

Releasing Search Queries and Clicks Privately

Aleksandra Korolova*
 Computer Science Department
 Stanford University
 Stanford, CA
 korolova@cs.stanford.edu

Krishnaram Kenthapadi, Nina Mishra, Alexandros Ntoulas
 Search Labs
 Microsoft Research
 Mountain View, CA
 {krisken, ninam, antoulas}@microsoft.com

ABSTRACT

The question of how to publish an anonymized search log was brought to the forefront by a well-intentioned, but privacy-unaware AOL search log release. Since then a series of ad-hoc techniques have been proposed in the literature, though none are known to be provably private. In this paper, we take a major step towards a solution: we show how queries, clicks and their associated perturbed counts can be published in a manner that rigorously preserves privacy. Our algorithm is decidedly simple to state, but non-trivial to analyze. On the opposite side of privacy is the question of whether the data we can safely publish is of any use. Our findings offer a glimmer of hope: we demonstrate that a non-negligible fraction of queries and clicks can indeed be safely published via a collection of experiments on a real search log. In addition, we select an application, keyword generation, and show that the keyword suggestions generated from the perturbed data resemble those generated from the original data.

Categories and Subject Descriptors: H.2.0 [Database Management]: Security, integrity, and protection; H.3 [Information Storage and Retrieval]; G.2.3 [Mathematics of Computing]; K.4 [Computers and Society];

General Terms: Algorithms, Experimentation, Human Factors, Legal Aspects, Measurement, Performance, Security, Theory

1. INTRODUCTION

Web search logs collect queries and clicks of users as they interact with a search engine. These logs have been successfully used by search engines in order to improve the quality of search results: for example, to fix spelling errors, suggest related searches, expand acronyms and estimate query popularity over time.

The release of these logs to the greater public would be a great benefit to society. Computer science researchers have been building a case for search log access [4, 25] so that they could study and analyze new IR algorithms via a common benchmark search log, as well as learn about user information needs and query formulation approaches. Social scientists could investigate the use of language in queries as well as discrepancies between user interests as revealed by their queries versus as revealed by face-to-face surveys [24]. Advertisers could use the logs to understand how users navigate to their pages, gain a better understanding of their competitors, and improve keyword advertising campaigns.

*Work done while interning at Microsoft Research.

On the other hand, the release of these logs to the greater public would be catastrophic from a privacy perspective. Users communicate with a search engine in an uninhibited manner, leaving behind an electronic trail of confidential thoughts and painfully identifiable information. For example, users search for their own name [17] or the names of their friends, home address, their health concerns, as well as names of their family and friends. Users even enter their credit card number and social security number just to find out what information is present on the web. It would be irresponsible for a search engine company to release this data without modification.

The open question to date is if there even exists a way to publish search logs in a perturbed fashion in a manner that is simultaneously useful and private. At first blush, the problem seems deceptively easy: why not just replace usernames with random identifiers? This simplistic view led to an AOL data release in 2006 in which the searches of an innocent citizen were quickly identified by a newspaper journalist [5]. As a consequence of releasing this private data set the CTO of AOL resigned, two employees were fired, a class action lawsuit is pending, and the event lives on as one of the “101 Dumbest Moments in Business” [14].

Further ad-hoc techniques have been explored in the literature. Kumar et al [19] consider tokenizing each search query and securely hashing the token into an identifier. Hashing gives only the appearance of privacy and, indeed, the authors show that hashes can be inverted by using token frequencies and other search logs.

An important lesson from decades of research in cryptography is that ad-hoc methods do not work. At issue is the fact that a data releaser does not know a priori what information an attacker will use in concert with the released data to deduce more private information. In the case of token-based hashing, prior search log releases were used. In the case of NetFlix [22], anonymized user ratings were combined with IMDb ratings to infer individuals.

1.1 Contributions

In this paper, we take a first significant step towards a solution. Rather than producing a search log, we consider the problem of releasing a query click graph. Rather than relying on intuition for showing that our approach provides privacy, we utilize a formal definition due to Dwork et al [11] that allows for an attacker with arbitrary prior knowledge, and design an algorithm for releasing a query-click graph that provably satisfies that definition.

Utility: We propose to publish a query click graph where the vertices correspond to both queries and URLs and there is an edge from a query to a URL with weight equal to the number of users who click on that URL given they posed the query. Each query node is labeled by the number of times this query was posed in the log. Similarly, there is an edge from one query to another query with weight equal to the number of users who posed one query and reformulated to another.

The query click graph is the basis for many uses of the search log [8, 3]. Query suggestions can be derived using common query reformulations. Spelling corrections can be inferred from queries with low click through and high reformulation rates. Similar queries can be found using common URLs clicked for those queries. Query classification and keyword generation can also be deduced from the query click graph [13].

Privacy: From the privacy side, we adapt the differential privacy definition [11]. In a nutshell, the definition states that upon seeing a published data set an attacker should gain little knowledge about any specific individual.

The algorithm for producing a Private Query Click graph is quite simple and can be intuitively described as “throw away tail queries”:

Queries: To determine which queries to publish, if the frequency of the query plus some noise exceeds some threshold, we publish the query, otherwise we do not. We also publish the number of times the query was posed plus noise.

URLs: Given the queries that are safe to publish, the ten URLs surfaced are also safe to publish because anyone can pose a query to a search engine and see the top ten links. To publish the number of users who click on a result, we compute the actual number and add noise.

Our privacy proofs demonstrate that each of these steps individually preserves privacy and further that the composition of the steps also preserves privacy. In addition, we precisely characterize the trade-off between privacy and threshold used for publishing a query: the more stringent the privacy requirement, the higher the threshold, and consequently the fewer the number of queries that can be safely published.

Experiments: Given the privacy theorems, we next consider whether what we can publish from a real search log is of any use. We show that the fraction of distinct queries that can be published, as well as the amount of search volume involving those queries, is non-negligible. We then select two applications, keyword generation and studying human fears, and demonstrate that keywords and fears obtained from our perturbed logs closely resemble those obtained from the original unperturbed data. The experiments are an indication that it may be possible, if only in a limited sense, to bridge the gap between privacy and utility.

2. RELATED WORK

We describe work related to both search log anonymization and differential privacy.

2.1 Search Log Anonymization

The AOL search log release sparked interest in the problem of search log anonymization. In that data set, usernames were masked with random identifiers [2] and, in a matter of days, a New York Times reporter identified Thelma Arnold, a 62-year old widow from Lilburn, GA as user #4417749 [5], and her queries ranging from landscapers in her town to diseases of her friends.

Following AOL’s release, many other ad-hoc techniques have been proposed. For example, if removing usernames is not enough, then it is natural to wonder if removing session IDs preserves privacy. It turns out that such an approach fails since one can design user models of approximate time that passes in between queries to stitch together sessions via the timestamps. Beyond inferring sessions, the heart of the problem lies in the fact that revealing a single query such as a credit card number breaks privacy.

If the queries themselves are private, then it is natural to wonder if hashing the queries preserves privacy. In fact, that too fails as Kumar et al [19] nicely argue. They show that tokenizing a query, hashing the tokens and publishing the hashes does not preserve pri-

vacy since an adversary who has access to another log can reverse-engineer the tokens by utilizing the frequency with which the query appears in the log.

Jones et al [16] study an application of simple classifiers to connect a sequence of queries to the location, gender and age of the user issuing the queries, and argue that releasing a query log poses a privacy threat because these three characteristics of the user can be used to create a set of candidate users who might have posed that query. Their more recent work [17] investigates privacy leaks that are possible even when queries from multiple users are grouped together and no user or session identifiers are released.

In short, while many papers describe ad-hoc techniques [25, 1] the results are, by and large, negative for privacy. Thus, the question of how to anonymize a search log remains open.

Our work focuses on a seemingly more attainable goal of releasing a private query click graph. While this graph is not as powerful as the actual search log, many computations can still be performed on the click graph with results similar to the actual search log, e.g., finding similar queries, keyword generation, and performing spell corrections.

2.2 Differential Privacy

If there is any lesson to be drawn from decades of cryptography and the many privacy compromises in published datasets, it is that ad-hoc techniques do not work [9]. The way to achieve privacy is to start with a rigorous definition and design an algorithm that satisfies the privacy definition.

Our paper is the first to take a concrete definition of privacy, differential privacy, and design an algorithm for producing a private query click graph that provably satisfies that definition. We choose differential privacy because it does not stipulate the prior knowledge of the attacker: regardless of what the attacker knows, releasing data that satisfies the differential privacy definition will not substantially increase the attacker’s chance of inferring something about an individual. A formal definition appears in Section 3.

Many algorithms exist for publishing data in a differentially private manner. The seminal work of Dwork et al [11] shows that any function with low sensitivity can be computed privately. A function has *low sensitivity* if the addition or removal of one person can only change the outcome of the function evaluation by a small amount. We use the ideas developed in [11] for our analysis. The aggregate statistical values reflected in our query click graph have low sensitivity, provided that each user issues a bounded number of queries.

Randomly sampling a data set is known to be differentially private [7], but only under the assumption that there are few rare values. In the web context, it is more likely that every person’s searches are a unique fingerprint to them; thus, randomly sampling the search log breaks privacy.

Prior to our work, no differentially private algorithm was known for efficiently publishing a query click graph. A mechanism due to McSherry and Talwar [20] and Blum et al [6] could be adapted to this task in theory but is not feasible in practice, as this mechanism takes time exponential in the size of the output space. More recently, McSherry and Talwar have proposed an algorithm for releasing synthetic queries [21] that holds promise for synthetic data releases.

3. PRIVACY DEFINITION

In this section we describe the differential privacy definition that was first conceived by Dwork et al [11]. Intuitively, a data releaser preserves privacy if no attacker gains significant knowledge about an individual’s private searches beyond what the attacker could

have learned from a similar neighboring search log in which that individual's searches are modified or removed. To formalize this intuition, we require that for all pairs of neighboring search logs that differ in one user's searches and all possible published query click graphs, the probability that any subset of query click graphs is published from one log is within an e^ϵ multiplicative factor of the probability that that subset is published from the neighboring log, plus an additive δ component.

Definition 1. A randomized algorithm A is (ϵ, δ) -differentially private if for all data sets D_1 and D_2 differing in at most one user and all $\hat{D} \subseteq \text{Range}(A) : \Pr[A(D_1) \in \hat{D}] \leq e^\epsilon \cdot \Pr[A(D_2) \in \hat{D}] + \delta$.

Differential privacy captures the desired notion of privacy that the risk to one's privacy should not substantially increase as a result of participating in the dataset (e.g. as a result of using the search engine), by guaranteeing that any given disclosure is almost as likely whether or not the individual participates in the dataset. Differential privacy also does not make any assumptions about adversary's computational power or ability to access additional data beyond the release. Although differential privacy is not an absolute privacy guarantee, in the setting when we have to trade-off the benefits of the data release with user privacy, differential privacy ensures that the amount of additional privacy risks incurred by users is limited.

In the first definition of differential privacy [11], $\delta = 0$. Subsequent work [10] relaxed the definition to include a non-zero additive component.

There are no hard and fast rules for setting ϵ and δ – it is generally left to the data releaser. One consideration to take into account when choosing δ is the number of users participating in the dataset. Indeed, imagine that every person's every query is private – e.g. the log consists of searches for names and social security numbers. Then $\delta > \frac{1}{\text{number of users}}$ suggests that at least one person's privacy is compromised. Of course, imagining that a search log consists of only sensitive queries is a very paranoid view of privacy but nonetheless, the magnitude of $\frac{1}{\text{number of users}}$ is useful to keep in mind when choosing δ .

4. ALGORITHM FOR RELEASING SEARCH QUERIES AND CLICKS

We next describe our algorithm for generating a private query click graph. The key components are: determining which queries to publish, together with the number of times the query was posed and, further, determining which URLs to publish, together with the number of times the URL was clicked for each query. Our basic method for accomplishing these tasks utilizes a noisy count: for any statistic x of the data, the *noisy count* of x is $x + \text{Lap}(b)$, where $\text{Lap}(b)$ denotes a random variable drawn independently from the Laplace distribution with mean zero and scale parameter b .

At a high-level, the algorithm proceeds as follows.

1. **Limit User Activity:** Keep only the first d queries posed by each user and first d_c URL clicks of each user (Line 2 of Algorithm 1).
2. **Queries:** If the noisy count of the number of times a query is posed exceeds a specified threshold, output the query together with its noisy count (Lines 3-5 of Algorithm 1).
3. **URLs:** If a query is safe to publish, then the ten URLs that are surfaced for that query are also safe to publish since anyone can pose the query to a search engine and see the ten results. For each query and ten surfaced URLs for that query,

we output the noisy count of the number of times each URL was clicked for that query (Line 6 of Algorithm 1).

Some comments about the algorithm are in order.

- (1) We limit user activity in order to preserve privacy: if a user can contribute an unbounded number of queries and clicks then they can have an unlimited influence on the set of queries that are published (and the URLs that are clicked). In practice, a typical user does not pose an unlimited number of queries anyway — studies suggest that an average user poses about 34 queries per month [12].
- (2) While the main idea of throwing away tail queries is quite natural and has been previously suggested in the literature [1], this paper is the first to mathematically quantify exactly how to perform this operation so as to preserve privacy with respect to a rigorous privacy definition. Indeed, our theorems in subsequent sections establish a direct connection between the threshold used for defining a tail query and the resulting privacy guarantees.
- (3) Because our algorithm does not release any fake queries (i.e. the queries it releases are a subset of the queries present in the original log), it won't satisfy differential privacy with $\delta = 0$. For example, in the event that a user poses a search query q in log D_1 that does not appear in the neighboring log D_2 , then there's a non-zero probability that q is published when starting from log D_1 and a zero probability that q is published when starting from log D_2 . Hence the former probability cannot be upper bounded by the latter probability without the extra help of a non-zero additive component δ .
- (4) It is crucial to note that every time our algorithm computes the noisy count, the noise should be generated independently from the Laplace distribution.

Algorithm 1 Release-Data

- 1: **Input:** D - search log, d, d_c - parameters that limit user activity, b, b_q, b_c - noise parameters, K - threshold that defines tail.
 - 2: **Limit-User-Activity:** $D \leftarrow$ Keep the first d queries and the first d_c URL clicks of each user.
 - 3: For each query q , let $M(q, D) =$ number of times q appears in D
 - 4: **Select-Queries:** $Q \leftarrow \{q : M(q, D) + \text{Lap}(b) > K\}$
 - 5: **Get-Query-Counts:** For each q in Q , output $\langle q, M(q, D) + \text{Lap}(b_q) \rangle$
 - 6: **Get-Click-Counts:** For each URL u in the top ten results for $q \in Q$, output $\langle q, u, \text{number of times } u \text{ was clicked when } q \text{ was posed} + \text{Lap}(b_c) \rangle$.
-

5. PRIVACY GUARANTEES

We now state formally the (ϵ, δ) -differential privacy guarantees that our algorithm provides. We then provide a sketch of the proof that each of the individual steps preserves privacy and, further, that their composition preserves privacy.

Let K, d, d_c, b, b_q, b_c be the parameters of Algorithm 1 such that $K \geq d$. Define $\alpha = \max(e^{1/b}, 1 + \frac{1}{2e^{(K-1)/b-1}})$ and the multiplicative and additive privacy parameters as $\epsilon_{alg} = d \cdot \ln(\alpha) + d/b_q + d_c/b_c$ and $\delta_{alg} = \frac{d}{2} \exp(\frac{d-K}{b})$.

THEOREM 1. *Algorithm 1 is $(\epsilon_{alg}, \delta_{alg})$ -differentially private for every pair of search logs differing in one user, where ϵ_{alg} and δ_{alg} are defined as above.*

5.1 Proof Overview

In order to prove Theorem 1, we will show that each step of the algorithm is differentially private for appropriate values of ϵ and δ and that their composition is also differentially private.

Lemma 1. **Select-Queries** is $(d \cdot \ln(\alpha), \delta_{alg})$ -differentially private if each user is limited to posing at most d queries, where α and δ_{alg} are defined as above.

Lemma 2. **Get-Query-Counts** is $(d/b_q, 0)$ -differentially private.

Lemma 3. **Get-Click-Counts** is $(d_c/b_c, 0)$ -differentially private.

Lemma 4. Suppose **Select-Queries** is (ϵ_1, δ) -differentially private, **Get-Query-Counts** is $(\epsilon_2, 0)$ -differentially private, and **Get-Click-Counts** is $(\epsilon_3, 0)$ -differentially private. Then Algorithm 1 is $(\epsilon_1 + \epsilon_2 + \epsilon_3, \delta)$ -differentially private.

Theorem 1 follows from Lemmas 1, 2, 3, and 4. We next sketch the key ideas in the proofs of the above lemmas.

5.2 Privacy for Selecting Queries

We first prove the guarantees for **Select-Queries** when each user can pose at most one query (Lemma 5) and then generalize the proof to hold for d queries per user to obtain Lemma 1.

Privacy for Select-Queries with $d = 1$:

Let $\alpha = \max(e^{1/b}, 1 + \frac{1}{2e^{(K-1)/b-1}})$ and $\delta_1 = \frac{1}{2}e^{\frac{1-K}{b}}$. Denote the Select-Queries algorithm by A .

Lemma 5. If each user can pose at most one query and $K \geq 1$, then Select-Queries satisfies $(\ln(\alpha), \delta_1)$ -differential privacy.

PROOF. Let D_1 and D_2 be arbitrary search logs that differ in exactly one query q_* posed by some user such that D_2 is the larger of the two logs. Let $\hat{D} \subseteq \text{Range}(A)$ denote an arbitrary set of possible outputs. Then we need to show the following:

$$\Pr[A(D_1) \in \hat{D}] \leq \alpha \Pr[A(D_2) \in \hat{D}] + \delta_1 \quad (1)$$

$$\Pr[A(D_2) \in \hat{D}] \leq \alpha \Pr[A(D_1) \in \hat{D}] + \delta_1 \quad (2)$$

We consider two different scenarios depending on whether q_* already appears as a query of some user in D_1 or it is a new query. For the first case, we do not need the additive parameter (i.e., $\delta_1 = 0$ works) as we can bound the ratio of the probabilities in each expression above. The key idea is to notice that q_* has a slightly higher probability of being released from D_2 compared to D_1 and to argue that the ratio of these probabilities can be bounded. However, for the second case, q_* can never be released from D_1 , and hence we cannot bound the ratio of the probabilities. Instead, we make use of the fact that q_* occurs only once in D_2 and use the additive parameter δ_1 to bound the probability that its noisy count exceeds the threshold K .

Recall that our algorithm A only produces a subset of queries contained in D_1 (or D_2). Hence, any set of queries O that contains some query not present in D_1 or D_2 need not be considered as part of \hat{D} in our analysis, as the probability that A produces O is zero, with D_1 or D_2 as input. We also partition \hat{D} into two subsets: \hat{D}^+ , the query sets in \hat{D} that contain q_* and \hat{D}^- , the query sets in \hat{D} that do not contain q_* .

Throughout the proof, we will utilize Observations 2, 3 and 4 described in the Appendix, which are useful tools for dealing with properties of ratios and of the Laplace distribution.

Case 1: $q_* \in D_1$

Let $M(q, D)$ be the number of times query q appears in the search log D . Then, $M(q_*, D_1) \geq 1$, $M(q_*, D_2) = M(q_*, D_1) + 1$.

We first prove inequality (1) with $\alpha = e^{1/b}$ and $\delta_1 = 0$ by upper bounding the ratio $\frac{\Pr[A(D_1) \in \hat{D}]}{\Pr[A(D_2) \in \hat{D}]}$, which we denote by R_2^1 .

From our partition of \hat{D} into \hat{D}^+ and \hat{D}^- and using observation 4,¹ we have:

$$R_2^1 = \frac{\Pr[A(D_1) \in \hat{D}]}{\Pr[A(D_2) \in \hat{D}]} = \frac{\Pr[A(D_1) \in \hat{D}^+] + \Pr[A(D_1) \in \hat{D}^-]}{\Pr[A(D_2) \in \hat{D}^+] + \Pr[A(D_2) \in \hat{D}^-]} \\ \leq \max\left(\frac{\Pr[A(D_1) \in \hat{D}^+]}{\Pr[A(D_2) \in \hat{D}^+]}, \frac{\Pr[A(D_1) \in \hat{D}^-]}{\Pr[A(D_2) \in \hat{D}^-]}\right)$$

Consider now the ratio $\frac{\Pr[A(D_1) \in \hat{D}^+]}{\Pr[A(D_2) \in \hat{D}^+]}$. Recall that in our algorithm, the decision to release a particular query is made independently for each query and that D_1 and D_2 differ only in the number of times that q_* occurs in each of them. Hence, for a particular possible output O , s.t. $q_* \in O$: $\frac{\Pr[A(D_1)=O]}{\Pr[A(D_2)=O]} = \frac{\Pr[q_* \text{ released by } A(D_1)]}{\Pr[q_* \text{ released by } A(D_2)]}$.

Generalizing this observation to all outputs $O_i \in \hat{D}^+$ we obtain: $\frac{\Pr[A(D_1) \in \hat{D}^+]}{\Pr[A(D_2) \in \hat{D}^+]} = \frac{\sum_{O \in \hat{D}^+} \Pr[A(D_1)=O]}{\sum_{O \in \hat{D}^+} \Pr[A(D_2)=O]} = \frac{\Pr[q_* \text{ released by } A(D_1)]}{\Pr[q_* \text{ released by } A(D_2)]} = \frac{\Pr[M(q_*, D_1) + \text{Lap}(b) > K]}{\Pr[M(q_*, D_2) + \text{Lap}(b) > K]}$

By analogous reasoning with respect to \hat{D}^- we obtain:

$$\frac{\Pr[A(D_1) \in \hat{D}^-]}{\Pr[A(D_2) \in \hat{D}^-]} = \frac{\Pr[M(q_*, D_1) + \text{Lap}(b) < K]}{\Pr[M(q_*, D_2) + \text{Lap}(b) < K]} = \frac{\Pr[M(q_*, D_1) + \text{Lap}(b) < K]}{\Pr[M(q_*, D_1) + 1 + \text{Lap}(b) < K]}$$

From these two bounds on the ratios, we bound R_2^1 :

$$R_2^1 \leq \max\left(\frac{\Pr[M(q_*, D_1) + \text{Lap}(b) > K]}{\Pr[M(q_*, D_1) + 1 + \text{Lap}(b) > K]}, \frac{\Pr[M(q_*, D_1) + \text{Lap}(b) < K]}{\Pr[M(q_*, D_1) + 1 + \text{Lap}(b) < K]}\right),$$

which by Observation 3 implies that

$$R_2^1 = \frac{\Pr[A(D_1) \in \hat{D}]}{\Pr[A(D_2) \in \hat{D}]} \leq \max(1, e^{1/b}) = e^{1/b} \quad (3)$$

proving inequality (1), as desired.

A similar analysis can be performed to show that

$$\frac{\Pr[A(D_2) \in \hat{D}]}{\Pr[A(D_1) \in \hat{D}]} \leq e^{1/b}, \quad (4)$$

yielding the proof of inequality (2) with $\alpha = e^{1/b}$ and $\delta_1 = 0$.

Case 2: $q_* \notin D_1, q_* \in D_2$

We now proceed to prove inequality (1) with $\alpha = \frac{1}{1 - 0.5 \exp(\frac{1-K}{b})}$ and $\delta_1 = 0$.

First consider outputs that do not contain the new query q_* . By similar reasoning as in Case 1, the probability of obtaining an output O , where $O \in \hat{D}^-$ when starting from the D_2 log differs from the probability of obtaining an output O , when starting from D_1 only in the choice that the algorithm has to make for query q_* . Hence, $\Pr[A(D_2) \in \hat{D}^-] = \Pr[q_* \text{ was not released by } A(D_2)] \cdot \Pr[A(D_1) \in \hat{D}^-]$ and

$$\frac{\Pr[A(D_1) \in \hat{D}^-]}{\Pr[A(D_2) \in \hat{D}^-]} = \frac{1}{\Pr[q_* \notin A(D_2)]} \quad (5)$$

$$= \frac{1}{\Pr[1 + \text{Lap}(b) < K]} = \frac{1}{1 - 0.5 \exp(\frac{1-K}{b})} \quad (6)$$

Since query q_* is not present in D_1 , our algorithm would not produce any output containing q_* when given D_1 as input, and so $\Pr[A(D_1) \in \hat{D}^+] = 0$.

Using the partition of \hat{D} into \hat{D}^+ and \hat{D}^- we have:

$$\frac{\Pr[A(D_1) \in \hat{D}]}{\Pr[A(D_2) \in \hat{D}]} = \frac{\Pr[A(D_1) \in \hat{D}^-]}{\Pr[A(D_2) \in \hat{D}^+] + \Pr[A(D_2) \in \hat{D}^-]} \\ \leq \frac{\Pr[A(D_1) \in \hat{D}^-]}{\Pr[A(D_2) \in \hat{D}^-]} \leq \frac{1}{1 - 0.5 \exp(\frac{1-K}{b})} \quad (7)$$

proving inequality (1), as desired.

¹Observation 4 holds only for positive denominators. In the event that either or both the denominators are zero, it can be shown that the differential privacy bound continues to hold.

It remains to show that in this case, inequality (2) is satisfied with $\alpha = 1 - 0.5 \exp(\frac{1-K}{b})$ and $\delta_1 = 0.5 \exp(\frac{1-K}{b})$.

Observe that

$$\begin{aligned} Pr[A(D_2) \in \hat{D}^+] &\leq Pr[q_* \text{ was released}] = \\ &= Pr[M(q_*, D_2) + Lap(b_1) > K] = 0.5 \exp(\frac{1-K}{b}) \quad (8) \end{aligned}$$

Combining (8) and (6), we have:

$$\begin{aligned} \frac{Pr[A(D_2) \in \hat{D}]}{Pr[A(D_1) \in \hat{D}]} &= \frac{Pr[A(D_2) \in \hat{D}^+] + Pr[A(D_2) \in \hat{D}^-]}{Pr[A(D_1) \in \hat{D}^-]} = \frac{Pr[A(D_2) \in \hat{D}^-]}{Pr[A(D_1) \in \hat{D}^-]} + \\ \frac{Pr[A(D_2) \in \hat{D}^+]}{Pr[A(D_1) \in \hat{D}^-]} &\leq 1 - 0.5 \exp(\frac{1-K}{b}) + \frac{0.5 \exp(\frac{1-K}{b})}{Pr[A(D_1) \in \hat{D}^-]}, \end{aligned}$$

which completes the proof of inequality (2).

Thus from the two cases, depending on whether q_* is an additional occurrence of an element already present in D_1 or it is an entirely new element to D_1 , we established that our algorithm satisfies the $(\ln(\alpha), \delta_1)$ -differential privacy, where

$$\begin{aligned} \alpha &= \max\left(e^{1/b}, 1 - 0.5 \exp(\frac{1-K}{b}), \frac{1}{1 - 0.5 \exp(\frac{1-K}{b})}\right) = \\ &= \max\left(e^{1/b}, \frac{1}{1 - 0.5 \exp(\frac{1-K}{b})}\right), \text{ and } \delta_1 = \frac{1}{2} e^{(\frac{1-K}{b})}. \quad \square \end{aligned}$$

Observation 1. [Need for δ_1] A crucial observation made in the proof of this lemma is that the necessity for δ_1 arises only when the extra query in D_2 is a query that was not previously present in D_1 , and we are attempting to upper bound the ratio of $\frac{Pr[A(D_2) \in \hat{D}]}{Pr[A(D_1) \in \hat{D}]}$.

We will use this observation next as we generalize the proof of privacy guarantees to the case where each user can pose at most d queries.

Privacy for Select-Queries for arbitrary d :

We next prove Lemma 1. We first show that straight-forward generalization does not work and hence perform a tighter analysis.

Straight-forward generalization: A natural approach towards this proof is to observe that a search log D_2 that differs from D_1 by at most d queries can be obtained from D_1 by adding the d queries to it, one query at a time. This enables the repeated application of the results of Lemma 5 to obtain:

$$\begin{aligned} Pr[A(D_1) \in \hat{D}] &\leq \alpha Pr[A(D_1 + q_1) \in \hat{D}] + \delta_1 \\ &\leq \alpha(\alpha Pr[A(D_1 + q_1 + q_2) \in \hat{D}] + \delta_1) + \delta_1 \leq \dots \\ &\leq \alpha^d Pr[A(D_2) \in \hat{D}] + \delta_1 \frac{\alpha^d - 1}{\alpha - 1} \end{aligned}$$

However, this approach yields $\delta_{alg} = \delta_1 \frac{\alpha^d - 1}{\alpha - 1}$, which will quickly exceed 1, yielding meaningless privacy guarantees. To avoid the blow-up in the additive component of privacy guarantees, we build on the insights of the Proof of Lemma 5, and especially, on Observation 1 in order to show better guarantees for δ_{alg} .

Tighter analysis for δ_{alg} :

As in the proof of Lemma 5, let D_2 be the larger of the two search logs, containing an additional d queries compared to D_1 . Denote by x_1, \dots, x_{n_x} those of the additional d queries that are already in D_1 and by y_1, \dots, y_{n_y} - those queries that are unique to D_2 . Note that $\sum_{i=1}^{n_x} (M(x_i, D_2) - M(x_i, D_1)) + \sum_{i=1}^{n_y} M(y_i, D_2) \leq d$ and $n_x + n_y \leq d$

We also split the elements of \hat{D} into two subsets as follows: denote by \hat{D}^- the set of elements of \hat{D} which can be obtained from both D_1 and D_2 , and by \hat{D}^+ - the set of elements of \hat{D} which can only be obtained from D_2 (as before, we remove those elements of \hat{D} that cannot be obtained from either D_1 or D_2 from consideration wlog).

Observe that in the proof of Lemma 5, the additive component δ arose only when considering the ratio $\frac{Pr[A(D_2) \in \hat{D}]}{Pr[A(D_1) \in \hat{D}]}$ and not when

considering the ratio $\frac{Pr[A(D_1) \in \hat{D}]}{Pr[A(D_2) \in \hat{D}]}$. We take advantage of this observation by proving the necessary upper bounds by recursively applying Lemma 5 in one case, and by performing a more careful analysis for the need for the additive component in the other case.

Proof of (1) with $\alpha_d = \alpha^d, \delta_{alg} = 0$:

We first observe that $\frac{Pr[A(D_1) \in \hat{D}]}{Pr[A(D_2) \in \hat{D}]}$ can be represented as a product of ratios of obtaining an output in \hat{D}^- when starting from datasets differing in one element as follows:

$$\begin{aligned} \frac{Pr[A(D_1) \in \hat{D}]}{Pr[A(D_2) \in \hat{D}]} &= \frac{Pr[A(D_1) \in \hat{D}^-]}{Pr[A(D_2) \in \hat{D}^+] + Pr[A(D_2) \in \hat{D}^-]} \leq \frac{Pr[A(D_1) \in \hat{D}^-]}{Pr[A(D_2) \in \hat{D}^-]} = \\ &= \frac{Pr[A(D_1) \in \hat{D}^-]}{Pr[A(D_1 + x_1) \in \hat{D}^-]} \cdot \frac{Pr[A(D_1 + x_1) \in \hat{D}^-]}{Pr[A(D_1 + x_1 + x_2) \in \hat{D}^-]} \cdot \dots \cdot \\ &= \frac{Pr[A(D_1 + x_1 + \dots + x_{n_x} + y_1 + \dots + y_{n_y}) \in \hat{D}^-]}{Pr[A(D_1 + x_1 + \dots + x_{n_x} + y_1 + \dots + y_{n_y}) \in \hat{D}^-]} \end{aligned}$$

Applying the above decomposition of the ratio into a product of ratios² and the results of intermediate steps (3) and (6) of Lemma 5 to each of the ratios in the product, we obtain: $\frac{Pr[A(D_1) \in \hat{D}]}{Pr[A(D_2) \in \hat{D}]} \leq \alpha^d$, as desired.

Proof of (2) with $\alpha_d = \alpha^d, \delta_{alg} = \frac{d}{2} \exp(\frac{d-K}{b})$:

By definition of \hat{D}^+ one can obtain an output in \hat{D}^+ when given the search log D_2 , only if at least one of the queries in D_2 which was not present in D_1 is chosen for release. Applying the union bound we have:

$$\begin{aligned} Pr[A(D_2) \in \hat{D}^+] &\leq \sum_{i=1}^{n_y} Pr[n_y \text{ was chosen for release}] = \\ &= \sum_{i=1}^{n_y} Pr[M(y_i, D_2) + Lap(b) > K] \leq \\ &= \sum_{i=1}^{n_y} Pr[Lap(b) > K - M(y_i, D_2)] = \frac{1}{2} \sum_{i=1}^{n_y} \exp\left(\frac{M(y_i, D_2) - K}{b}\right) \\ &\text{(applying the knowledge that } n_y \leq d \text{ and } M(y_i, D_2) \leq d) \\ &\leq \frac{d}{2} \exp\left(\frac{d-K}{b}\right) \quad (9) \end{aligned}$$

Decomposing the ratio of outputs being in \hat{D}^- as a product³ of obtaining \hat{D}^- when starting with dataset differing in one element we have:

$$\begin{aligned} \frac{Pr[A(D_2) \in \hat{D}^-]}{Pr[A(D_1) \in \hat{D}^-]} &= \frac{Pr[A(D_1 + x_1) \in \hat{D}^-]}{Pr[A(D_1) \in \hat{D}^-]} \cdot \frac{Pr[A(D_1 + x_1 + x_2) \in \hat{D}^-]}{Pr[A(D_1 + x_1) \in \hat{D}^-]} \cdot \dots \cdot \\ &= \frac{Pr[A(D_1 + x_1 + \dots + x_{n_x} + y_1 + \dots + y_{n_y}) \in \hat{D}^-]}{Pr[A(D_1 + x_1 + \dots + x_{n_x} + y_1 + \dots + y_{n_y}) \in \hat{D}^-]} \leq \end{aligned}$$

(applying the result of (4) to those datasets differing in a query already present and the result of (5) to those datasets differing in a query not yet present)

$$\begin{aligned} &\leq \prod_{i=1}^{n_x} e^{1/b} \cdot \prod_{i=1}^{n_y} \max(e^{1/b}, Pr[M(y_i, D_2) + Lap(b) < K]) \\ &\text{(using the value of } M(y_i, D_2) \text{ maximizing the probability term for each } i) \leq e^{n_x/b} \cdot \left(\max(e^{1/b}, Pr[1 + Lap(b) < K])\right)^{n_y} = \\ &= e^{n_x/b} \cdot \left(\max(e^{1/b}, 1 - 0.5 \exp(\frac{1-K}{b}))\right)^{n_y} \leq e^{\frac{n_x + n_y}{b}} \leq \alpha^d \end{aligned}$$

We now use (9) and the inequality above to obtain the desired upper bound:

$$\begin{aligned} \frac{Pr[A(D_2) \in \hat{D}]}{Pr[A(D_1) \in \hat{D}]} &= \frac{Pr[A(D_2) \in \hat{D}^-] + Pr[A(D_2) \in \hat{D}^+]}{Pr[A(D_1) \in \hat{D}^-]} = \frac{Pr[A(D_2) \in \hat{D}^-]}{Pr[A(D_1) \in \hat{D}^-]} + \\ \frac{Pr[A(D_2) \in \hat{D}^+]}{Pr[A(D_1) \in \hat{D}^-]} &\leq \alpha^d + \frac{0.5d \exp(\frac{d-K}{b})}{Pr[A(D_1) \in \hat{D}^-]} = \alpha^d + \frac{0.5d \exp(\frac{d-K}{b})}{Pr[A(D_1) \in \hat{D}^-]}, \\ \text{hence } Pr[A(D_2) \in \hat{D}] &\leq \alpha^d Pr[A(D_1) \in \hat{D}] + 0.5d \exp(\frac{d-K}{b}), \end{aligned}$$

as desired.

²As long as $Pr[A(D_1) \in \hat{D}^-] \neq 0$, the denominator of all the ratios involved in the product is guaranteed to be non-zero; whereas, if $Pr[A(D_1) \in \hat{D}^-] = 0$, then the upper bound of α^d holds for it automatically.

³The denominator of any of the product terms is 0 only if $Pr[A(D_1) \in \hat{D}^-] = 0$, in which case the desired differential privacy guarantees follow from (9).

