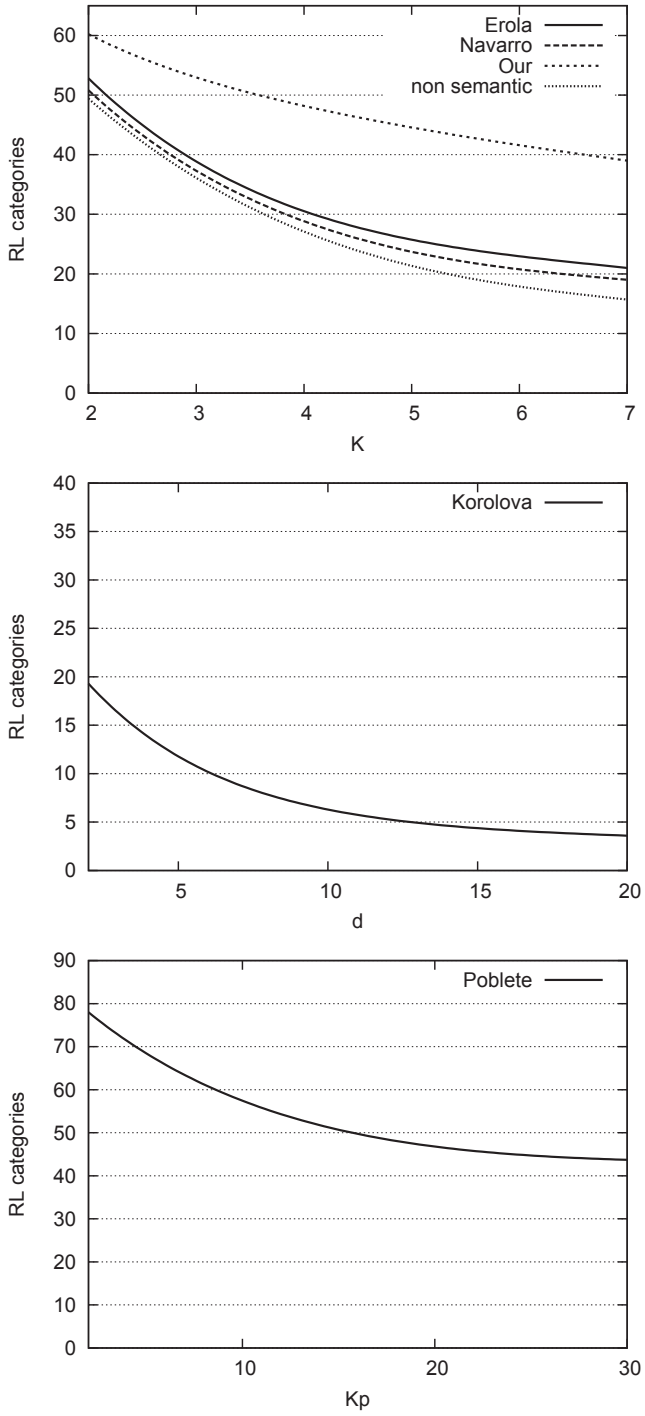


FIGURE 5.17: Record Linkage (RL) with categories.

maintaining their distribution and semantics) contributes to maintain record linkage at levels comparable to those of related works. Hence, queries of the masked log can be different from those in the original one, although they will cover similar topics.

Figure 5.17 demonstrates the above statement. The RL of categories that aggregation methods obtain is close, except in the case of our method, which achieves significantly better results. As it happens with the frequencies, this is the effect of considering semantics and distributions during anonymization. The methods by Poblete et al. and Korolova et al. achieve similar observations to RL of queries.

The amount of information that anonymized query logs exposes is shown in Figure 5.18. Korolova et al. method obtains the best results (lowest exposure), yet it is at the expense of a vast amount of query suppressions. On the other hand, the method by Poblete et al. exposes a great deal of information while its information loss is low. The aggregation based methods achieve close results with the theoretical PEL, which is inversely proportional to k.

Finally, Figure 5.19 shows PEL results using categories. It can be noticed that results are lower than PEL results using categories. As we stated before, categories define a more constrained set of modalities than queries, thus the probability to find similarities increases. Methods based on query aggregation obtain a quite comparable amount of linkages, which decrease almost linearly as the k-anonymity level increases. From these, the non-semantic method achieves slightly worst results, due to the simple Boolean comparison of the queries. The methods by Poblete et al. and Korolova et al. achieve similar observations to PEL with queries. The former seems to achieve lineal results, but this is caused by high similarity values, which range from 95% to 93%.

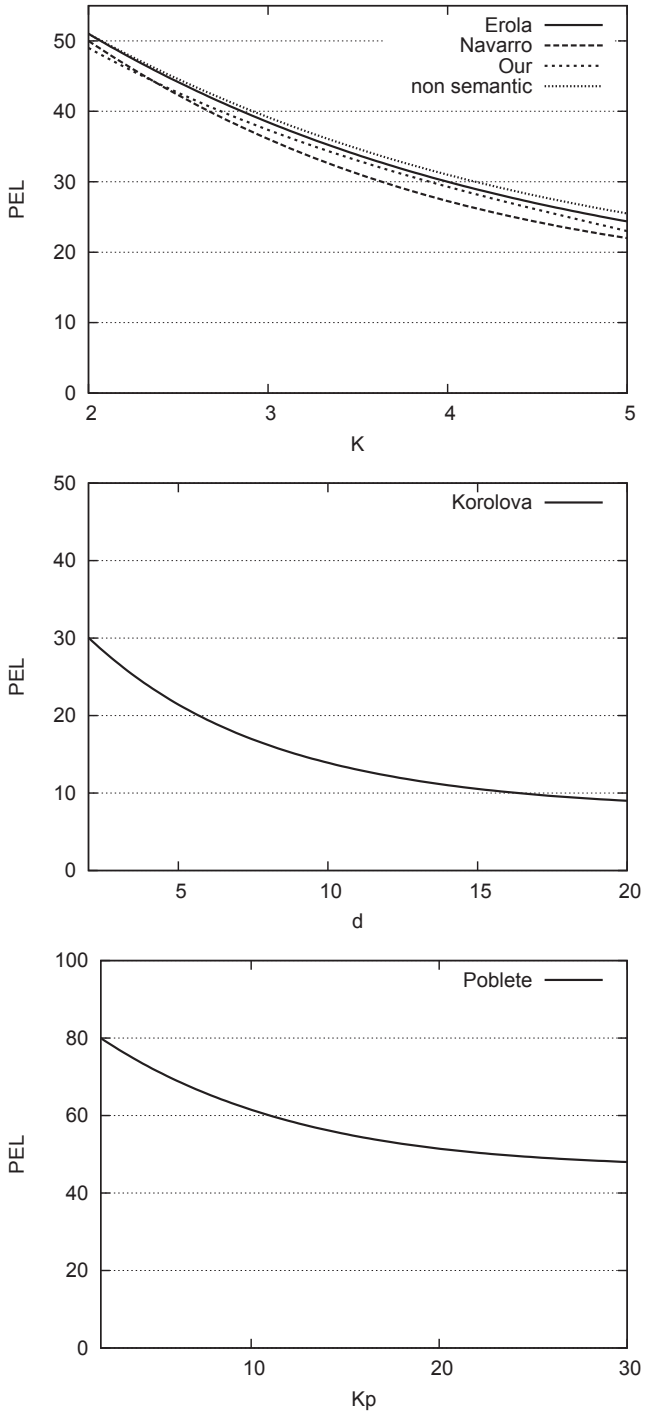


FIGURE 5.18: Profile Exposure Level (PEL).

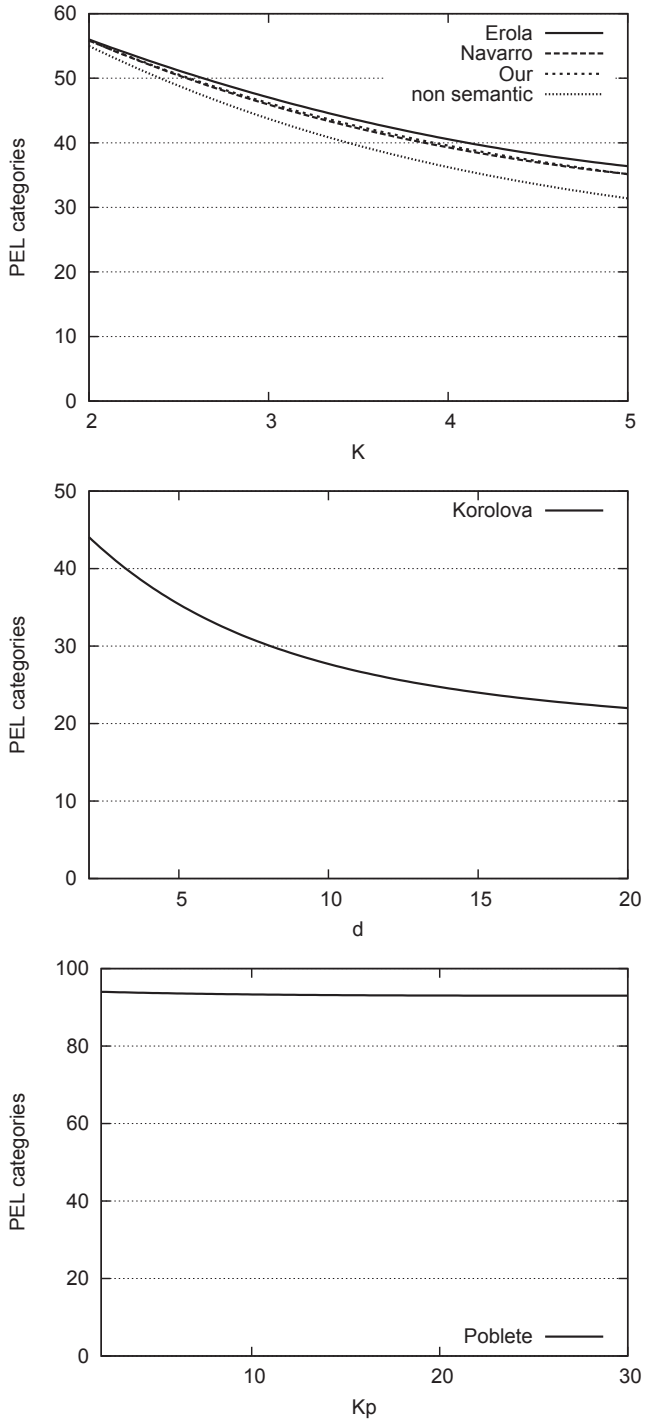


FIGURE 5.19: Profile Exposure Level (PEL) with categories.

5.5.3 Conclusions

We presented a novel query log microaggregation technique that semantically interpret queries by extracting their conceptualizations from the ODP structured set of categories. This enables to aggregate query logs semantically by means of an adaptation of the MDAV algorithm. Suitable semantic operators to compare and average query logs have been proposed for that purpose. Finally, a method to generate synthetic query logs composed by real queries, which replace original ones, is also proposed. This method preserves the semantics and distribution of original queries, while keeping disclosure risk at a reasonable level.

The evaluation carried out with a set of real query logs extracted from the AOL dataset sustains the suitability of our method. Compared to related works based on query removal or non-semantic query aggregation, our proposal better retained data utility while maintaining a desirable level of disclosure risk.

5.5.3.1 Publications

- M. Batet, A. Erola, D. Sánchez, J. Castellà-Roca, *Utility preserving query log anonymization via semantic microaggregation*, Information Sciences, Vol. 242, pp. 49-63, Sep 2013, ISSN: 0020-0255.

Chapter 6

Conclusions

6.1 Contributions

The present thesis has dealt with privacy disclosure risks in web search. While most of the research on privacy protection of queries focuses on minimizing disclosure risks, our aim has been to ensure the privacy of the users while minimizing information loss. To that end, we have addressed the problem from two different sides: the client-side and the server-side.

In the former, we have presented a collaborative method where users provide privacy in a social network. Basically, a user submits queries on behalf of her neighbors and conversely. Thus, users' profiles become obfuscated. The evaluation has shown that users achieve some privacy degree because they expose a small number of their real queries, even with the presence of selfish users. Moreover, the acceptable query delay makes the system usable in practice.

In the latter, three techniques to anonymize query logs by means of microaggregation have been presented. Our proposals provide a high degree of privacy (k-anonymity) while maintaining some data utility. 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 re-identification is directly related to the size of the parameter k used in the microaggregation. Moreover, our proposals have the advantage that can be adapted to offer higher privacy guarantees, as l-diversity and t-closeness.

Therefore, in order to find a good trade-off between privacy and utility, information loss should be minimized. To that end, partition and aggregation operators to deal with query logs should be defined. Our first proposal uses an aggregation of several distance functions and the MDAV algorithm to cluster the logs. Then, the aggregation operator takes special care to maintain frequencies of the queries. However, while preserving some properties of the original queries, they do not take the meaning of the queries into account.

WSE use query logs that contain the users' interests, in order to create detailed profiles of their users. Hence, the utility of the data is related to ability to construct these profiles. By pursuing this idea, we introduce the concept of semantic microaggregation in our second proposal. To that end, the partition operation takes the semantics of the query terms into account. Semantics are extracted by classifying the query terms in a structured knowledge base, which provides a metric space to define the distance between two logs. Experimental results show that the method outperforms the utility of the previous proposals.

However, while the interpretation of the real meaning (user's interest) of simple queries (queries composed of one or two terms) is possible in the knowledge base, the interpretation of complex queries (queries composed of several words) requires some advanced techniques. The interpretation of complex queries from the interpretation of their terms perturbs the query's semantics, as the relations between their terms are omitted. In order to solve this problem, we have proposed our third method, which maps users' queries to their formal semantics using some linguistic analysis and the same knowledge base. The method has been compared with our previous proposals, a

random proposal and Korolova et al. [57] and Poblete et al. [82] proposals. The evaluation using data mining methods shows that it clearly maintains more data utility than other proposals, while the record linkage is as low as the obtained by the other methods at minimum.

6.2 Future work

Here, we sketch some open problems that can be addressed in the future.

- In the client-side method, the quality of the service is related to the reliability of the interests of the user. We will consider using a specialized social network in order to get more homogeneous shared interests between users. This enhancement should improve the quality of the service. Nevertheless, there are some privacy issues that must be investigated.
- The vast amount of queries WSEs receive every day, should be taken into consideration in order to apply the presented methods. While the methods offer good privacy and data utility, their performance dealing with large datasets has not been evaluated. The way to deal with vast volumes of queries should be studied.
- The interpretation of queries is conditional on the knowledge base. Different knowledge bases (or even when the same knowledge base has been updated) can have different query' interpretations, thus providing different distances between two queries. Therefore, the obtained results by using our methods can be altered if we change the knowledge base. Different ways to compare search logs that do not depend on external sources should be investigated.

6.3 Publications

The publications accepted during this distertation are:

- A. Erola, J. Castellà-Roca, A. Viejo and J.M. Mateo-Sanz, *Exploting Social Networks to Provide Privacy in Personalized Web Search*, Journal of Systems and Software, Vol. 84, no. 10, pp. 1734-17445, Oct 2011, ISSN: 0164-1212

- G. Navarro-Arribas, V. Torra, A. Erola and J. Castellà-Roca, *User k-anonymity for privacy preserving data mining of query logs*, Information Processing and Management, Vol. 48, no. 3, pp. 476-487, May 2012, ISSN: 0306-4573.

- A. Erola, J. Castellà-Roca, G. Navarro-Arribas and V. Torra, *Semantic microaggregation for the anonymization of query logs using the open directory project*, SORT-Statistics and Operations Research Transactions, Vol. 0, Special issue, pp. 41-58, Sep 2011, ISSN: 1696-2281

- M. Batet, A. Erola, D. Sánchez, J. Castellà-Roca, *Utility preserving query log anonymization via semantic microaggregation*, Information Sciences, Vol. 242, pp. 49-63, Sep 2013, ISSN: 0020-0255.

- A. Erola, J. Domingo-Ferrer and J. Castellà-Roca, *Ofuscación del perfil del usuario de un motor de búsqueda mediante una red social y protocolos criptográficos*, XI Reunión Espaola sobre Criptología-RECSI 2010, Tarragona, Sep 2010.

- G. Navarro-Arribas, V. Torra, A. Erola, J. Castellà-Roca, *Microagregación para la k-anonimidad en logs de buscadores Web*, Reunión Espaola sobre Criptología y Seguridad de la Información-RECSI 2010, Tarragona, Spain, Sep 2010.

- A. Erola, J. Castellà-Roca, G. Navarro-Arribas and V. Torra, *Semantic Microaggregation for the Anonymization of Query Logs*, Lecture Notes

in Computer Science, Vol. 6344 (Privacy in Statistical Databases-PSD 2010), pp. 127-137, Sep 2010, ISSN: 0302-9743.

- A. Erola and J. Castellà-Roca, *Anonimización de registros de búsqueda mediante la semántica de las consultas*, RECSI 2012, Donostia, Spain, In Actas de la XII Reunión Espaola sobre Criptología y Seguridad de la Información, pp. 279-284, ISBN: 978-84-615-9933, Sep 2012.

Bibliography

[1] E. Adar. User 4xxxxx9: Anonymizing query logs. In *Query Logs workshop*, 2007.

[2] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 19–26, New York, NY, USA, 2006. ACM.

[3] R. Anderson. The eternity service. In *Proc. PRAGO-. CRYPT 96*, pages 242–252, 1996.

[4] Inc. AOL. Aol keyword searches. http://dontdelete.com/default.asp, 2006.

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

[6] J. Bar-Ilan. Access to Query Logs An Academic Researcher's Point of View. In Einat Amitay, G. Craig Murray, and Jaime Teevan, editors, *Query Log Analysis: Social And Technological Challenges. A workshop at the 16th International World Wide Web Conference (WWW 2007)*, May 2007.

[7] M. Barbaro and T. Zeller. A face is exposed for aol searcher no. 4417749. The New York Times, August 2006.

[8] M. Batet, D. Sánchez, and A. Valls. An ontology-based measure to compute semantic similarity in biomedicine. *J. of Biomedical Informatics*, 44(1):118–125, February 2011.

[9] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 407–416, 2000.

[10] S.M. Beitzel, E.C. Jensen, A. Chowdhury, O. Frieder, and D. Grossman. 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, 2007.

[11] S.M. Beitzel, E.C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder. Hourly analysis of a very large topically categorized web query log. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 321–328, New York, NY, USA, 2004. ACM.

[12] D.J. Brenes and D. Gayo-Avello. Stratified analysis of aol query log. *Information Sciences*, 179(12):1844 – 1858, 2009.

[13] T. Caliski and J Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974.

[14] J. Castellà-Roca, A. Viejo, and J. Herrera-Joancomartí. Preserving user's privacy in web search engines. *Computer Communications*, 32:1541–1551, 2009.

[15] Center for Democracy and Technology. Ask.com puts you in control of your search privacy with the launch of *AskEraser*. http://www.prnewswire.com, 2007.

[16] Center for Democracy and Technology. Search privacy practices: A work in progress. http://www.cdt.org/privacy/20070808searchprivacy.pdf, 2007.

[17] D.L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, 1981.

[18] D.R. Choffnes and F.E. Bustamante. Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems. In *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 363–374, 2008.

[19] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *FOCS '95: Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, 1995.

[20] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Journal of the ACM*, volume 45, pages 965–981, 1998.

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

[22] R.L. De Mántaras. A distance-based attribute selection measure for decision tree induction. *Machine Learning*, 6(1):81–92, 1991.

[23] D. Defays and P. Nanopoulos. Panels of enterprises and confidentiality: the small aggregates method. In *Proc. of 92 Symposium on Design and Analysis of Longitudinal Surveys, Statistics Canada*, pages 195–204, 1993.

[24] J. Domingo-Ferrer. A public-key protocol for social networks with private relationships. In *MDAI '07: Proceedings of the 4th international conference on Modeling Decisions for Artificial Intelligence*, pages 373–379, 2007.

[25] J. Domingo-Ferrer. A survey of inference control methods for privacy-preserving data mining. In C. Aggarwal and P. Yu, editors, *Privacy-Preserving Data Mining*, volume 34 of *Advances in Database Systems*, pages 53–80. Springer US, 2008.

[26] J. Domingo-Ferrer. Coprivacy: towards a theory of sustainable privacy. *PSD2010, LNCS*, 6344:258–268, 2010.

[27] J. Domingo-Ferrer and M. Bras-Amorós. Peer-to-peer private information retrieval. In *PSD '08: Proceedings of the UNESCO Chair in data privacy international conference on Privacy in Statistical Databases*, pages 315–323, 2008.

[28] J. Domingo-Ferrer, M. Bras-Amorós, Q. Wu, and J. Manjón. User-private information retrieval based on a peer-to-peer community. *Data Knowl. Eng.*, 68(11):1237–1252, November 2009.

[29] J. Domingo-Ferrer, A. Martínez-Ballesté, JM. Mateo-Sanz, and F. Sebé. Efficient multivariate data-oriented microaggregation. *The VLDB Journal*, 15(4):355–369, November 2006.

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

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

[32] J. Domingo-Ferrer, A. Solanas, and J. Castellà-Roca. h(k)-private information retrieval from privacy-uncooperative queryable databases. *Journal of Online Information Review*, 33:720–744, 2009.

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

[34] EFF. AOL's massive data leak. Electronic Frontier Foundation, http://w2.eff.org/Privacy/AOL/, 2009.

[35] Y. Elovici, B. Shapira, and A. Maschiach. A new privacy model for hiding group interests while accessing the web. In *WPES '02: Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*, pages 63–70, 2002.

[36] Y. Elovici, B. Shapira, and A. Maschiach. A new privacy model for web surfing. In *NGITS '02: Proceedings of the 5th International Workshop on Next Generation Information Technologies and Systems*, pages 45–57, 2002.

[37] A. Erola, J. Castellà-Roca, G. Navarro-Arribas, and V. Torra. Semantic microaggregation for the anonymization of query logs using the open directory project. *SORT-Statistics and Operations Research Transactions*, 35(Special issue):25–40, Sep 2011.

[38] A. Erola, J. Castellà-Roca, G. Navarro-Arribas, and Vicenç Torra. Semantic microaggregation for the anonymization of query logs. In *Proc. Privacy in Statistical Databases (PSD 2010)*, volume 6344 of *LNCS*, pages 127–137, June 2010.

[39] A. Erola, J. Castellà-Roca, A. Viejo, and JM. Mateo-Sanz. Exploiting Social Networks to Provide Privacy in Personalized Web Search. *Journal of Systems and Software*, May 2011.

[40] R. L. Goldstone. Similarity. In *The MIT encyclopedia of the cognitive sciences MIT Press*. 1999.

[41] Google. Google personalized search. http://www.google.com/psearch, 2009.

[42] Google. Google official history. http://www.comscore.com/, 2012.

[43] Google. Google's income statement information. http://investor.google.com, 2012.

[44] N. Guarino. Formal ontology and information systems. pages 3–15. IOS Press, 1998.

[45] S. Hansell. Increasingly, internet's data trail leads to court. The New York Times, February 2006.

[46] Y. He and J. Naughton. Anonymization of set-valued data via top-down, local generalization. *Proceedings of the VLDB Endowment*, 2(1):934–945, 2009.

[47] Y. Hong, X. He, J. Vaidya, N. Adam, and V. Atluri. Effective anonymization of query logs. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1465–1468, 2009.

[48] D.C. Howe and H. Nissenbaum. TrackMeNot: Resisting surveillance in web search. In Ian Kerr, Valerie Steeves, and Carole Lucock, editors, *Lessons from the Identity Trail: Anonymity, Privacy, and Identity in a Networked Society*, chapter 23, pages 417–436. Oxford University Press, Oxford, UK, 2009.

[49] Inc. iProspect.com. iprospect blended search results study. http://www.iProspect.com, 2009.

[50] ISO. Iso/iec 9126-1, software engineering - product quality - part 1: Quality model, 2001.

[51] B.J. Jansen and A. Spink. An analysis of web searching by european AlltheWeb.com users. *Information Processing & Management*, 41(2):361–381, 2005.

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

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

[54] X. Jin, N. Zhang, and G. Das. Asap: Eliminating algorithm-based disclosure in privacy-preserving data publishing. *Inf. Syst.*, 36(5):859–880, July 2011.

[55] R. Jones, R. Kumar, B. Pang, and A. Tomkins. "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*, CIKM '07, pages 909–914, New York, NY, USA, 2007. ACM.

[56] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 387–396, New York, NY, USA, 2006. ACM.

[57] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 171–180, 2009.

[58] T. Kuflik, B. Shapira, Y. Elovici, and A. Maschiach. Privacy preservation improvement by learning optimal profile generation rate. In *User Modeling*, pages 168–177, 2003.

[59] R. Kumar, J. Novak, B. Pang, and A. Tomkins. On anonymizing query logs via token-based hashing. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 629–638, New York, NY, USA, 2007. ACM.

[60] E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *In Proc. of the 38th Annu. IEEE Symp. on Foundations of Computer Science*, pages 364–373, 1997.

[61] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, pages 106–115, 2007.

[62] D. Liben-Nowell. *An algorithmic approach to social networks*. PhD thesis, MIT Computer Science and Artificial Intelligence Laboratory, 2005.

[63] W. Lixia and H. Jianmin. Utility evaluation of k-anonymous data by microaggregation. In *International Conference on Communication System, Networks and Applications, 2009 ICCSNA*, volume 4, pages 381–384, 2009.

[64] M. Arrington. Aol proudly releases massive amounts of private data. TechCrunch, 2007.

[65] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

[66] S. Martínez, D. Sánchez, and A. Valls. Semantic adaptive microaggregation of categorical microdata. *Computers & Security*, 31(5):653 – 672, 2012.

[67] S. Martínez, D. Sánchez, A. Valls, and M. Batet. Privacy protection of textual attributes through a semantic-based masking method. *Information Fusion*, 13(4):304 – 314, 2012. ¡ce:title¿Information Fusion in the Context of Data Privacy¡/ce:title¿.

[68] S. Martínez, A. Valls, and D. Sánchez. Semantically-grounded construction of centroids for datasets with textual attributes. *Knowledge-Based Systems*, 35(0):160 – 172, 2012.

[69] N. Matatov, L. Rokach, and O. Maimon. Privacy-preserving data mining: A feature set partitioning approach. *Information Sciences*, 180(14):2696 – 2720, 2010. ¡ce:title¿Including Special Section on Hybrid Intelligent Algorithms and Applications¡/ce:title¿.

[70] G. Miller. WordNet - about us. WordNet. Princeton University, http://wordnet.princeton.edu, 2009.

[71] E. Mills. AOL sued over web search data release. CNET News, http://news.cnet.com/8301-10784_3-6119218-7.html, September 2006.

[72] A. Mislove, K.P. Gummadi, and P. Druschel. Exploiting social networks for internet search. In *Proceedings of the 5th Workshop on Hot Topics in Networks (HotNets'06)*, 2006.

[73] G. Navarro-Arribas and V. Torra. Tree-based microaggregation for the anonymization of search logs. In *WI-IAT '09: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 155–158, 2009.

[74] G. Navarro-Arribas and V. Torra. Privacy-preserving data-mining through micro-aggregation for web-based e-commerce. *Internet Research*, 20(3):366–384, 2010.

[75] G. Navarro-Arribas, V. Torra, A. Erola, and J. Castellà-Roca. User k-anonymity for privacy preserving data mining of query logs. *Information Processing & Management*, 48(3):476 – 487, 2012.

[76] J. Nin, J. Herranz, and V. Torra. On the disclosure risk of multivariate microaggregation. *Data Knowl. Eng.*, 67(3):399–412, December 2008.

[77] ODP. Open directory project. http://www.dmoz.org, 2010.

[78] A. Oganian and J. Domingo-ferrer. On the complexity of optimal microaggregation for statistical disclosure control. *Statistical Journal of the United Nations Economic Comission for Europe*, 18:345–354, 2001.

[79] OpenNLP. Opennlp projects. http://opennlp.sourceforge.net/projects.html, 2012.

[80] G. Pass, A. Chowdhury, and C. Torgeson. A picture of search. In *Proceedings of the 1st international conference on Scalable information systems*, page 1, 2006.

[81] J. Pitkow, H. Schütze, T. Cass, R. Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. Breuel. Personalized search. *Commun. ACM*, 45(9):50–55, 2002.

[82] B. Poblete, M. Spiliopoulou, and R. Baeza-Yates. Website privacy preservation for query log publishing. In *First International Workshop on Privacy, Security, and Trust in KDD (PinKDD 2007)*, pages 80–96, 2008.

[83] M. F. Porter. Readings in information retrieval. chapter An algorithm for suffix stripping, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[84] Privoxy. http://www.privoxy.org, 2009.

[85] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *Proc. of the 15th international conference on World Wide Web*, pages 727–736, 2006.

[86] M.K. Reiter and A.D. Rubin. Crowds: anonymity for web transactions. *ACM Trans. Inf. Syst. Secur.*, 1(1):66–92, 1998.

[87] M.K. Reiter and A.D. Rubin. Anonymous web transactions with crowds. *Commun. ACM*, 42(2):32–48, 1999.

[88] S. Romanosky. Foxtor. http://www.romanosky.net, 2009.

[89] S. Hansell. Increasingly, internet's data trail leads to court. The New York Times, 2006.

[90] F. Saint-Jean, A. Johnson, D. Boneh, and J. Feigenbaum. Private web search. In *WPES '07: Proceedings of the 2007 ACM workshop on Privacy in electronic society*, pages 84–90, 2007.

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

[92] D. Sánchez, M. Batet, D. Isern, and A. Valls. Ontology-based semantic similarity: A new feature-based approach. *Expert Syst. Appl.*, 39(9):7718–7728, July 2012.

[93] D. Sánchez, J. Castellí-Roca, and A. Viejo. Knowledge-based scheme to create privacy-preserving but semantically-related queries for web search engines. *Inf. Sci.*, 218:17–30, January 2013.

[94] X. Shen, B. Tan, and C. Zhai. Privacy protection in personalized search. *SIGIR Forum*, 41(1):4–17, 2007.

[95] C. Soghoian. The problem of anonymous vanity searches. *I/S: A Journal of Law and Policy for the Information Society*, 3(2), 2007.

[96] M. Speretta and S. Gauch. Personalized search based on user search histories. In *Proc. of International Conference of Knowledge Management -CIKM'04*, 2004.

[97] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *Proc. of the 13th international conference on World Wide Web*, pages 675–684, 2004.

[98] N. Summers. Walking the cyberbeat. Newsweek. http://www.newsweek.com/id/195621, May 2009.

[99] L. Sweeney. k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness, and knowledge-based systems*, 10(5):557–570, 2002.

[100] B. Tancer. Click: What millions of people are doing online and why it matters. Hyperion, 2008.

[101] J. Teevan, S.T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456, 2005.

[102] M. Terrovitis, N. Mamoulis, and P. Kalnis. Privacy-preserving anonymization of set-valued data. *Proc. VLDB Endow.*, 1(1):115–125, August 2008.

[103] Inc. The Tor Project. Tor project. http://www.torproject.org, 2009.

[104] V. Torra. Microaggregation for categorical variables: A median based approach. In *Proc. Privacy in Statistical Databases (PSD 2004)*, volume 3050 of *LNCS*, pages 162–174, June 2004.

[105] V. Torra. Towards privacy-preserving query log publishing. volume Proceedings of the Query Log Analysis: Social and Technological Challenges Workshop at the 16th World Wide Web Conference. 2007.

[106] V. Torra. Constrained microaggregation: Adding constraints for data editing. *Transactions on Data Privacy*, 1(2):86–104, 2008.

[107] V. Torra. Towards knowledge intensive data privacy. In Joaquin Garcia-Alfaro, Guillermo Navarro-Arribas, Ana Cavalli, and Jean Leneutre, editors, *Data Privacy Management and Autonomous Spontaneous Security*, volume 6514 of *Lecture Notes in Computer Science*, pages 1–7. Springer Berlin Heidelberg, 2011.

[108] A. Viejo and J. Castellà-Roca. Using social networks to distort users' profiles generated by web search engines. *Computer Networks*, 54:1343–1357, 2010.

[109] L. Xiong and E. Agichtein. Towards privacy-preserving query log publishing. In *Query Log Analysis: Social and Technological Challenges, Workshop in 16 International World Wide Web Conference*, 2007.

[110] Y. Xu, K. Wang, B. Zhang, and Z. Chen. Privacy-enhancing personalized web search. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 591–600, 2007.

[111] K. Zetter. Yahoo issues takedown notice for spying price list. Wired. http://www.wired.com/threatlevel/2009/12/yahoo-spy-prices/#more-11725, December 2009.