Privacy Friendly Search

Abhishek Bandivadekar
Rob Craaijo
Rico te Wechel

Radboud University

How vital is web search

- **Google is estimated to process 99,000 search queries per second**
- **That is 8.5 billion searches per day**
- **2 Trillion global searched per year**
- **The sheer volume of data processed and collected is huge and that poses a question:**

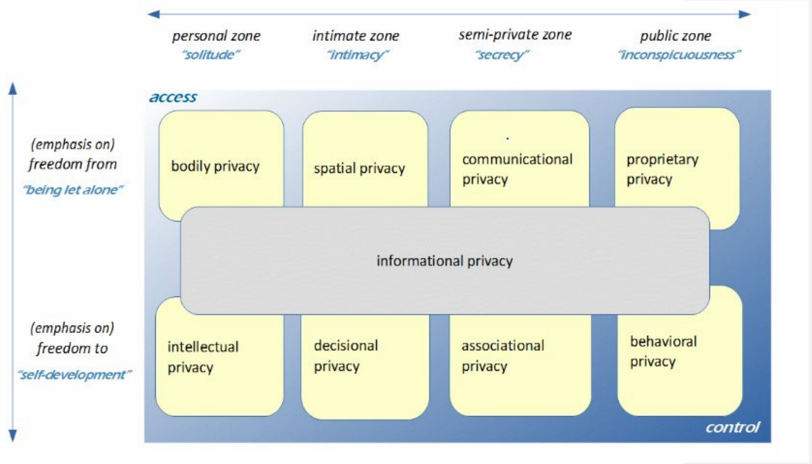
[https://blog.hubspot.com/marketing/google-search-statistics]

How privacy friendly are our search queries?

Privacy Friendly Web Search

Importance

- Right to privacy and data protection
- Confidentiality
  - Mass-surveillance Government
  - Instead of limiting surveillance to viable threats
  - Provides security and freedom of expression
- Anonymity
  - Not everything you search is public knowledge… Or is it?
  - Data can tie loose ends together quickly
- Personal security
  - Influencing decisions/behaviour
  - Crimes
  - Other malicious behaviour
- User control
  - Being in control over your own data
  - Avoid profiling
  - Targeted ads



[T. Sharon and B.-J. Koops, "The ethics of inattention: revitalising civil inattention as a privacy-protecting mechanism in public spaces," *Ethics and Information Technology*, vol. 23, no. 3, pp. 331–343, Jan. 2021]

Legal context

GDPR

- Doesn’t mention web search specifically
- Does establish requirements regarding the processing of personal data, which apply to any service
- Extraterritorial reach

Other laws outside of GDPR

- Charter of Fundamental Rights of the European Union, article 8 (1):
  - Everyone has the right to the protection of personal data concerning him or her [EU Fundamental Rights Agency]
- Universal Declaration of Human Rights, article 12:
  - No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honor and reputation. Everyone has the right to the protection of the law against such interference or attacks. [United Nations]

Legal benefits of Privacy Friendly Web Search

- Privacy-friendly web search services often adhere to stringent privacy laws and regulations, providing users with a level of legal protection for their data.
  - Mitigates part of the legal risks of data breaches
- This compliance contributes to a more trustworthy and accountable online environment.

Outline

Query Obfuscation

Private Web Search Using Proxy-Query Based Query Obfuscation Scheme

Introduction

Definition

- Obfuscation: “The act of making something less clear and less easy to understand.” [Cambridge Dictionary]

General Mechanism

- Obfuscation-Based Private Web Search (OB-PWS) solutions make it possible for users to conceal their interests
- Search engines maintain a database of pointers to pages in the Web
- The pointers are indexed by keywords and relate to the content of the pages
- The search engine compiles a list of web pages
  - based on the keywords of the users
- Search queries are closely related to our interests and concerns
  - They can be used to perform user profiling.

Obfuscation-Based Private Web Search (OB-PWS)

Requirements for a Modern OB-PWS scheme

Importance Requirements

- Failing to accomplish one or more requirements impacts:
  - Performance
  - Effectiveness
  - Web Search Privacy (WSP) trust
  - Security

Requirements

1. Hide users
2. Retrieval effectiveness
3. Trust in PWS-scheme
4. Non-noisy cover queries
5. Web Search Privacy (WSP) for related queries
6. WSP on multiple devices
7. Storage and computation
8. Web search without true query

General OB-PWS schemes

Private Information Retrieval (PIR) schemes

- Techniques used to privately retrieve information from databases:
  1. Partitioning the databases
  2. Distributing them onto multiple servers
- Two-server PIR scheme
  - the query is split between two servers
  - combining the responses retrieves the desired information
  - the server doesn’t learn what the user exactly accessed

Proxy-based PWS schemes

- Retrieves information through anonymous web browsing
- Disadvantage: real IP-addresses are not hidden from proxy servers
- Example: TOR

Obfuscation-Based Private Web Search (OB-PWS)

General OB-PWS schemes

Collaborative peer-to-peer PWS schemes

- Achieves private web searches through users’ collaboration
- It uses peer-to-peer protocols to indirectly submit search queries
- Web Search Engine (WSE) cannot identify actual users
- Disadvantages?
  - Requires availability of the same set of peers
  - Slow query response time
  - Complex implementation of Cryptographic protocols
- Example: YaCy

Query scrambling based PWS schemes

- Scrambles true search queries
- Objective: hide intent behind the true query by retrieving information that is more general than specific
- Disadvantage: Reduces the retrieval effectiveness
- Example: QS

Obfuscation-Based Private Web Search (OB-PWS)

General OB-PWS schemes

Query obfuscation based PWS schemes

- Automatic generation of cover queries are sent
- Different techniques:
  - Query log-based approach
  - Pseudo-cover queries
- Challenge?
  - create non-noisy cover queries
- Disadvantage: Related cover queries are not provided
- Examples: TrackMeNot, GooPIR, Qu-OB-PWS

Obfuscation-Based Private Web Search (OB-PWS)

General OB-PWS schemes

Query obfuscation based PWS schemes by considering relatedness with past queries

- Relatedness should be considered while producing cover queries
- This can be accomplished by capturing relatedness across cover queries via a collection of hierarchically arranged topics.
- It creates cover queries from the subject hierarchy rather than just random non-noisy cover queries.
- As a result, if a series of related true queries have relatedness between them, the subject hierarchy assures that relatedness exists between cover queries as well.
- Disadvantages:
  - It indexes a large collection on client computers
  - Query relatedness across different devices cannot be captured

Obfuscation-Based Private Web Search (OB-PWS)

Proxy-query based PWS - General Mechanism

1. A user issues a true query
2. The true query is transformed to the proxy query
3. The Information Retrieval system generates and processes all cover queries
4. The Information Retrieval system returns a back-rank list of all cover queries to the client machine
5. The client machine only returns the result of the true query

- The scheme builds proxy mapping using queries of topics instead of individual terms
- This is useful for both single and sequence related queries
- Given that it is hard to revert the proxy query back to the true query

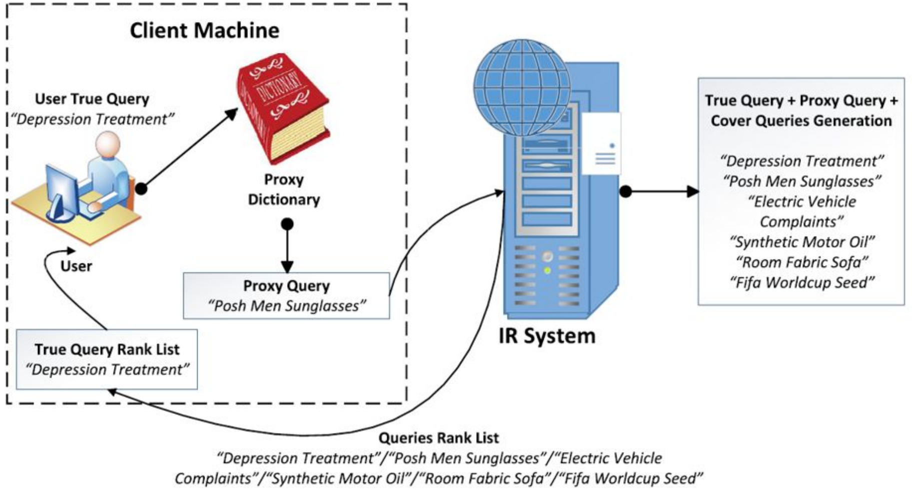


FIGURE 1. An architecture of proxy-query based private web search.

Proxy-query based PWS - Generating Proxy-Query Mapping (1)

Complexity

- System needs to explore optimal mapping for:

  $O(Q^M)$

Initial proxy-query mapping

- For each query q ∈ Q, topics are rated using TF-IDF-scores
- q is assigned to the topic with the greatest TF-IDF-score
- The algorithms partition the queries into P proxy groups
- For each proxy group, the algorithm selects queries from T topics with not more than one query from each topic

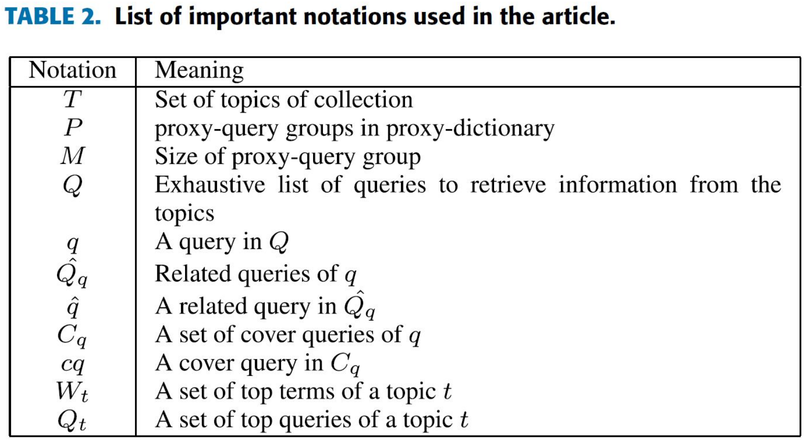


TABLE 2. List of important notations used in the article.

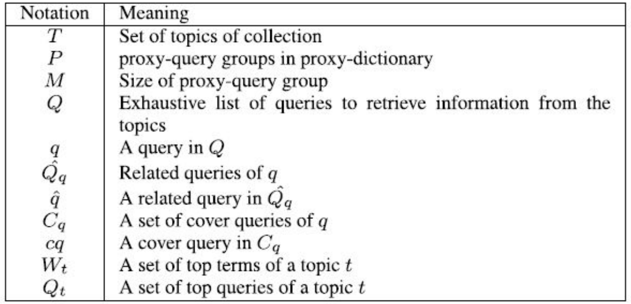
Notation	Meaning
T	Set of topics of collection
P	proxy-query groups in proxy-dictionary
M	Size of proxy-query group
Q	Exhaustive list of queries to retrieve information from the topics
q	A query in Q
$\hat{Q}_q$	Related queries of q
$\hat{q}$	A related query in $\hat{Q}_q$
C_q	A set of cover queries of q
cq	A cover query in C_q
W_t	A set of top terms of a topic t
Q_t	A set of top queries of a topic t

Obfuscation-Based Private Web Search (OB-PWS)

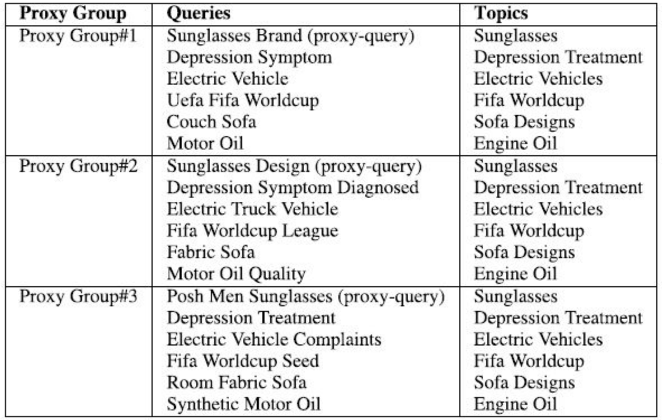
Proxy-query based PWS - Generating Proxy-Query Mapping (2)

Related queries and cover queries

- Queries in $\hat{Q}_q$ are ranked by similarity with q (average cosine similarity)
- Query similarity is computed using the average cosine similarity from the top k documents
- q is a proxy query that a user issues to an Information Retrieval (IR) system.
- Optimal query to proxy group mapping:
  - the cover queries of $\hat{q}$ are also related the cover queries of q



Notation	Meaning
T	Set of topics of collection
P	proxy-query groups in proxy-dictionary
M	Size of proxy-query group
Q	Exhaustive list of queries to retrieve information from the topics
q	A query in Q
$\hat{Q}_q$	Related queries of q
$\hat{q}$	A related query in $\hat{Q}_q$
C_q	A set of cover queries of q
cq	A cover query in C_q
W_t	A set of top terms of a topic t
Q_t	A set of top queries of a topic t



Proxy Group	**Queries**	**Topics**
Proxy Group#1	Sunglasses Brand (proxy-query)	Sunglasses
	Depression Symptom	Depression Treatment
	Electric Vehicle	Electric Vehicles
	Uefa Fifa Worldcup	Fifa Worldcup
	Couch Sofa	Sofa Designs
	Motor Oil	Engine Oil
Proxy Group#2	Sunglasses Design (proxy-query)	Sunglasses
	Depression Symptom Diagnosed	Depression Treatment
	Electric Truck Vehicle	Electric Vehicles
	Fifa Worldcup League	Fifa Worldcup
	Fabric Sofa	Sofa Designs
	Motor Oil Quality	Engine Oil
Proxy Group#3	Posh Men Sunglasses (proxy-query)	Sunglasses
	Depression Treatment	Depression Treatment
	Electric Vehicle Complaints	Electric Vehicles
	Fifa Worldcup Seed	Fifa Worldcup
	Room Fabric Sofa	Sofa Designs
	Synthetic Motor Oil	Engine Oil

Obfuscation-Based Private Web Search (OB-PWS)

Proxy-query based PWS - Generating Proxy-Query Mapping (3)

Equations

- Equation 3: Captures average similarity between q and cq to $\hat{q}$ and $c\hat{q}$
- Equation 2: Captures the average similarity between a query q and its related queries $\hat{Q}$
- Equation 1: Captures how well the queries in Q are related to each other
- <u>Main goal: Achieve a high score for fitness(Q)</u>

$$fitness(Q) = \frac{\sum_{q}^{Q} rQ(q)}{|Q|} \tag{1}$$

$$rQ(q) = \frac{\sum_{\hat{q}}^{\hat{Q}} cQ(q, \hat{q})}{|\hat{Q}|} \tag{2}$$

$$cQ(q, \hat{q}) = \frac{\sum_{c\hat{q}}^{C_{\hat{q}}} |cos(q, \hat{q}) - cos(cq, c\hat{q})|}{|C_{\hat{q}}|} \tag{3}$$

Obfuscation-Based Private Web Search (OB-PWS)

Proxy-query based PWS - Greedy Search Hill Climbing Heuristic

Heuristic mechanism

- Optimal mapping should be explored for good effectiveness of proxy groups
- The heuristic iteratively improves the performance of the current solution:
    1. It explores neighbour solutions by making many random changes to the current solution.
    2. If neighbour solutions improve the fitness, the heuristic replaces the current with the new solution.
    3. It continues exploring the search space and ends when no improvement is obtained after many changes.
    4. The system indexes the discovered mapping in the proxy dictionary

- Main objective: To improve fitness of initial proxy-query groups using Equation 1

Obfuscation-Based Private Web Search (OB-PWS)

Proxy-query based PWS - Example

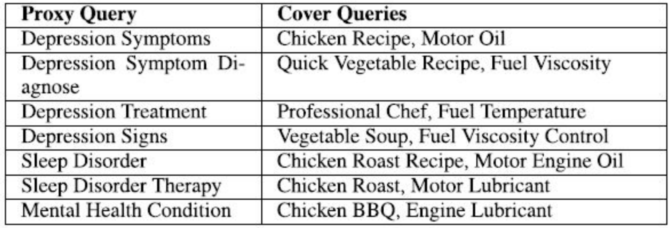
Before heuristic

- Topic: "Depression Treatment"
- Given the sequence of related queries *Depression Symptoms* and *Depression Symptom Diagnose*, the proxy-queries *Chicken Recipe* and *Quick Vegetable Recipe*, and cover queries *Motor Oil* and *Fuel Viscosity.*
- Does this mapping have a bad fitness?
  - Yes, due to the difference in relatedness of the true, proxy and cover queries
  - This makes it easier for the Information Retrieval system to identify the true query

After heuristic

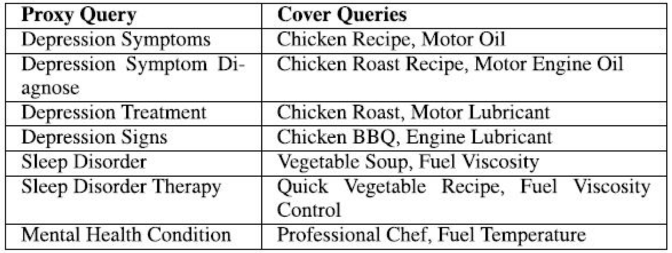
- Related, proxy and cover queries have stronger relationships between themselves
- This makes it harder for the Information Retrieval system to identify the true query

TABLE 4. **Initial Proxy Groups for the topics (Depression Treatment, Chicken Roast Recipe, and Motor Oil).**



Proxy Query	Cover Queries
Depression Symptoms	Chicken Recipe, Motor Oil
Depression Symptom Diagnose	Quick Vegetable Recipe, Fuel Viscosity
Depression Treatment	Professional Chef, Fuel Temperature
Depression Signs	Vegetable Soup, Fuel Viscosity Control
Sleep Disorder	Chicken Roast Recipe, Motor Engine Oil
Sleep Disorder Therapy	Chicken Roast, Motor Lubricant
Mental Health Condition	Chicken BBQ, Engine Lubricant

TABLE 5. **Improved proxy groups after searching optimal proxy-query mapping.**



Proxy Query	Cover Queries
Depression Symptoms	Chicken Recipe, Motor Oil
Depression Symptom Diagnose	Chicken Roast Recipe, Motor Engine Oil
Depression Treatment	Chicken Roast, Motor Lubricant
Depression Signs	Chicken BBQ, Engine Lubricant
Sleep Disorder	Vegetable Soup, Fuel Viscosity
Sleep Disorder Therapy	Quick Vegetable Recipe, Fuel Viscosity Control
Mental Health Condition	Professional Chef, Fuel Temperature

Distributed Protocols

A paper by Mohib Ullah, Rafiullah Khan, Muhammad Inam Ul Haq, Atif Khan, Wael Alosaimi, Muhammad Irfan Uddin, and Abdullah Alharbi

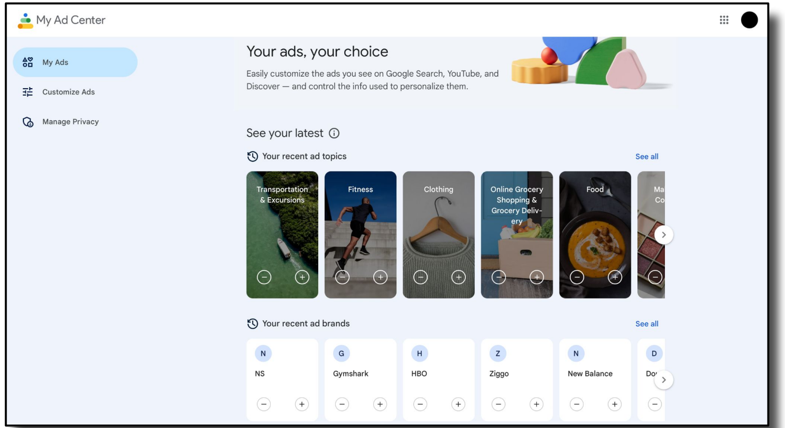
Multi-Group ObScure Logging (MG-OSLo) A Privacy-Preserving Protocol for Private Web Search

Introduction

- What is user profiling?
- User profiling is used by web search engines to maintain information on its users and make a profile around the user to broadcast relatable content to them
- A user's query may reveal health information, gender orientation, religion, politics, faith, believes, etc., which may be deemed sensitive for a possessor
- A user's queries often contain important information like Unique User ID, name, user's employer's details, location, etc.
- The disclosure of a query log poses severe risks in terms of privacy. Preserving the web search privacy of a user is the genuine concern of today's online life

Distributed Protocols

Google’s User Profile for advertisement



My Ad Center

My Ads

Customize Ads

Manage Privacy

Your ads, your choice

Easily customize the ads you see on Google Search, YouTube, and Discover — and control the info used to personalize them.

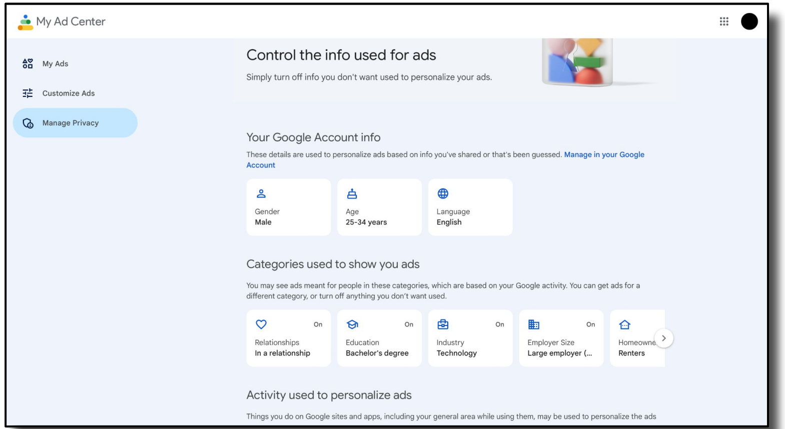
See your latest

Your recent ad topics — See all

Transportation & Excursions · Fitness · Clothing · Online Grocery Shopping & Grocery Deliv- ery · Food · Ma Co

Your recent ad brands — See all

NS · Gymshark · HBO · Ziggo · New Balance · Do



My Ad Center

My Ads

Customize Ads

Manage Privacy

Control the info used for ads

Simply turn off info you don't want used to personalize your ads.

Your Google Account info

These details are used to personalize ads based on info you've shared or that's been guessed. Manage in your Google Account

Gender: Male

Age: 25-34 years

Language: English

Categories used to show you ads

You may see ads meant for people in these categories, which are based on your Google activity. You can get ads for a different category, or turn off anything you don't want used.

Relationships (On): In a relationship

Education (On): Bachelor's degree

Industry (On): Technology

Employer Size (On): Large employer (...

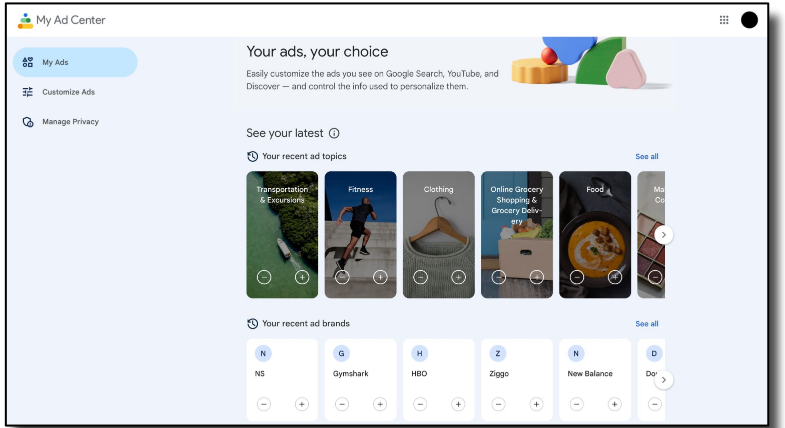
Homeowne: Renters

Activity used to personalize ads

Things you do on Google sites and apps, including your general area while using them, may be used to personalize the ads

Distributed Protocols

Google’s User Profile for advertisement



My Ad Center

My Ads
Customize Ads
Manage Privacy

Your ads, your choice

Easily customize the ads you see on Google Search, YouTube, and Discover — and control the info used to personalize them.

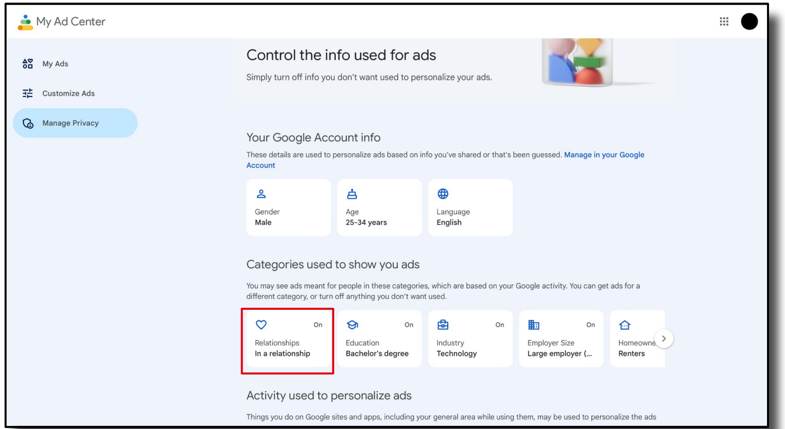
See your latest

Your recent ad topics — See all

Transportation & Excursions
Fitness
Clothing
Online Grocery Shopping & Grocery Deliv-ery
Food

Your recent ad brands — See all

NS
Gymshark
HBO
Ziggo
New Balance



My Ad Center

My Ads
Customize Ads
Manage Privacy

Control the info used for ads

Simply turn off info you don't want used to personalize your ads.

Your Google Account info

These details are used to personalize ads based on info you've shared or that's been guessed. Manage in your Google Account

Gender	Age	Language
Male	25-34 years	English

Categories used to show you ads

You may see ads meant for people in these categories, which are based on your Google activity. You can get ads for a different category, or turn off anything you don't want used.

Relationships (On)	Education (On)	Industry (On)	Employer Size (On)	Homeowne
In a relationship	Bachelor's degree	Technology	Large employer (...	Renters

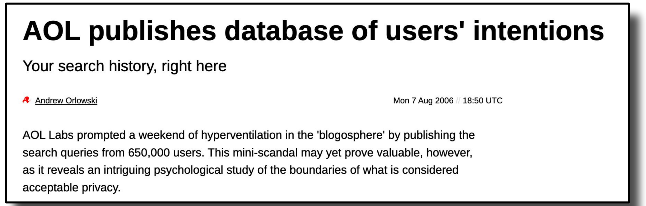
Activity used to personalize ads

Things you do on Google sites and apps, including your general area while using them, may be used to personalize the ads

Disclaimer : This detail is incorrect!

Radboud University

Do you think it is possible to identify you based on your search queries?

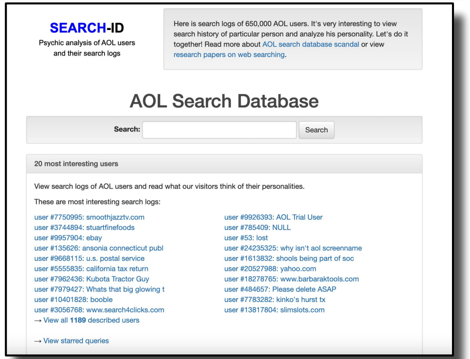


AOL publishes database of users' intentions

Your search history, right here

Andrew Orlowski — Mon 7 Aug 2006 18:50 UTC

AOL Labs prompted a weekend of hyperventilation in the 'blogosphere' by publishing the search queries from 650,000 users. This mini-scandal may yet prove valuable, however, as it reveals an intriguing psychological study of the boundaries of what is considered acceptable privacy.



SEARCH-ID
Psychic analysis of AOL users and their search logs

Here is search logs of 650,000 AOL users. It's very interesting to view search history of particular person and analyze his personality. Let's do it together! Read more about AOL search database scandal or view research papers on web searching.

AOL Search Database

20 most interesting users

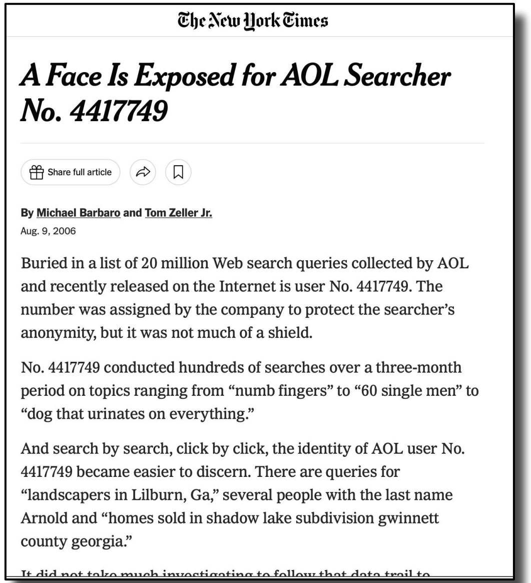
View search logs of AOL users and read what our visitors think of their personalities.

These are most interesting search logs:

user #7750995: smoothjazztv.com	user #9926393: AOL Trial User
user #3744894: stuartfinefoods	user #785409: NULL
user #9957904: ebay	user #53: lost
user #135626: ansonia connecticut publ	user #24235325: why isn't aol screenname
user #9668115: u.s. postal service	user #1613832: shools being part of soc
user #5555835: california tax return	user #20527988: yahoo.com
user #7962436: Kubota Tractor Guy	user #18278765: www.barbaraktools.com
user #7979427: Whats that big glowing t	user #484657: Please delete ASAP
user #10401828: booble	user #7783282: kinko's hurst tx
user #3056768: www.search4clicks.com	user #13817804: slimslots.com

→ View all **1189** described users

→ View starred queries



The New York Times

A Face Is Exposed for AOL Searcher No. 4417749

Share full article

By Michael Barbaro and Tom Zeller Jr.

Aug. 9, 2006

Buried in a list of 20 million Web search queries collected by AOL and recently released on the Internet is user No. 4417749. The number was assigned by the company to protect the searcher's anonymity, but it was not much of a shield.

No. 4417749 conducted hundreds of searches over a three-month period on topics ranging from "numb fingers" to "60 single men" to "dog that urinates on everything."

And search by search, click by click, the identity of AOL user No. 4417749 became easier to discern. There are queries for "landscapers in Lilburn, Ga," several people with the last name Arnold and "homes sold in shadow lake subdivision gwinnett county georgia."

It did not take much investigating to follow that data trail to

MG-OsLo as a Distributed Scheme

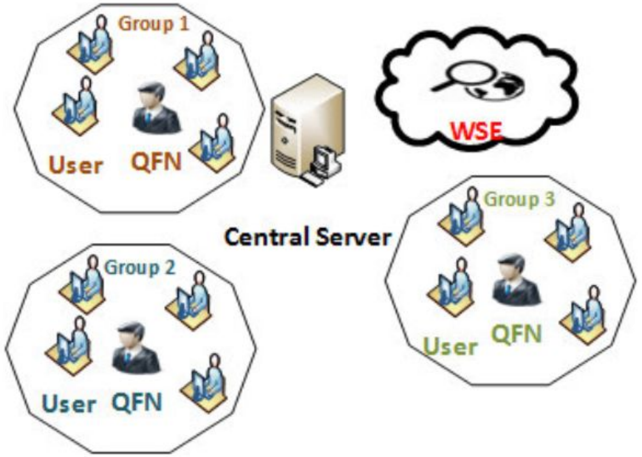
- Idea behind distributed scheme is to distort the user profile that the web search engine can generate
- Distributed scheme consists of multiple users forming a group
- A user other than the query originator forwards the queries of the user
- This makes for the search engine difficult to track which query belonged to which user and therefore makes user profiling difficult

Main Objective of MG-OsLo

- User’s query and its results must remain concealed from the group members
- The unlinkability between the user and the query must be assured, and WSE should not be able to build an accurate user profile
- User should get answer to all their queries – Functionality

Entities in MG-OsLo

- User
- Central Server
- Query Forwarding node
- Web Search Engine



Steps in MG-OsLo

- Connection setup, group creation, and QFN selection
- Query sending process
- Query shuffling
- Query forwarding to WSE and result processing

Distributed Protocols - MG-OsLo

Connection Setup, Group Creation, And QFN Selection

- M Groups each Group has K users
- CS listens to connection requests from users
- User must send connection request to CS to covertly search a query through MG-OsLO
- CS places user in group having vacant slot or creates new group.
- CS collects (IP and port number)
- Select QFN and asks them for encryption key and the group ID (G_ID)
- CS broadcasts the information about all groups, users in each group, and the details about the QFNs
- Every user is selected as QFN in round robin fashion by the CS

Distributed Protocols - MG-OsLo

Query Sending Process

- After CS broadcasts the group information, each user holds details of users in the group and the QFN
- User generates query 'q', an encryption key K_Ui, and a query ID (q_ID)
- The K_Ui is a 128bits AES share key used to encrypt the result (r) for the query ’q’
- User matches the q_ID to recognize that the result(r) is for his or her query(q)
- User makes QMsg, query message, by concatenating ’q,’ K_Ui, and q_ID
- How do we achieve confidentiality of this query, ‘q’ ?
- User selects the QFN from the list and encrypts QMsg with QFN’s public key resulting in eQ encrypted query
- User concatenates G_ID and eQ, producing an encrypted query message eQ_Msg.
- QFN uses the G_ID to confirm that query (q) is encrypted with their public key
- The eQ_Msg is shuffled in two stages to disassociate the query and the user

Query Shuffling

- Query shuffling takes place in 2 steps
  - Intra group shuffling - to avoid group members from identifying the query originator
  - Inter group shuffling - hides user's group identity and disassociates the query and group
- Intra Group Shuffling
  - user tosses coin to determine destination of eQ_Msg
  - If toss is heads, eQ_Msg is forwarded to QFN ending the intra group shuffling
  - If toss is tails, eQ_Msg is forwarded to random user of the group
  - shuffling continues until eQ_Msg reaches the QFN
  - When QFN receives eQ_Msg it checks the GID to identify if query q is encrypted with their public key if yes, eQ_Msg is decrypted

Query Shuffling

- Inter Group Shuffling
    - QFN flips a coin in the inter-group shuffling heads means eQ_Msg is forwarded to CS which in turn forwards to other QFNs
    - In case of tails, QFN sends the eQ_Msg to another random QFN.
    - QFN who receives an eQ_Msg from CS checks the G_ID. If it matches QFN decrypts eQ_Msg. Otherwise query is dropped

Distributed Protocols - MG-OsLo

Query Forwarding to WSE and Result Processing

- eQ_Msg arrives at the destined QFN, the process of decryption starts
- QFN decrypts the eQ_Msg using the private key to acquire the query message Q_Msg
- QMsg is dis-concatenated to acquire the query ‘q’, query ID (q_ID), and encryption key (eK_Ui)
- query (q) is forwarded to the WSE, which processes the ‘q’ and returns the query results (Q_Result).
- The QFN encrypts the result (Q_Result) with the user’s encryption key (eK_Ui), making an En_Res
- En_Res is concatenated with q_ID, thus creating an encrypted answer message eAns_Msg
- forwards eAns_Msg to CS, which sends the results to all QFNs
- each QFN broadcasts the eAns_Msg in their respective group. The user who has the decryption key decrypts the eAns_Msg
- The user uses the q_ID to confirm that the results are for the query ‘q’ what he or she has sent
- q_ID matches, the user decrypts the packet with the decryption key and retrieves the query’s result.

How MG-OsLo computes privacy

- **Local privacy**, which is the privacy maintained within group i.e. a group member should not be able to identify the query originator within their group

- **User Profile Privacy** which is the amount of information exposed of a certain user towards their profile and it is evaluated for two situations:
    - First, in which a self-query submission is allowed
    - Second, in which a self-query submission is not allowed.

User Profiling

Local Privacy computation

The local privacy of a user depends on the way the CS groups the users. The users can be grouped in different ways, whereby a user's privacy gets affected by the users' grouping.

Non-overlapping group design: Each user appears in one group; all groups are distinct. Second, overlapping group design: A user appears in multiple groups simultaneously.

- Non Overlapping Design
    - Privacy Relative to QFN

      $$Pr(S = Ui, M = Sl | P = QFN) = \frac{1}{(K(b-1))}$$

    - Privacy Relative to the Core Server

      $$Pr(S = Ui, M = Sl | P = QFN, S \notin Mc) = \frac{1}{K(b-1)}$$

    - Privacy Relative to Group Users

      $$Pr(S = Ui | M = Sl,\ P = QFN) = \frac{1}{K}$$

Local Privacy computation

- Overlapping Design A user appears in multiple groups in the overlapping group design and the QFN is supposed to forward the queries of all peer users to the WSE
    - Query source and QFN belong to different groups

      $$\frac{r}{(b-1)\cdot K}$$

    - Query source and QFN belong to same groups

      $$\frac{1}{\lambda \cdot (K-2)}\ if\ \ i \neq j\, and\, j \neq QFN$$

Distributed Protocols - MG-OsLo

Results and Simulations

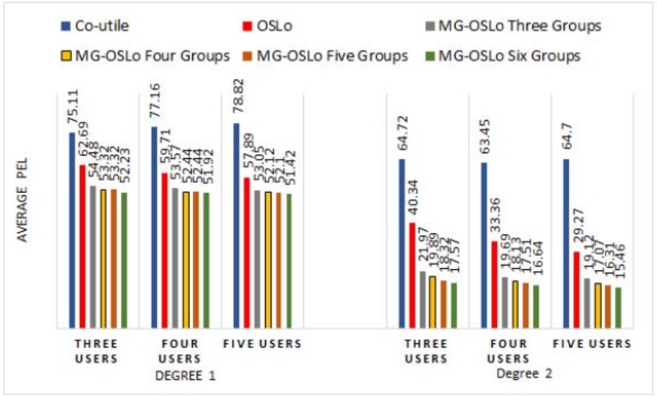
- The MG-OSLo is simulated for two situations, i.e.
  - self-query submission is allowed
    - a user forwards one of his/her queries when forwarding other group users' queries
  - self-query submission is not allowed
    - The user only delivers the queries of group users but not his/her query to the WSE.

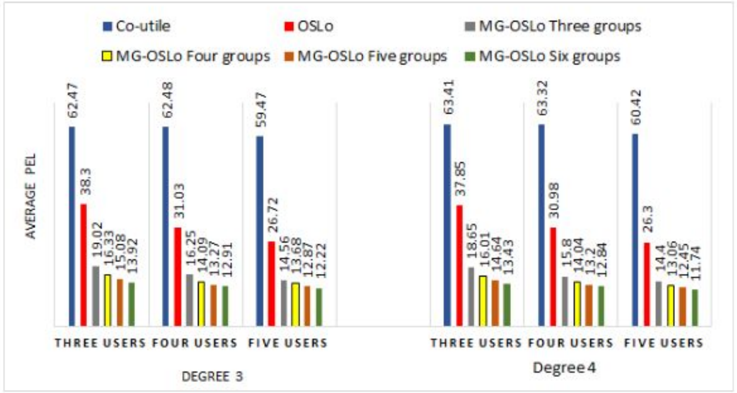
- The user's profile privacy attained by executing MG-OSLo is compared with the state-of-art OSLo and Co-utile protocol

Distributed Protocols - MG-OsLo

Results and Simulations - Profile privacy comparison

Self Query Submission Allowed



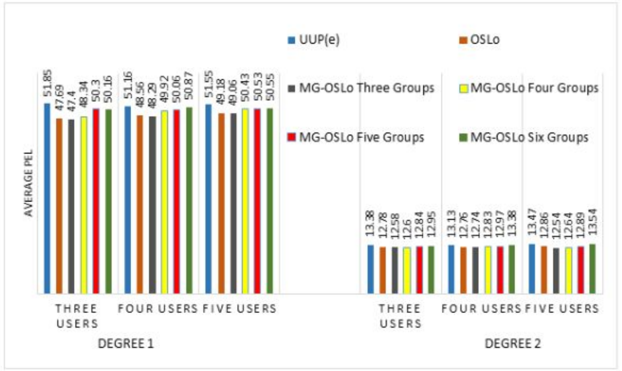


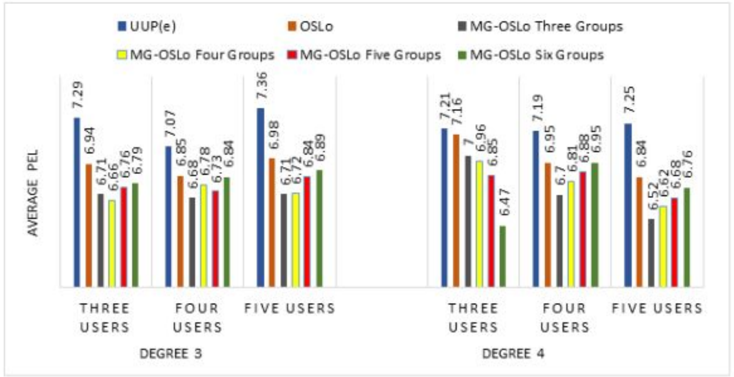
- The result indicates that MG-OSLo achieves an average PEL of 54.48% for the group count of three, each having three users at degree 1 of the ODP hierarchy.
- When the group count increased to four, the value dropped to 53.32%.
- The results depict that increasing the group count declines the average PEL value to 52.23%.
- Likewise, when the group size is increased to four users, the results illustrate that MG-OSLo attains an average PEL of 53.57% for the group count of three

Distributed Protocols - MG-OsLo

Results and Simulations - Profile privacy comparison

Self Query Submission Not Allowed





- The results infer that when self-query submission is not allowed, a group count of three, each containing three users, achieves the minimum average PEL because the user's profile is obfuscated to its maximum level.
- Therefore, based on the result, it is recommended that the group count of three and each group having three users provide the best results in terms of average PEL for a situation when self-query submission is not allowed.

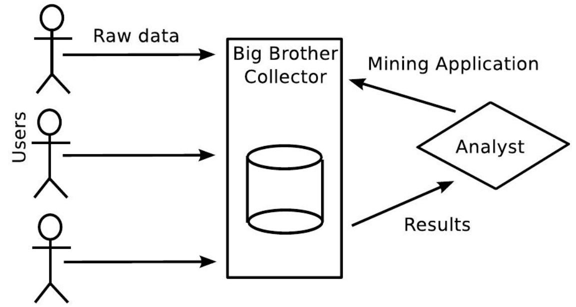
Radboud University

Case study:
CrowdLogging

A paper by Henry Feild, James Allan & Joshua Glatt

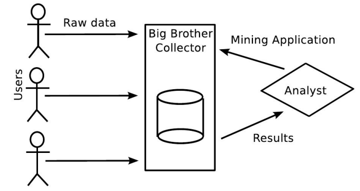
CrowdLogging
Introduction

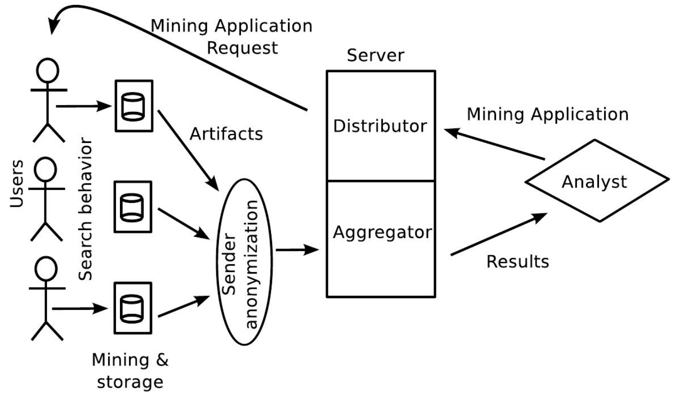
- 'Big brother'
  - Raw query logs
  - Easy to find out A LOT about one user
    - Browser (tracking) cookies
  - Very revealing mining actions possible
    - AOL incident
  - Centralized approach:
    - Users have no control
    - User's privacy is in hands of collector
    - Shared log data has reduced utility for research



The idea

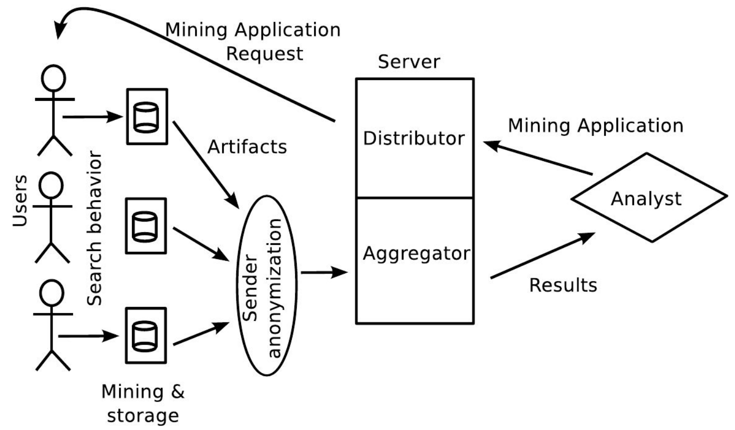
- CrowdLogging
- Goals:
  1. Source of search queries is unknown
  2. Queries containing revealing information should not be released
  3. Researchers still have access to useful data
- How?
  - Decentralized approach
  - (Partially) Encrypted query logs
  - Anonymized query logs
  - Rare search queries should be hidden
  - Methods to mine encrypted data
- Three-layer framework
  1. System
  2. Privacy policies
  3. Applications





CrowdLogging
System

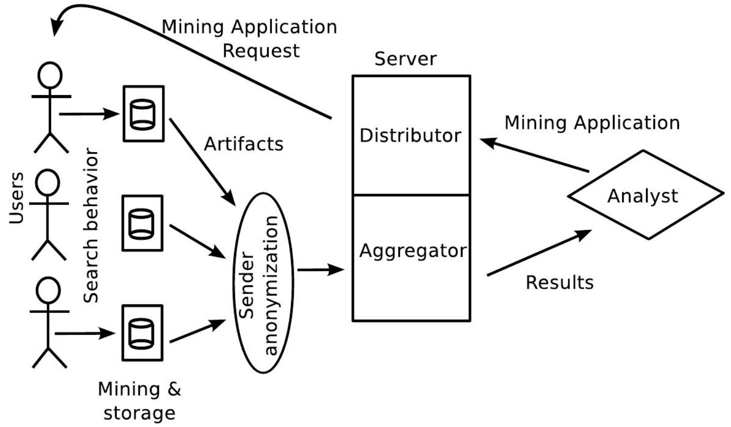
- Distributed
  - No more storage to big database
  - Distributed storage of raw data
  - Local query logs
  - Full control
- Private
  - Introduce: Search artifacts
    - Only the results of a mining task can leave a computer
  - Encryption: “Secret sharing”
  - Key pieces
    - Artifact, ID, k, n (where k<n)
    - K-anonymity
- Anonymous
  - Anonymizing system
    - Metadata
  - Aggregator does not release key part information
    - Part ID



Privacy policies

- A variable privacy factor
- Four questions:
  1. What can be mined and sent to server?
  2. How is data packaged to server?
  3. How is data aggregated at server?
  4. What is released to analysts?

- Sample of policies:
  1. Artifact frequency thresholding
  2. User frequency thresholding
  3. Artifact frequency differential privacy
  4. User frequency differential privacy



Privacy policies

Artifact frequency thresholding (FT_a)

Analyst can access search artifact if occurrence >= k

1. What can be mined?
   - No constraints
2. How does it get packaged?
   - Encrypted + random key part
3. How is data aggregated?
   - Combination of k key parts
4. What is released?
   - Pairs of artifacts and counts

- Weak privacy policy
- User could enter query that is both identifying and sensitive k times
- Can be used in conjunction with more private policies

Privacy policies

User frequency thresholding (FT_u)

Analyst can access search artifact if distinct users >= k

1. What can be mined?
   - No constraints
2. How does it get packaged?
   - Encrypted + same key part for same artifact of a user
3. How is data aggregated?
   - Combination of k key parts
4. What is released?
   - Pairs of artifacts and counts

- If >= k users, artifact likely not sensitive
- (Not as) weak privacy policy
- Sensitive information of one user: would be released under FT_a but not under FT_u
- Can be used in conjunction with more private policies

Differential privacy

- Drawback up till now: no provable guarantee of privacy.
- Solution: Differential Privacy (Cynthia Dwork, 2006)
  - Motivation: Database and query result in noisy answer
    - Querier has low probability of determining if database contains a person
  - Important: Quantifies privacy loss
- Example: Algorithm for producing query click graph from search log:
  1. Take **first d queries** from each user
  2. Add **noise** to frequency of query **in dataset**, select for release if > k
  3. For released queries, add **noise** to frequency of query **in released subset**
  4. Add noisy click counts for top 10 results of query
- Three constraints:
  1. Number of mined artifacts of a user must be limited (d)
  2. Laplace noise must be added to any artifact count before comparing with threshold
  3. Small number of mining tasks per dataset: Each mining task depletes privacy quota
- We introduce two variations of the algorithm for CrowdLogging
  - Queries replaced with artifacts
  - Omit step of releasing URL click counts

Artifact frequency differential privacy (DP_a)

- Algorithm
  1. Take **first d artifacts** from each user
  2. Add **noise** to each artifact frequency, select for release if > k_a
  3. For released artifacts, add **noise** to frequencies

1. What can be mined?
   - Anything, but only d artifacts are sent to server
2. How does it get packaged?
   - Encrypted + random key part + encrypted user ID
3. How is data aggregated?
   - First ensure that per user there are d artifacts, then combination of k_a key parts
4. What is released?
   - Pairs of artifacts and noisy counts

- Strong and measurable privacy policy

Privacy policies

User frequency differential privacy (DP_u)

- Algorithm
  1. Take **first d artifacts** from each user
  2. Add **noise** to frequency of users with specific artifact, select for release if > k_u
  3. For released artifacts, add **noise** to frequencies

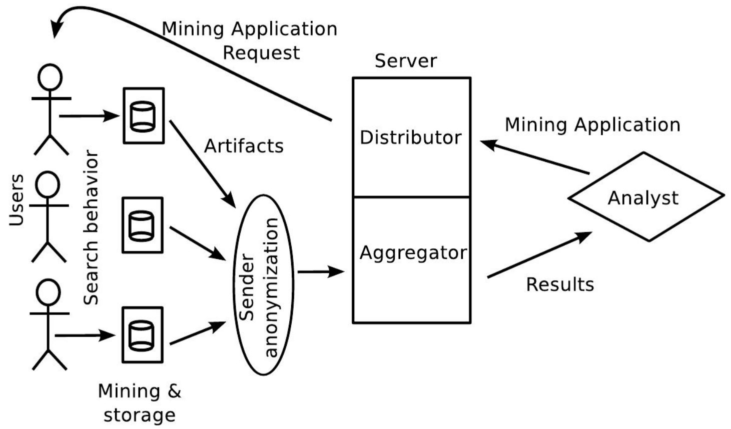
1. What can be mined?
   - Anything, but only d artifacts are sent to server
2. How does it get packaged?
   - Encrypted + same key part for same artifact of a user + encrypted user ID
3. How is data aggregated?
   - First ensure that per user there are d artifacts, then combination of k_u key parts
4. What is released?
   - Pairs of artifacts and noisy counts

- Strong and measurable privacy policy

CrowdLogging
Applications

- Applications are the mining tasks
- Distributed to consenting users
- Input: Local search log
- Output: Search artifacts limited by PP
- The processing happens at the uer

- Sample of mining applications:
  1. Query frequency mining
  2. Query reformulation mining
  3. Query-URL frequency mining
  4. Learning to rank feature mining



Applications

Query frequency mining

- Goal: Prevent direct occurence of original queries
- Interest: How rare are searches?
- Use both original and normalized query (semantically identical)
  - Normalizing: “doggfood” becomes “dog food”
- Primary private field & secondary private field
  - Primary = normalized
  - Secondary = original
  - Encrypted with same key derived from primary field
    - Secondary only available if primary can be decrypted
- Aggregation based on primary private field
  - Once enough key parts, decryption is possible
  - Privacy policy decides this
- Generation of tuples
- Makes frequency mining possible
  - Both encrypted and decrypted
  - Encrypted version still useful

$$\langle E(N(query_1)), E(query_1), key\text{-}portion \rangle$$

$$\langle \text{“dog food”}, \quad \text{“DOG FOOD”}, \quad 1 \rangle$$
$$\langle \text{“dog food”}, \quad \text{“dog} \quad \text{food”}, \quad 100 \rangle$$

Query reformulation mining (EXTRA)

- When user changes query shortly after search
- Interest: Are they connected?
  - Over half of the users in 24h changed query
  - Spelling, related search, expansion, alternate formulation
- Again with primary and secondary private field
  - Normalization
  - Replace with pairs of queries
  - Reveal at k distinct queries
- Can be run together with freq. mining (by PP)
  - What percentage of queries get rewritten to what?
  - Lost: infrequent reformulation pairs

$$\langle E(N(q_{1,1} : q_{1,2})), E(q_{1,1} : q_{1,2}), \textit{key-portion}\rangle$$
$$\langle E(N(q_{2,1} : q_{2,2})), E(q_{2,1} : q_{2,2}), \textit{key-portion}\rangle$$

“dog”	$\rightarrow$	“puppy”	:	50%
“dog”	$\rightarrow$	“dog food”	:	20%
“dog”	$\rightarrow$	?	:	30%

Applications

Query-URL frequency mining (EXTRA)

- When URLs are clicked after search
- Interest: Relevance
  - What pages are clicked after query?
  - What queries result in certain pages?
- Same approach, second query with page
  - Normalization + release at k
- Server does not know who issued queries
- Variations
  - Can increase commonality by shortening URL
  - Can drop the query to analyze popular URLs

$$\langle E(N(q_1 : URL_1)), E(q_1 : URL_1), \textit{key-portion}\rangle$$
$$\langle E(N(q_1 : URL_2)), E(q_1 : URL_2), \textit{key-portion}\rangle$$

Learning to rank feature mining (EXTRA)

- When we want to rank search results
- Interest: Setting up an LTR-system
  - Based on page features and relevance w.r. to query
  - What features are most relevant for user?
- Idea: Concat query, URL, and features
  - Normalization + release at k
- Pitfall: Full feature list is unlikely to be complete
  - Solution: Break up the feature list
- URL shortening still applicable
- Can also map features to buckets

$$\langle E(N(q_1 : URL_1 : f_1 : f_2 \ldots)), E(\ldots), \textit{key-portion}\rangle$$
$$\langle E(N(q_1 : URL_2 : f_1 : f_2 \ldots)), E(\ldots), \textit{key-portion}\rangle$$

$$\langle E(N(q_1 : URL_1 : \mathrm{F1}{:}f_1)), E(q_1 : URL_1 : \mathrm{F1}{:}f_1), \textit{key-portion}\rangle$$
$$\langle E(N(q_1 : URL_1 : \mathrm{F2}{:}f_2)), E(q_1 : URL_1 : \mathrm{F2}{:}f_2), \textit{key-portion}\rangle$$

Simulations

- Motivation: AOL blunder
  - 3 months of logs in 2006
  - 10M unique queries, 21M total, 650K users
- Motivation: MSN logs
  - Session IDs instead of users
  - 1 month of logs in 2006
  - 6.6M unique queries, 13M total, 7.5M users
- What would've been the impact using CrowdLogging?
- Motivation: What data could have been discovered…
  - …but can't because of distributed, private and anonymous approach
    1. How much data would be lost?
    2. How well would privacy have been protected?
- Research:
  - All 4 privacy policies for QF mining, QR mining, QU-pair mining
  - Interest: Impact on the full logs
  - Interest: Impact on a single AOL user
  - Parameters: k, largest d that k allows, …

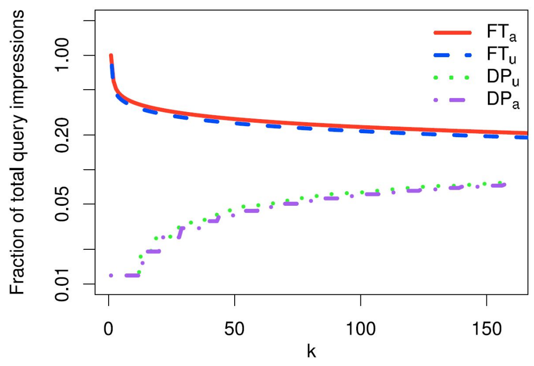
Simulations

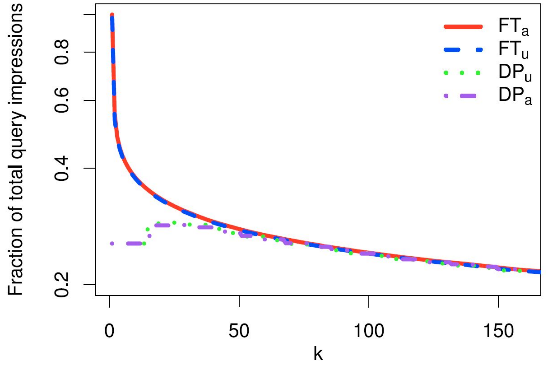
Results

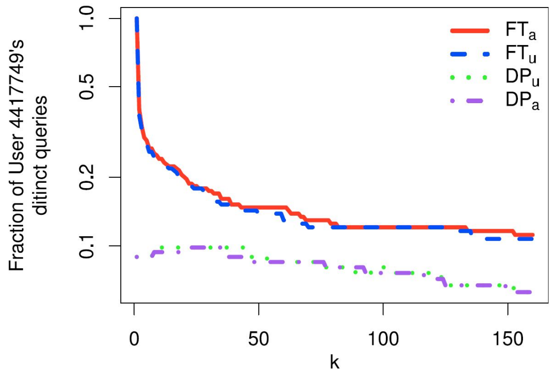
- Interest: How many queries get released as anonymity k increases?
  - Higher k means higher threshold specific to policy
    - So.. Higher k means less identifying queries!
  - For AOL? For MSN? For 1 AOL user?
- Query frequency mining
  - Both for fractions of total queries and for fractions distinct queries
  - Most extensive results
  - Quite resembling for the other two
- Query reformulation mining
  - Mostly numbers and conclusion
- Query-URL frequency mining
  - Mostly numbers and conclusion
- Conclusions:
  - …

Simulations

Query frequency mining







- Differential privacy great on AOL
  - d < k
- MSN logs give different results
  - d > queries in session
- What about distinct queries?
  - Constant as k increases

- User 4417749: 224/239 queries distinct
- Non-differential scores relatively bad
  - k>1: Relative names
  - k>2: Location information
  - k>20: Possibly sensitive (recall anonymization!)
  - k>7500: Most common
- Differential methods are conservative
  - 1<=k<=7: Both policies allow 1 query per user (d=1)
  - k increases: only most common left
  - Penalty: Only 10% is revealed

Simulations

Query reformulation mining (EXTRA)

- AOL: 18.7M unique pairs, 20.4M instances
- MSN: 5.1M unique pairs, 5.5M instances
- Same trends and graphs
  - Major difference: Smaller scale for all k>1
  - Lower privacy requirements to preserve privacy
- User 4417749
  - Never entered same reformulation twice
  - k = 2: Only 6 released under FTu
    - Lonelyness -> Loneliness (recall anonymity)

Simulations

Query-URL frequency mining (EXTRA)

- Interest: How often is a URL clicked following a query?
- AOL: 9.2M pairs total, 5.4M unique
- MSN: 2.4M unique pairs, 1.6M instances
- On average both logs: Just under 2 associated URLs per query
- Same trends and graphs
  - Moderate difference: Bit of a smaller scale
  - Makes sense, only limited URLs possible vs endless reformulations
- If k low enough: Ranking suitable with non-differential PPs
- User 4417749
  - Similar to query frequency
  - Mostly navigational (e.g. search term “Google”)
    - Get released
  - Some informational (e.g. search term “Sinus infection”)
    - Not released

Simulations

Conclusions

- Interest: How many queries get released as anonymity k increases?
  - Higher k means higher threshold specific to policy
    - So.. Higher k means less identifying queries!
  - For AOL? For MSN? For 1 AOL user?
- Query frequency mining
  - Both for fractions of total queries and for fractions distinct queries
  - Most extensive results
  - Quite resembling for the other two
- Query reformulation mining
  - Mostly numbers and conclusion
- Query-URL frequency mining
  - Mostly numbers and conclusion
- Conclusions:
  - Privacy policies with stronger guarantees have larger utility costs
    - Some tasks (like QR mining) require larger logs for differential policies
  - Differential privacy provides better privacy guarantees for low k
  - Privacy of user 4417749 would have been protected under any policy
    - But comes with a tradeoff for researchers

Wrapping up

- Collecting, storing and mining of search logs
    - Distributed, private, anonymous
- Motivation: Search logs available to researchers, while maintaining user privacy
    - Researchers: Have access to up-to-date data
    - Users: Have control over their data
- System is versatile: Different privacy policies and application possible
- Simulations show:
    - Tradeoff: Privacy enhancement and control vs. ease of use and research data
    - Much can be revealed, but dependent on application, privacy policy, and k
    - Can work in practice (pilot user study)

Conclusions

Privacy Friendly Web Search

Radboud University

Conclusions

- Importance
  - Fundamental rights
  - Control over:
    - Confidentiality
    - Anonymity
    - Personal security
- Frontend methods: Query Obfuscation
- Backend framework: Distributed Protocols
- Combination: CrowdLogging