Privacy-Aware Personalization for Mobile Advertising

Michaela Götz
Cornell University, Ithaca, NY, USA
goetz@cs.cornell.edu

Suman Nath
Microsoft Research, Redmond, WA, USA
sumann@microsoft.com

***Abstract*—We address the problem of personalizing ad delivery to a smart phone, without violating user privacy. We propose a flexible framework where users can decide how much information about their private context they are willing to share with the ad server. Based on this limited information the server selects a set of ads and sends them to the client. The client then picks the most relevant one based on all its private contextual information. The optimization of selecting the most relevant ads to display is done *jointly* by the user and the server under two constraints on (1) privacy, i.e., how much information is shared, and (2) communication complexity, i.e. how many ads are sent to the client. We formalize the optimization problem, show that it is NP-hard, and present efficient algorithms with tight approximation guarantees. We then propose the first differentially-private distributed protocol to compute various statistics required for our framework *even in the presence of a dynamic and malicious set of participants*. Our experiments on real click logs show that reasonable levels of privacy, efficiency, and ad relevance can be achieved simultaneously.**

I. Introduction

The increasing availability of smartphones, equipped with various sensors and Web browsing capability, provides new opportunities for personalizing online services delivered to the phones based on users' activities and contexts. Recent work has shown that data collected from phone sensors such as GPS, accelerometer, audio, and light sensors can be used to infer a user's current context (such as whether she is at home or at work, alone or with friends, walking or driving, etc.) [26]. Based on such context, online services such as advertising can be personalized. For example, an online advertiser can use users' past and current contexts and activities, along with their browsing and click history, to show ads preferentially to the users who are more likely to be influenced by the ads. If a user who likes Italian food (inferred from her past browsing history) is found to be walking (inferred from the accelerometer sensor data) alone (inferred from the audio data) around lunch time, she can be shown ads of popular (inferred from other users' past locations) Italian restaurants within walking distance of her current location (inferred from the GPS). Such highly targeted advertising can significantly increase the success of an ad-campaign in terms of the number of resulting views or clicks of an advertised web page or the number of resulting purchases made. Similarly, context-aware personalization can improve the result quality of searches for close-by businesses.

However, such personalization raises serious privacy concerns. Personalized services rely on private information about a user's preferences and current and past activities. Such information might allow for identification of the user and her activities, and hence the user may not be willing to share information required for personalization. In the previous example, the user may not be willing to reveal the fact that she is out of office during office time. Moreover, clicks on ads personalized with private data can also leak private information about the user [14], [21].

To address such privacy concerns, several recent works proposed privacy-preserving personalization techniques based on a user's past browsing and click history, mostly for online advertising and search. In the context of privacy-preserving targeted advertising, recent works developed specific solutions for letting users control how much private information leaves their browser [9], disguising user identities and obfuscating/aggregating private information before releasing it [15], and correctly billing advertisers without revealing what results a user clicked on [32]. However, these proposals do not address the key problem of specifying *how* the personalization is done.

A personalized ad delivery system has two main components: (1) *statistics gathering* to learn personalization rules by interpreting users' contexts and clicks as implicit relevance feedback, and (2) *ad delivery* to select the best set of ads to show to a user based on her current context. Both components use private data – each leading to privacy concerns that must be addressed to ensure users' end-to-end privacy. In this paper, we provide such a comprehensive solution. We take a user-centric, flexible and modular approach. We empower the user to decide for each component if at all and if so, how much and under which privacy guarantees to supply her data: For the ad delivery, she can trade-off information disclosure, communication complexity, and utility. For statistics gathering, she can decide whether or not to participate in the collection of aggregate statistics. If she decides to participate, she is provided a strong formal privacy guarantee, namely a probabilistic version of differential privacy [7]. Our privacy guarantees for ad delivery and statistics gathering compose nicely to an overall privacy model: Independent of how much information a user discloses, if she decides to participate in the gathering of statistics to learn personalization rules, she is guaranteed that those aggregate statistics hardly affect an adversary's belief about her. This overall privacy model follows because that differential privacy does not make assumptions about an adversary's background knowledge.

Private Ad Delivery. We first formalize the task of choosing

and delivering personalized ads from a server to a client as an optimization problem with three important variables: (1) privacy, i.e., how much information about the user's context is shared with the server, (2) communication efficiency, i.e. how few ads are sent to the client, and (3) utility, i.e., how useful the displayed ads are to the user in terms of revenue and relevance. We show in Section III that it is impossible to maximize all three design goals simultaneously. Several previous works on privacy-preserving search present extreme points in the trade-off space: they propose to personalize search results based on a user's private information either at the server only [22], [34] or at the client only [30]. A server-only solution achieves optimal efficiency at the cost of privacy or utility, while a client-only solution ensures optimal privacy but sacrifices efficiency or utility. Other recent work on privacy-aware location-based services [27] proposes a hybrid solution where the server and a client jointly select public objects near the user without revealing her exact location. The client sends a broad region to the server, gets back a set of all objects in the region, and displays the closest objects to the user. This approach can achieve perfect or near-perfect utility and a user-configurable level on privacy but can potentially incur a high communication cost.

While these three solutions might be desirable for some systems they represent only three extreme points in a vast trade-off space of privacy, communication efficiency, and utility. Other systems may find other points in the space more attractive. Our goal is to develop a more flexible and tunable hybrid optimization framework to trade off privacy, communication efficiency and utility. In our framework users can decide how much information about their sensor readings or inferred contexts they are willing to share with the server. Based on this limited information the server selects a set of ads or search results, with bounded communication overhead, and sends them to the client. The client then picks and displays the most relevant ad based on all its private information. The ads sent by the server and the ad displayed at the client ought to be chosen in a way that maximizes utility. In other instantiations, our framework can optimize efficiency given a lower bound of revenue and a privacy constraint, or a combination of revenue and efficiency. Such a flexible framework is extremely useful in practice as different systems may have different priorities on these variables. Note that, the aforementioned privacy-preserving personalization solutions are special cases of our framework; and our framework can be configured to explore other attractive points in the trade-off space of privacy, communication efficiency, and utility. We formalize hybrid personalization as an optimization problem between users and the ad-serving server and show that the problem is NP-Hard. We present an efficient greedy algorithm for hybrid personalization with tight approximation guarantees.

Private Statistics Gathering. Our framework chooses personalized ads based on historical information of which ads users in some given context click on (i.e., click through rates or CTRs of ads). However, estimating CTRs constitutes a big privacy challenge: users are often unwilling to reveal their exact context and even their clicks because they leak information about their private contexts. To this end, we propose a novel differentially-private aggregation protocol to compute CTRs without a trusted server. A unique aspect of dealing with a large population of mobile users is that a small fraction of users can become unavailable or behave maliciously *during* the course of computing CTRs. For example, a user may turn off her mobile device any time or may want to answer an aggregation query only at a convenient time when her phone is being charged and connected through a local WiFi network. Another user might decline to participate in the exchange of certain messages in the protocol. Yet another user might leave or join the community of mobile users. Existing algorithms [6], [29], [31] do not efficiently handle such dynamics during query time (more details in Section V-C), making them unsuitable for estimating CTRs from mobile users. In contrast, our algorithm can handle such dynamics.To the best of our knowledge, our algorithm is the first differentially-private algorithm that provides accurate aggregate results efficiently even when a large fraction of data sources become unavailable or behave maliciously during one or more queries.

We have evaluated our algorithm with a large click log of location-aware searches in Microsoft Bing for mobile. Our experimental results show that even though privacy, communication efficiency, and utility are conflicting goals, reasonable levels of all these three goals can be achieved simultaneously.

In summary, we make the following contributions:

- We formalize personalization of ads based on private mobile contexts as an optimization problem between users and the ad-serving server (Section III).
- We show that the optimization problem is NP-hard and present an efficient greedy algorithm with a tight approximation guarantee (Section IV).
- We develop a differentially-private protocol for estimating statistics required for personalization without a trusted server. In contrast to existing algorithms, our algorithm can tolerate a small fraction of users being unavailable or malicious during query time (Section V-C).
- We evaluate effectiveness and robustness of our solution on a large click log of location-aware searches in Microsoft Bing for mobile. Our results illustrate important trade-offs between privacy, communication efficiency and utility in personalized ad delivery (Section VI).

Note that our results can be applied to personalize not just online advertising but also other online services based on user's fine-grained contextual information including local search and recommendation services. For concreteness, we consider advertising throughout the paper.

II. Related Work

Personalization has been successfully implemented in a varieties of areas including Web search and advertising. Recent work has raised privacy concerns about personalization based on private information because a click on an ad/Web page can

leak some private information about the user [14], [21]. Therefore, privacy-preserving personalization has recently received a lot of attention not just in the academic community but also in the media.[1]

Targeted Advertising. Before we discuss previous work on privacy-aware targeted advertising, let us briefly review how existing search engines personalizes ads to be displayed alongside with Web search results: The advertisers bid money on keywords. For an incoming query, the subset of ads bidding on keywords in the query is determined. From this subset the ads with the highest bids multiplied with their quality score are chosen. The most important factor of the quality score is the click-through-rate. The context considered encompasses the exact query and also geographic information available from the user submitting the query. Other factors of the quality score are the quality of the landing page and its page loading time.[2] Our work builds up on this approach by incorporating private data from sensors on mobile phones into the context and adding privacy guarantees to the overall scheme.

Several recent works have addressed various aspects of privacy-preserving targeted advertising [9], [15], [20], [32]. Adnostic [32] considers doing personalization at the client and proposes a privacy-aware accounting tool to correctly bill the advertisers without leaking which user clicked on what ads. RePriv [9] supports verified miner through a browser plug-in which allows a user to control how much private information leaves her browser and to which web site. Privad [15] proposes disguising user's identity and obfuscating/aggregating private information before releasing it. All these works are orthogonal to our work in that they do not specify *how* the personalization is done and some of them consider doing the personalization in the client only. Our personalization algorithm can easily be incorporated into these existing systems.

Personalized Search. Here a user's interest profile is established based on her browsing history and search results are being re-ranked based on how well the content of the page matches her interest. The re-ranking can either be done by the user [30] or the search engine [22], [34]. If the re-ranking is done by the search engine then users can decide how much information they are willing to share about their profile for which help and guidance is given by some work [22], [34]. Our approach to personalization is very different than these existing approaches: we personalize at the granularity of contexts, instead of individual users (i.e., the same user will see different results/ads under different contexts) and we personalize jointly at the server and the client.

Location-Based Services. Our context generalization technique is similar to the *region cloaking* technique in privacy-aware location-based services, where instead of the user's exact location, a region around the location is released [5], [12], [11], [19], [27]. Similar techniques have been used for releasing non-spatial attributes as well [28]. When used for interactive services, most of these solutions (e.g., [27]) follow a hybrid approach in which, given a generalized context, the server returns a superset of the results [19]. This can lead to high communication cost. Our approach differs by taking this cost into consideration. Unlike these works, we formalize the problem of optimizing personalization within the trade-offs of information disclosure, utility and communication cost.

[1] http://www.nytimes.com/2010/10/23/technology/23facebook.html

[2] See for instance http://adwords.blogspot.com/2008/08/quality-score-improvements.html

III. The Framework

Our framework has three classes of participants: The *users* who are served ads (also referred to as clients) in their mobile contexts, the *advertisers* who pay for clicks on their ads, and the *ad service provider* (also referred to as the *server*) who decides which ads to display and who is being paid for clicks by the advertisers.

Overall Privacy Model. Our two components of ad delivery and statistics gathering use private data differently—ad delivery uses a user's current context, while statistics gathering uses historical context and click data from many users. For statistics gathering, we provide a version of differential privacy [7] which is a very strong privacy guarantee. However, it seems to be incompatible with personalization that requires a single user's current context. Therefore, in the spirit of many existing personalization systems [9], [22], [34], [19], we ensure user privacy through limited information disclosure. Here a user gets to decide how much information about her context to share with the server. We can further extend our privacy guarantee to anonymity: A user can determine, based on the gathered statistics, how much she has to generalize her context so that w.h.p. at least k other users reported the same context. Exploring other privacy guarantees is an interesting direction for future work. As mentioned in Section I, our privacy guarantees for two components compose nicely to an overall privacy model: The participation in the gathering of statistics hardly influences an adversary's belief about a user, irrespective of how much information the user discloses for ad delivery.

A. Desiderata for Ad Delivery

Our desiderata include goals of the individual participants as well as general system requirements. Informally, we have the three orthogonal design goals: Privacy, efficiency, and revenue and relevance.

► **Privacy:** The user would like to limit the amount of information about her mobile context that is sent to the server. The information disclosure about a user in context c can be limited by generalizing the user's context and only sending the generalized version $\hat{c}$ to the server, e.g. instead of revealing that the user is *skating* in *Central Park*, the user only discloses to be *exercising* in *Manhattan*. The generalization of context is done over a hierarchy that we will describe later. For a context c that can be generalized to $\hat{c}$ we write $c \to \hat{c}$.

► **Efficiency:** The ad serving system should be efficient both in terms of communication and computational cost since the user wants ads fast and without draining much battery power on her mobile device and the ad service provider wants to run

his system at low operating cost. For simplicity, we focus on communication cost between the server and a client since it is the most dominant cost of serving ads to mobile devices. Our results can trivially be extended to consider computational cost of the server and the client as well.

► **Revenue and Relevance:** The ad service provider seeks to maximize its revenue. The user is only interested in relevant ads. The goal of the ad service provider is to display an ad from a given set of ads $\mathcal{A}$ that maximizes the expected revenue. For a click on ad a the ad service provider is being paid p_a from the advertiser. Clearly not all users click on an ad. We denote by $\mathsf{CTR}(a|c)$ the context-dependent click-through-rate, i.e., the fraction of users who actually clicked on it in context c among those who were served the ad in context c. The expected revenue of displaying an ad a to a user in context c is $p_a \cdot \mathsf{CTR}(a|c)$. We view clicks as an indicator for relevance: Users who are interested in an ad click on it. Maximizing the relevance means maximizing the expected number of clicks (by displaying to a user in context c the ad a with the highest context-dependent $\mathsf{CTR}(a|c)$), which is related to the goal of maximizing the expected revenue.

B. Trade-Offs

Our three design parameters, limited information disclosure, efficiency, and relevance are conflicting. It is easy to see that *optimizing all three design goals simultaneously is impossible*. Consider the task of showing only one ad to the user. Then, in case of minimum information disclosure (i.e., the user does not send any information about her context) and highest communication efficiency (i.e., the server is only allowed to send a single ad), the server needs to choose the ad without any knowledge of user's context or preference. As long as there is an ad whose CTR varies depending on the exact context, any solution with minimum information disclosure and highest communication efficiency will not be able to detect when to show this ad yielding very suboptimal relevance and revenue. If we want to improve the relevance, either the user needs to send some information to the server, or the server needs to send more than one ads for the user to perform local personalization; either way, information disclosure or efficiency becomes suboptimal.

If we drop any of our three design goals the problem becomes trivial. If there were no concerns about privacy, we could use a *server-only* scheme, where the user sends her context c to the ad service provider who serves the ad that maximizes the expected revenue, i.e., $p_a \cdot \mathsf{CTR}(a|c)$. This is a very efficient scheme that maximizes revenue. If there were no efficiency concerns, we could use a *client-only* scheme, where the server simply sends all ads $\mathcal{A}$ so that the user can pick the ad that maximizes the expected revenue. Alternatively, we could use expensive cryptographic protocols for private information retrieval [10]. No user information is disclosed to the server and optimal revenue is achieved. However, due to the large number of ads the performance is terrible. Finally, if there was no financial incentive and no interest in relevant ads then one could stop serving ads all together to avoid any concerns regarding efficiency and privacy. In practice, one has to find reasonable trade-offs between the three design goals.

Fig. 1. Example use of our framework.

C. Our Optimization Goal

In our framework, the user gets to decide what information about her context to share with the server. Based on this information the server selects some k ads $A \subset \mathcal{A}$ that are sent to the user. Here, the parameter k determines the communication cost.[3] The user is going to pick an ad among those to display. The set of ads and the ad to display should be chosen in a way that maximizes the revenue. Figure 1 shows an example of an execution of our framework.

Our flexible framework can be optimized for various objective functions involving the three parameters mentioned before. For concreteness, we now assume that there are constraints for both the information disclosure (determined by the users) and the communication cost (determined based on the current load of the network); and we seek to maximize the revenue under these constraints. We will discuss alternative objective functions later in this section.

1) Server-Side Computation: The server needs to determine the best k ads to send to the user given only the partial information $\hat{c}$ it has. Suppose that server not only has information on the click-through-rates, but also on the frequencies of the contexts. If this is all the information the server has then from its point of view the expected revenue of sending a set A of ads to the user depends on the users true context c; it is $\max_{a \in A} p_a \cdot \mathsf{CTR}(a|c)$. Since the server knows only the generalized context $\hat{c}$, it considers the probability of each of the possible contexts $c' \to \hat{c}$ and the expected revenue of A in this context c'. With this limited information the expected revenue of a set of ads A for a generalized context $\hat{c}$ is

$$\mathrm{E}[\mathsf{Revenue}(A|\hat{c})] = \sum_{c:c\to\hat{c}} \Pr[c] \cdot \max_{a\in A} p_a \cdot \mathsf{CTR}(a|c).$$

It is the server's task to select the set A^* of k ads from $\mathcal{A}$ that maximize the expected revenue given only the generalized context $\hat{c}$ of the user, i.e.,

$$A^* = \arg \max_{A\subset\mathcal{A}:|A|=k} \mathrm{E}[\mathsf{Revenue}(A|\hat{c})]$$

Finding these k ads is NP-hard as we will show in the next section. However, we can employ approximation techniques to efficiently select a set of k ads with revenue close to the optimal revenue.

[3]Computation cost can also be included into k if needed.

2) Client-Side Computation: For a given set of ads A (chosen by the server as described above), a client in context c maximizes the revenue by selecting the ad

$$a^* = \arg\max_{a \in A} p_a \cdot \mathsf{CTR}(a|c).$$

3) Instantiations of the Framework: Our framework encompasses client-side personalization by setting $\hat{c}$ to the most generalized context that does not leak any information about the client's true context. In this case the personalization takes place exclusively on the client-side. Our framework also encompasses server-side personalization by setting $k = 1$ in which case the client simply displays the ad sent by the server without further personalization. However, higher revenue can be achieved in our framework when some information is disclosed and the server sends back $k > 1$ results.

4) Extensions: **Alternative Objective Functions.** So far, we treated both the amount of information disclosure and the communication cost k as hard constraints and we tried to maximize the revenue under these constraints. Instead, one might consider the communication cost as a variable and include it in an objective function that maximizes the value of (revenue $-\alpha \cdot k$). As we will see, all of these objectives can be solved with the techniques we discuss next.

Additional Constraints. While high revenue and high relevance of ads are related goals, they are not the same. Suppose the ad service provider receives a request from a user in context c. Suppose further there are two ads a_1, a_2 with $\mathsf{CTR}(a_1|c) = 0.1, \mathsf{CTR}(a_2|c) = 0.9$ and $p_{a_1} = \$0.1, p_{a_2} = \$0.01$. Ad a_1 has the higher expected revenue but a_2 is more relevant. While displaying a_1 maximizes short-term revenue it might not be the best long-term strategy. Recent work has found that the time users spend on viewing ads depends on the predictability of the quality of the ads [3]. Our framework can reconcile relevance and short-term and long-term revenue goals by adding a constraint on CTR.

IV. Optimization Algorithms

In this section we explain how client and server can efficiently compute their part of the optimization. We consider a specific instantiation of the optimization problem where the user fixes her privacy requirement; the client and the server then try to maximize revenue for a given bounded communication complexity (k). At the end of the section we discuss extensions and alternatives.

The client is supposed to compute the equation in Sec. III-C2, which can be computed quickly since the number of ads from which the client picks one is small (bounded by k).

However, the server's task to select a set of k ads from $\mathcal{A}$ that maximize the expected revenue given only the generalized context $\hat{c}$ of the user is much more demanding. In fact, a reduction from the maximum coverage problem shows:

Proposition 4.1: For a generalized context $\hat{c}$ it is NP-hard to select the revenue-maximizing set of k ads A^*:

$$A^* = \arg \max_{A \subset \mathcal{A}: |A| = k} \sum_{c: c \to \hat{c}} \Pr[c] \cdot \max_{a \in A} p_a \cdot \mathsf{CTR}(a|c)$$

Algorithm 1 Greedy algorithm for selecting ads maximizing the expected revenue.

```
Greedy(ads A, generalized context ĉ, threshold k)
Init A = ∅
while |A| < k do
  for a ∈ A do
    b_a ← E[Revenue(A ∪ {a}|ĉ)] − E[Revenue(A|ĉ)]
  A ← A ∪ {argmax_a b_a}
return A.
```

Proof: We prove this by a reduction from the maximum coverage problem. In the maximum coverage problem we are given a collection of sets $\mathcal{S}$ over some finite universe U and our goal is to find a subset of these sets S of size at most k that maximizes the number of covered elements in the universe $\left|\bigcup_{s \in S} s\right|$.

We set the set of contexts equal to the universe U. For each context $c \in U$ we set $\Pr[c] = 1/|U|$. For each set s in the collection $\mathcal{S}$ we create an ad a so that for the elements c in s we set $\mathsf{CTR}(a|c) = 1$ and 0 for all other elements. We set $p_a = 1$ for all ads. Consider a generalized context $\hat{c}$ that generalizes all contexts c. Then for a given set A of k ads, the expected revenue is $1/|U| * \sum_c \max_{a \in A} \mathsf{CTR}(a|c)$, which is equal to the number of covered elements by the corresponding set of sets S divided by the universe size. Thus an optimal solution to the ad selection problem yields an optimal solution to the maximum coverage problem. ∎

Moreover, the maximum coverage problem cannot be approximated within $\frac{e}{e-1} - o(1)$ [8] assuming $P \neq NP$.

A. Approximation Algorithm

Algorithm 1 shows a greedy algorithm, called Greedy, that constructs a set A of k ads incrementally. It starts with an empty set A of ads and in each round, the ad that increases the expected revenue the most is being added A.

Interestingly, the output of this simple greedy algorithm approximates the optimal value to within a factor of $(1-1/e)$. Even though such greedy algorithms have been shown to provide a similar guarantee for the maximum coverage problem [18], our problem is considerably more complex: In the coverage problem a set either *fully* covers an element or not. In our case an ad a can *partially* "cover" a context c that can be generalized to $\hat{c}$. Thus a new analysis is required. We first define a *benefit function* of adding a set A' to a set A:

$$\mathrm{B}(A, A') = \mathrm{E}[\mathsf{Revenue}(A \cup A'|\hat{c})] - \mathrm{E}[\mathsf{Revenue}(A|\hat{c})].$$

In order to prove an approximation guarantee we make the crucial observation that the benefit function is submodular.

Fact 4.2: The benefit function is submodular, i.e., for all sets of ads $A_1 \subset A_2$ and for all A: $\mathrm{B}(A_1, A) \geq \mathrm{B}(A_2, A)$.

However, due to the complex nature of our problem, the submodularity property alone does not imply our approximation guarantee.

Let $a_1, \ldots, a_k$ be the k ads chosen by Greedy in the order they were chosen. To simplify the analysis we define the

benefit of the i^{th} ad to be b_i and the expected revenue after adding the first l ads to be $b(l) = \sum_{i=1}^{l} b_i$. Similarly, let $a_1^*, \ldots, a_k^*$ be the k optimal ads in any fixed order. We define the benefit of the i^{th} ad to be b_i^* and the expected revenue after adding the first l ads to be $b^*(l) = \sum_{i=1}^{l} b_i^*$.

Lemma 4.3: $\forall l \in [k]$: $b_l \geq \frac{b^*(k)-b(l-1)}{k}$.

Proof: The benefit of adding $a_1^*, \ldots, a_k^*$ to $a_1, \ldots, a_{l-1}$:

$$\mathrm{B}(\{a_1, \ldots, a_{l-1}\}, \{a_1^*, \ldots, a_k^*\}) \geq b^*(k) - b(l-1).$$

Now, since this benefit can also be written as $\sum_{i=0}^{k} \mathrm{B}(\{a_1, \ldots, a_{l-1}\} \cup \{a_1^*, \ldots, a_{i-1}^*\}, \{a_i^*\})$ it follows from an averaging argument that $\exists i, 1 \leq i \leq k$: $\mathrm{B}(\{a_1, \ldots, a_{l-1}\} \cup \{a_1^*, \ldots, a_{i-1}^*\}, \{a_i^*\}) \geq \frac{b^*(k)-b(l-1)}{k}$. By submodularity this implies that

$$\exists i, 1 \leq i \leq k : \mathrm{B}(\{a_1, \ldots, a_{l-1}\}, \{a_i^*\}) \geq \frac{b^*(k) - b(l-1)}{k}.$$

Since the greedy algorithm in round l selected the ad a_l that maximizes $\mathrm{B}(\{a_1, \ldots, a_{l-1}\}, \cdot)$, the benefit of that ad, b_l, has to be at least $\frac{b^*(k)-b(l-1)}{k}$ which completes the proof. ∎

We use this lemma to prove by induction.

Lemma 4.4: $\forall l \in [k]$: $b(l) \geq (1 - (1 - 1/k)^l) b^*(k)$.

Proof: Proof by induction on l.

$l = 1$**.** Lemma 4.3 tells us that $b_1 \geq \frac{b^*(k)}{k} = (1 - (1 - 1/k)^1) b^*(k)$.

$l \to l + 1$**.**

$$b(l+1) = b(l) + b_{l+1} \geq b(l) + \frac{b^*(k) - b(l)}{k}$$
$$= \frac{b^*(k)}{k} - b(l)(1 - 1/k) \geq \frac{b^*(k)}{k} - (1 - (1 - 1/k)^l) b^*(k)(1 - 1/k)$$
$$= (1 - (1 - 1/k)^{l+1}) b^*(k)$$

The first inequality follows from Lemma 4.3 and the second follows from the induction hypothesis. ∎

The main theorem on the approximation guarantee of the greedy algorithm follows.

Theorem 4.5: The greedy algorithm approximates the optimal value to within a factor of $(1 - 1/e)$.

Proof: By Lemma 4.4 we have that

$$b(k) \geq (1 - (1 - 1/k)^k) b^*(k) \geq (1 - 1/e) b^*(k)$$

∎

B. Extensions

Alternative Objective Functions. As mentioned in Section III-C4, an objective function may include the communication cost as well. One straightforward way to handle such an objective function is to run Greedy for all values of k and pick the outcome that maximizes our new objective function.

However, by exploiting the submodularity of the benefit function, we can compute the alternative objective function much more efficiently. All we have to do is to replace the while condition in Algorithm 1 by a new one that checks whether the current value of $\mathrm{E}[\mathsf{Revenue}(A)] - \alpha \cdot |A|$ is increasing. This modification works correctly because the following argument shows that as we increase k, our new objective function increases until at some point it starts to decrease and never increases again. Suppose in round k' the expected revenue of $A = \{a_1, \ldots, a_{k'}\}$ minus $\alpha \cdot k'$ is not increasing any longer, i.e.,

$$\mathsf{Revenue}(\{a_1, \ldots, a_{k'}\}) - \alpha k'$$
$$\leq \mathsf{Revenue}(\{a_1, \ldots, a_{k'-1}\}) - \alpha(k' - 1).$$

At this point the benefit of adding $a_{k'}$ is at most α. Due to submodularity, the benefit of any future ad being added to A can only be smaller and thus will never lead to an increase of the objective function.

Additional Constraints. We can incorporate a constraint on the ad relevance by setting the CTR to zero whenever it is below a certain threshold. Then no ad with CTR below this threshold will ever be displayed at the client. Our algorithm remains close to optimal under this constraint as well.

Advertisers' Control. Our algorithm can also incorporate additional restrictions posed by advertisers on the contexts in which their ads are being displayed. Very much like advertisers for sponsored results in Web search who can bid on keywords in a query, our advertisers can bid on contexts of the users. To make sure the ad is only being displayed on these contexts we can make the payment p_a context-dependent and set it to 0 for all but the contexts the advertiser bids on.

V. Privacy-Preserving Statistics Gathering

The optimization framework described in previous sections uses various statistics; in this section we will describe how to obtain those. Specifically, we will develop protocols for computing the probability distribution over contexts, $\Pr[c]$, and the context-dependent click-through-rates, $\mathsf{CTR}(a|c)$. $\Pr[c]$ can be estimated as the number of times a user reported to be in context c divided by the total number of reported contexts. For an ad a and a context c, $\mathsf{CTR}(a|c)$ can be estimated as the number of times a user in context c reported to have clicked on a divided by the number of times a user in context c reported to have viewed on a. These estimations are based on Count (and related) queries; hence we focus on privacy-preserving computation of Count queries in the rest of the paper.

Note that the above statistics can be estimated well for contexts with a lot of user data (e.g., clicks). However, one challenge of considering context attributes beyond location (unlike location-based services) is that sufficient click data may not be available for a large number of contexts. For such rare or new contexts, the estimates can be noisy or not even be defined. One option would be to simply not show any ads to a user in a rare context. However, this would seriously harm utility since there is typically a long tail of rare contexts. Instead we suggest to estimate $\Pr[c]$ and $\mathsf{CTR}(a|c)$ for a rare context c based on contexts similar to c for which we have enough click data. Coming back to our example from Section III-B, if we do not have enough click data for users who were taking weekly Yoga classes in San Francisco (c), we might use clicks from users in close-by locations who were doing some sort of physical activity ($\tilde{c}$) in order to estimate

the statistics for the context c. This helps us to increase the coverage of the targeted ads. However, those ads might be of lower quality. We can trade-off coverage and relevance by adding a constraint on the CTR for the displayed ads as discussed in Sections III-C4 and IV-B.

A. Desiderata

We have the following goals in computing the statistics.

► **Privacy in the Absence of a Trusted Server.** Since users in our optimization framework do not trust the server with their private data, we do not assume to have a trusted server who could collect all user data to compute statistics. Without a trusted server, we need a distributed aggregation protocol that protects user privacy, even under adversarial scenarios such as when a fraction of the participants behave maliciously, send bogus messages, or collude with each other. This requirement sets our work apart from previous work on publishing privacy-preserving count statistics of a search log that all assume a trusted third party (see [2] and the references therein).

► **Scalability.** We need to scale the computation to a large number of users and contexts.

► **Robustness to a Dynamic User Population.** With a large number of transient mobile phone users, we cannot expect all of them to be available and willing to engage in all rounds of our protocol. Users decide what queries they are willing to answer and when (e.g., when the phone is being charged and connected through a WiFi network). Therefore, our protocol should be able to deal with a dynamic user population without sacrificing privacy or scalability.

B. Privacy Preliminaries and Assumptions

The main mechanism we employ to build a scalable and robust protocol is to use two servers: one responsible for key distribution and the other responsible for aggregation. The idea of using two servers to build secure protocols has been used previously [1], [10], [15] in different applications; we here use it for privacy-preserving aggregation. We assume secure, reliable, and authenticated communication channels between servers and users. In addition, we make the following two key assumptions, similar to those made in several previous work [29], [31].

1. Honest-but-Curious Servers. *The two servers honestly follow the protocol. They are curious but do not collude with anyone.*

2. Honest Fraction of Users. *At most t fraction of users can be malicious or unavailable during the protocol. This means, at least a fraction of $1-t$ users honestly follow the protocol. The honest users can be curious but do not collude with anyone.*

We aim to ensure user privacy with respect to all participants in the distributed protocol. There are many ways to define privacy in data publishing including k-anonymity,ℓ-diversity, proximity privacy, t-closeness, and (d,γ)-privacy. We refer the reader to an excellent survey [4]. For the purpose of this paper, we work with ϵ-differential privacy [7], a strong guarantee that does not make assumptions about an adversary's background knowledge. This privacy guarantee can therefore nicely be combined with the limited information disclosure from the optimization. The idea behind it is that whether or not the contexts and clicks of a single user were used in the computation hardly affects the released outcome. Therefore, a user given the choice of whether or not to supply her data has hardly any incentive to withhold it. The parameter ϵ determines how much the outcome may be affected. We use a probabilistic relaxation of differential privacy proposed by Machanavajjhala et al. This definition was developed under the assumption that a trusted server holds all the click data in a context-driven ad log L.

Definition 1: [23] An algorithm M satisfies (ϵ,δ)-probabilistic differential privacy if for all context-driven ad logs L we can divide the output space Ω into two sets Ω_1,Ω_2 such that

$$(1)\ \Pr[M(L)\in\Omega_2]\le\delta,\ \text{and}$$

for all neighboring ad logs L' differing and for all $O\in\Omega_1$:

$$(2)e^{-\epsilon}\Pr[M(L')=O]\le\Pr[M(L)=O]\le e^{\epsilon}\Pr[M(L')=O]$$

This definition differs from ϵ-differential privacy in that it has a second parameter δ. The two definitions are equivalent for $\delta=0$.[4] In general, δ bounds the probability that a privacy breach (according to ϵ-differential privacy) occurs. The set Ω_2 contains all outputs that are considered privacy breaches according to ϵ-differential privacy; the probability of such an output is bounded by δ. We can sanitize the output of any real-valued function $f:$ ad logs $\to\mathbb{R}^d$ to achieve probabilistic differential privacy by adding Gaussian noise to $f(L)$. The standard deviation of the noise depends on the L_2-*sensitivity* of f which describes how much the value of the function can change if a single user's data is deleted from the input. This change is measured by the L_2-norm.

Theorem 5.1: For $\epsilon\le 1$ and $\sigma^2\ge s^2 2\log(2/\delta)/\epsilon^2$ adding Gaussian noise with variance σ^2 to a query with L_2-sensitivity s gives (ϵ,δ)-probabilistic differential privacy.

This theorem has been established for δ-approximate ϵ-indistin-guishability [6], another probabilistic relaxation of differential privacy. Probabilistic differential privacy, however, is the stronger definition [13]. It is not difficult to see that the proof of [6], [25] extends to probabilistic differential privacy. Probabilistic differential privacy has a nice composition property: Concatenating the output of a (ϵ_1,δ_1)-probabilistic differentially private algorithm with that of a (ϵ_2,δ_2)-probabilistic differentially private algorithm guarantees $(\epsilon_1+\epsilon_2,\delta_1+\delta_2)$-probabilistic differential privacy.

We can adapt the privacy definition to the case where no trusted server exists to run M and instead some distributed protocol is used. We show that the output of the protocol

[4]Our motivation for resorting to probabilistic differential privacy as opposed to differential privacy is that it can be realized in a distributed manner much more easily: Gaussian noise with variance σ^2 can be generated in a distributed manner by N parties, by summing up N independent random samples from the Gaussian distribution with variance σ^2/N. Adding Gaussian noise however is not sufficient to achieve ϵ-differential privacy.

preserves privacy. For the two servers we show that their view of the messages sent and received can be generated from the output only. Thus they cannot learn anything beyond what they can learn from the privacy-preserving output. For the users we treat their entries in L (L', resp.) as input. We denote by $M(L)$ ($M(L')$, resp.) all the messages they send and receive and show that it they preserve differential privacy even when a subset of the users collude.

A basic building block of our protocol is a procedure for computing a sum over user values. We will use it to compute how many users clicked on an ad a in context c.

C. A Privacy-Preserving, Distributed Counting Protocol

1) Previous Work: Previous work on distributed counting protocols [29], [31], [6] provides strong privacy guarantees (and [6] even gives accuracy guarantees). However, they are inefficient when the user population changes quickly. This is problematic in our setting with a large number of transient mobile users.

Early work on by Dwork et al. [6] exchange a number of messages quadratic in the number of users. This is prohibitively expensive in our setting. Follow-up work by Rastogi et al. [29] and Shi et al. [31] reduced the number of messages being exchanged to be linear in the number of users. Briefly, both the protocols of [29], [31] start with a setup phase in which an honest server generates secrets and distributes them to users such that secrets of all users add up to a constant (e.g., 0 for [31]). After this setup phase, a series of count queries can be computed in a privacy-preserving manner assuming that all available users in the setup phase will participate in the aggregation phase. However, when a single user becomes unavailable, no further queries can be answered, until a new setup is established or until the user returns. Thus, for a query to be successfully answered, the setup phase followed by the aggregation phase must be repeated until they both run on the same stable set of users. In Section VI-D, we empirically show that even under modest assumptions on user dynamics, these protocols require impractically high numbers of setup phases for each query.

2) Our counting protocol: Protocol 1 describes our protocol $\mathsf{Count}(t, \sigma^2)$. Each user u_i for $i = 1, \ldots, N$ holds a bit b_i. The protocol computes a noisy version of the sum $\sum b_i$.[5] The parameter t is an upper bound on the fraction of malicious or unavailable users, σ^2 is the amount of noise. If the upper bound t is violated and more users turn out to be malicious or unavailable the privacy guarantee degrades and/or the protocol needs to be aborted before Step 4 and restarted (with a larger value for t). As t increases the share of noise each participant adds to its bit increases.

Efficiency. The number of messages exchanged in Count is linear in the number of users, similar to the most efficient previous solutions [29], [31]. Also, when executing Count multiple times messages can be batched.

[5]In practice, we would use a discretized version of Gaussian noise and do all computations modulo some large number q as done in [31]. We would pick the r_i uniformly at random from $0, \ldots, q-1$.

Protocol 1 Robust, distributed count computing a privacy-preserving version of the sum over all private user bits b_i.

$\mathsf{Count}(\sigma^2, t)$

1) Each user i with bit b_i samples a number k_i uniformly at random.
2) Each user i samples r_i from $\mathcal{N}(\sigma^2/((1-t)N-1))$.
3) Each user i uses reliable communication to atomically send k_i to Server 1 and $m_i = b_i + r_i + k_i$ to Server 2.
4) Server 2 sums up all incoming messages m_i. It forwards $s = \sum m_i$ to Server 1.
5) Server 1 subtracts from s the random numbers k_i it received and releases the result $\sum b_i + r_i$.

Robustness. Unlike previous protocols [29], [31], Count successfully terminates as long as at least $(1-t)N$ users send messages to Server 1 and 2. When Count is executed multiple times, it suffices that for each execution, at least $(1-t)N$ possibly different users participate. Thus, our protocol can deal with unavailable users much more efficiently than previous protocols. Unlike these previous protocols, our protocol does not expect the secrets of participating users to add up to a predefined constant. Rather, it lets users independently choose their own secrets k_i (Step 1) and uses secrets of only the users who have participated throughout the entire protocol (Step 5). Thus, even if some t fraction of users become unavailable during the protocol, the protocol simply ignores their secrets and successfully terminates.

Privacy. We split the privacy analysis into two parts. First we consider the case where at most a fraction of t users is unavailable. In the second part we consider malicious users.

Proposition 5.2: Consider $\epsilon \leq 1$ and $\sigma^2 \geq 2\log(2/\delta)/\epsilon^2$. The output of $\mathsf{Count}(\sigma^2, t)$, guarantees (ϵ, δ)-probabilistic differential privacy in the presence of up to tN unavailable users.

Sketch: In Step 4 of our protocol, Server 1 receives s that contains at least $(1-t)N$ messages from honest users each containing a noise term $\mathcal{N}(\sigma^2/((1-t)N-1))$. From the linear decomposition property of Gaussian noise it follows that the decrypted sum $\sum_{i \in U} r_i + b_i$ includes noise with magnitude at least $\mathcal{N}(\sigma^2(1-t)N/((1-t)N-1))$. We now consider privacy with respect to various parties in the protocol.

(a) A non-participant of the protocol who can only observe the output. As long as $\epsilon \leq 1$ and $\sigma^2 \geq 2\log(2/\delta)/\epsilon^2$ releasing this sum preserves (ϵ, δ)-probabilistic differential privacy according to Theorem 5.1.

(b) User i who can view her own bit b_i, her own noise r_i, as well as the output. Note that, for user i,

$$\Pr[\mathsf{Count}(b_1, \ldots, b_N) = O | b_i, r_i] = \Pr[\sum_{j \neq i} b_j + r_j = O - b_i - r_i]$$

Now, using the fact that $\sum_{j \neq i} b_j + r_j$ still contains a noise

term of $\mathcal{N}(\sigma^2)$, we get the same (ϵ, δ)-probabilistic differential privacy according to Theorem 5.1.

(c) Server 1 sees $s = \sum m_i$ and the keys k_i. Those keys are completely independent of the private bits: If we have access to an oracle sampling $o = \sum_i b_i + r_i$, we can generate the same probability distribution over Server 1's view by sampling values $k_1, \ldots, k_N$ for the keys uniformly at random and setting $s = o + \sum k_i$. Thus the privacy guarantee of releasing only $\sum_i b_i + r_i$ extends to the view of Server 1.

(d) Server 2 who observers the messages m_i and the output. Informally speaking, random k_is perfectly hide the private bits: If we have access to an oracle sampling $o = \sum_i b_i + r_i$, we can generate the same probability distribution over Server 2's view by sampling values for m_is uniformly at random. Thus, the privacy guarantee of releasing only $\sum_i b_i + r_i$ extends to the view of Server 2. ∎

To finish our privacy analysis we consider malicious users.

Theorem 5.3: Consider $\epsilon \leq 1$ and $\sigma^2 \geq 2\log(2/\delta)/\epsilon^2$. Protocol $\mathsf{Count}(\sigma^2, t)$ guarantees (ϵ, δ)-probabilistic differential privacy of the honest or unavailable users in the presence of up to a fraction of t unavailable or malicious users.

Sketch: The privacy of unavailable users is trivially preserved. Thus we are concerned in the following with the privacy of honest users. In the worst case the adversary controls a set U' of tN users. For each user the adversary determines the messages m_i and k_i. The adversary can decide not to send any of the messages. Server 1 and 2 expect proper values as messages and will drop any other messages. W.l.o.g. assume that all messages sent by users in U' are proper values. Let $m_{U'}$ denote the sum of all messages m_i that are sent from users in M to Server 2 and let $k_{U'}$ denote the sum of all keys that are sent from users in M to Server 1. Note that some users in U' might contribute only to one sum and not the other. The output of the protocol will be $m_{U'} - k_{U'} + \sum_{i \notin U'} b_i + r_i$. We have that

$$\begin{aligned} &\Pr[\mathsf{Count}(b_1, \ldots, b_N) = O | m_{U'}, k_{U'}] \\ =&\Pr[\sum_{i \notin U'} b_i + r_i = O - m_{U'} + k_{U'}] \end{aligned}$$

Note that, the r_i are chosen so that $\sum_{i \notin U'} b_i + r_i$ contains enough noise to satisfy probabilistic differential privacy of the honest users.

Arguments similar to those in the proof of Proposition 5.2 show that privacy is also preserved regarding non-participants, honest users and the two servers. ∎

Additional Guarantees. Our protocol also provides some guarantees in case either Server 1 or Server 2 (but not both) are corrupted by an adversary (but not colluding with anyone). If Server 2 is corrupted by an adversary, we guarantee that the adversary will not be able to learn information that breaches privacy. This guarantee holds since Server 2 sends only the very last message of the protocol upon which no further action is taken. Similarly, if Server 1 is corrupted we guarantee that the adversary will not be able to learn information that breaches privacy. The adversary may send any value to Server 2 from which $\sum k_i$ will be subtracted. The output will be random in case the value is not a sum that contains exactly one term for each received messages m_i. In this case privacy is trivially preserved. Otherwise the value contains sufficient noise to preserve privacy. These guarantees are meaningful with regard to adversaries who want to learn private information. We do not guarantee that a malicious adversary cannot breach privacy: An adversary corrupting Server 2 can release the keys that allow the honest-but-curious Server 1 to learn $b_i + r_i$ which breaches privacy.

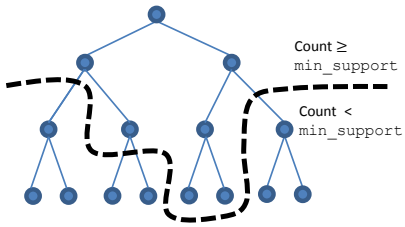


Fig. 2. Hierarchy H over contexts.

Generalization. Count computes a noisy sum over private bits. It can also be used to sum up real numbers. Moreover, it can be used to answer a set of sum queries with bounded L_2-sensitivity by setting σ according to Theorem 5.1.

D. Employing Count to compute CTRs

Possible Solutions. One possible way to employ Count to obtain privacy-preserving the statistics for various contexts is to compute noisy counts for all possible contexts and all possible ads. Another alternative approach, with better utility, would be to use multi-dimensional histograms [33], [16]. However, all these approaches have a running time at least linear in the number of possible contexts rendering them infeasible. Moreover, a large part of the computation is wasteful, since, as mentioned before, statistics computed for rare contexts are almost meaningless.

To address this problem, we opt for a simple top-down approach that can efficiently deal with sparse data by identifying and pruning the computations for rare contexts. The solution requires using a context hierarchy that specifies similarity of contexts and a top-down algorithm that computes the statistics by pruning rare contexts. Such a top-down algorithm has been used recently to find frequent signatures by gradually expanding the prefix of signatures with high noisy counts [24]. We adapt it to compute CTRs over a given context hierarchy.

Context Hierarchy. To define similarity over contexts, we reuse the hierarchies over which users generalize their context attributes. The context hierarchy is built by merging various attribute hierarchies; a concrete example will be shown in Section VI-A. This hierarchy tells us for each generalized

Algorithm 2 Privacy-preserving Estimates.

Estimates(context-driven ad log, noise scale λ, threshold `min_support`, contribution bound m, hierarchy H)
for each user **do**
  Delete all but m views or clicks on ads and their contexts of this user from the ad log.
return TopDown(ad log, root(H), λ, `min_support`)

context which attribute to generalize next.[6] Given some rare context, we can generalize it until we have a sufficient number of clicks for the generalized context. With these clicks we estimate the CTRs. The parameter `min_support` specifies how many clicks are sufficient. Figure 2 shows a hierarchy H over the contexts with leaf nodes being the exact contexts and intermediate nodes being generalized contexts. It shows a cut-off through the hierarchy so that all (generalized) contexts above the cut-off have at least `min_support` many clicks in the training data for descendant contexts. The contexts below the threshold need to be generalized before estimates on the CTRs can be obtained.[7]

A Top-Down Algorithm. In order to compute privacy-preserving CTRs for the generalized contexts in the hierarchy H, algorithm TopDown starts at the root and moves down in the hierarchy. For each traversed node v and for each ad a, it estimates the $\mathsf{CTR}(a|v)$ by counting how often users in a descendant context of v have clicked (or only viewed) a and adding Laplacian noise with magnitude λ to the count. The results of this computation are referred to as $\texttt{clicks}_{a,v}$ ($\texttt{no\_clicks}_{a,v}$, resp.). The estimated click-through-rate is then simply $\widehat{\mathsf{CTR}}(a|v) = \frac{\texttt{clicks}_a}{\texttt{clicks}_{a,v}+\texttt{no\_clicks}_{a,v}}$. TopDown also computes the total number of times a descendant of v appears in the ad log and adds noise to this count. If the count is above `min_support` then the algorithm recurses on v's children, otherwise the children are pruned. We note that accuracy of Estimates can be further improved by using the post-processing techniques of Hay et al. to make sure the counts of all children add up to the parent's count [17]. In order to bound the sensitivity and to guarantee differential privacy, we limit the number of entries per user in the ad log. Estimates deletes from the ad log all but m entries per user and then calls TopDown.

We now analyze privacy and efficiency of our algorithm.

Proposition 5.4 (Efficiency): With high probability Estimates makes $O(|A| \cdot N \cdot m/\texttt{min\_support})$ calls to Count and is thus independent of the number of contexts. Moreover, when we use Count in Estimates we can batch all the messages within one level in the hierarchy.

In Estimates we employ Count to obtain noisy estimates $\texttt{clicks}_{a,v}, \texttt{no\_clicks}_{a,v}, \texttt{count}_v$. We consider all queries of one level in the context hierarchy to be a large histogram query with L_2-sensitivity of m and apply Theorem 5.1 to analyze the privacy. To go down in the hierarchy we use the composition property to analyze the privacy.

Corollary 5.5 (Privacy): Consider $\epsilon \leq 1$ and $\sigma^2 \geq m^2 2\log(2/\delta)/\epsilon^2$. When Estimates employs $\mathsf{Count}(t, \sigma^2)$ as a subroutine for counting $\texttt{clicks}_{a,v}$, $\texttt{no\_clicks}_{a,v}$, $\texttt{count}_v$, it guarantees $(\text{depth}(H)\cdot\epsilon, \text{depth}(H)\cdot\delta)$-probabilistic differential privacy. A fraction of t unavailable or malicious users during each call of $\mathsf{Count}(t, \sigma^2)$ can be tolerated.

[6]It is recommend but not required that users generalize the contexts they send to the server to a node in the hierarchy.

[7]Note, that there are other ways to define similarity, for example using the lattice structure imposed by the attributes' hierarchies. Our experimental results show only a minor effect on quality when using a fixed combined hierarchy as opposed to a lattice structure.

Algorithm 3 Top-Down computation of noisy statistics.

TopDown(context-driven ad log, node v in the hierarchy, noise scale λ, threshold `min_support`)
$\mathcal{A}' =$ set of ads with bids on context of v
for $a \in \mathcal{A}'$ **do**
  $\texttt{clicks}_{a,v} =$ Count (# of clicks on a in v in ad log)
  $\texttt{no\_clicks}_{a,v} =$ Count (# of views of a w/o clicks in v)
  release $\widehat{\mathsf{CTR}}(a|v) = \frac{\texttt{clicks}_{a,v}}{\texttt{clicks}_{a,v}+\texttt{no\_clicks}_{a,v}}$
$\texttt{count}_v =$ Count (# of appearances of node v appears)
release $\texttt{count}_v$
if $\texttt{count}_v > \texttt{min\_support}$ **then**
  for each child w of v **do**
    return TopDown(ad log, w , λ, `min_support`)

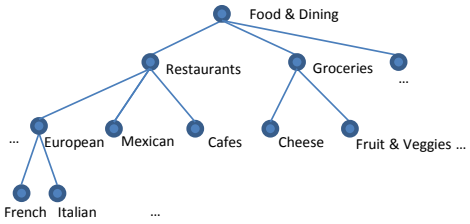


Fig. 3. Hierarchy over businesses.

VI. Experiments

A. Experimental Setup

Dataset. Ideally, we would like to evaluate our algorithms with real usage logs from a context-aware ad service. However, since no such real systems exist, we emulate such a system by using a log of location-aware searches in Microsoft Bing for mobile.[8] The log has the following schema: ⟨ *user-ID, query, user-location, business-ID* ⟩. Each record in the log describes an event of a user (with user-ID) issuing a query from a location and then clicking on a business. The log consists of total 1,519,307 records. In our evaluation we focus on clicks to "Food & Dining" business, which is the largest category of business in the log. We also filter out users with less than 3 clicks in the log, as we cannot generate an interest profile for such a user. We use the first 90% of the log as training data and the remaining log to compute targeted advertisements (i.e., businesses).

Context. We use the above log to emulate a context-aware ad service as follows. We assume that each business with id

[8]http://m.bing.com

i has an ad with the same id i, and hence our goal is to deliver target business-IDs to the users. Ideally, we would like to use context based on the sensor readings of smart phones for personalization. However, this information is not present in our log and we therefore limit our evaluation to contexts consisting of the following set of attributes.

► Location: The user's current location as latitude and longitude coordinates.

► User Interest: A history (i.e., a multi-set) of the business-IDs the user clicked in the past.

► Query: The search query the user sends.

Attribute Generalization. To limit information disclosure, we let users generalize context attributes according to fixed hierarchies.

► Location: We use five levels of generalization for user's location, depending on how many decimal points we truncate from her latitude and longitude. More specifically, `Level-i` location, $0 \leq i \leq 5$ of a user is her latitude and longitude, after keeping all, 4, 3, 2, 1, and 0 decimal points respectively.

► Interest: We generalize user interest using a fixed hierarchy for the businesses, as shown in Figure 3. In `Level-0`, `Level-1`, `Level-2`, the interest set contains business categories, generalized business categories, and only the most general business category ("Food and Dining"), respectively, of the user's clicks.

► Query: Like user interest, we use the business hierarchy to generalize query attribute into three possible levels. `Level-0` is the exact query issued by the user, `Level-1` is the business category of the clicked business, and `Level-2` is the generalized category of the business.

For all attributes, `Level-i` is more general, and hence more privacy preserving, than `Level-j`, for $i > j$. As a short-hand, we use (x, y, z) to denote (`Level-x` location gen., `Level-y` interest gen., `Level-z` query gen.).

Context Hierarchy. We combine the attribute hierarchies to a context hierarchy. To generalize a context, we generalize one attribute at a time. We use the following sequence to generalize one context to a more private context: $(0,0,0) \rightarrow (0,0,1) \rightarrow (0,1,1) \rightarrow (1,1,1) \rightarrow (1,2,1) \rightarrow (2,2,1) \rightarrow (3,2,1) \rightarrow (3,2,2) \rightarrow (4,2,2)$. As an example consider the context at level $(0,0,0)$

$\langle$(61.22913, -149.912044), [B-ID2011, B-ID124], "Starbucks"$\rangle$.

Generalizing each attribute one level yields at level $(1,1,1)$

$\langle$(61.2291, -149.9120), [Peruvian Restaurants, Wine], "Coffee"$\rangle$.

We show how generalization helps personalization with sparse data. Figure 4 shows the frequency distributions in log-log scale of queries (Figure 4(a)), and their generalizations (Figures 4(b) and (c)), in our dataset. The query distribution has a power-law shape which shows that a small fraction of unique queries account for a large fraction of the query log. We can see that roughly 100,000 queries appear only once in our dataset. For these queries it is impossible to personalize the search results because we have not seen the same query before. However, if we generalize queries to the categories of the businesses they are referring to we can reduce this number by an order of magnitude. Similarly, we can deal with the sparsity of the other context attributes by generalizing them. This is how we increase the coverage.

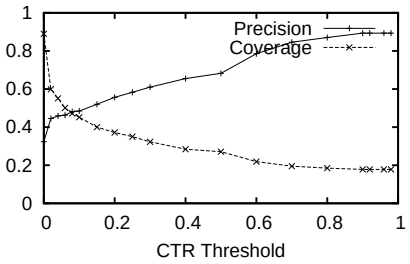


Fig. 5. Varying the minimum CTR.

Metrics. We use the following two metrics for our prediction.
► **Precision:** This the fraction of targeted ads computed by our framework on which the users actually click. Assuming that advertisers pay the same amount of money for each ad, revenue is proportional to precision.
► **Coverage:** This is the fraction of contexts in the training data for which our framework computes and displays a targeted business. The higher the precision and coverage values, the better the performance of our framework is. We report average precision and coverage for 1000 random contexts from the testing data; the averages become fairly stable after 1000 predictions.

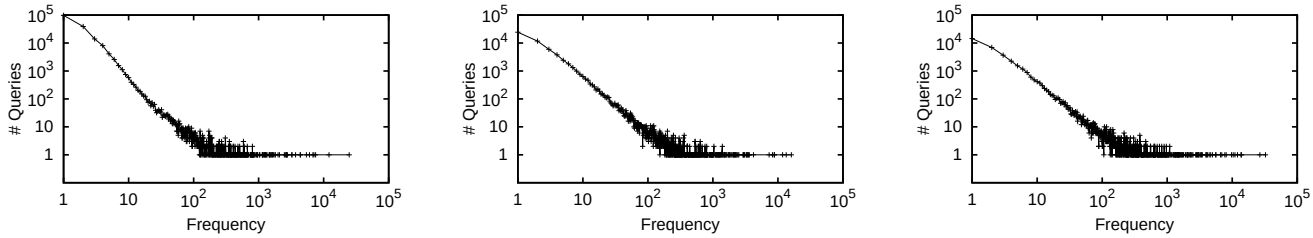
Parameters. Unless otherwise stated, we use the following default configuration. For limited information disclosure, we use $(4,2,2)$ generalization. We set the upper bound on communication complexity, k, to be 10, the threshold on click-through-rate to be 0.3, and the threshold on support to be 2.

B. Evaluating Tradeoffs

Effect of CTR Threshold. The CTR threshold introduces a tradeoff in precision and coverage of our overall prediction. Figure 5 shows this trade-off. For a high value of the CTR threshold, an ad will be shown only if it is highly relevant. Thus, this increases the precision of our algorithm and improves the relevance of the displayed ads. On the other hand, a high threshold reduces the number of ads being displayed and with that the number of clicks and the revenue. Interestingly, as we can see, *high levels of both precision (0.48) and coverage (0.47) can be achieved simultaneously*.[9]

Effect of Communication Complexity. Figure 6 shows the effect of increasing the communication complexity k (i.e. having the server return more ads to the client) on precision and coverage. We expect both to improve with increasing value of k since the client can choose an ad from a larger set. The

[9]Precisions and coverages close to 0.5 are considered high in predicting user clicks. Our numbers are higher than the ones reported for other personalization techniques [35].



(a) Exact query (b) `Level-1` generalization (c) `Level-2` generalization

Fig. 4. Effect of generalization of queries

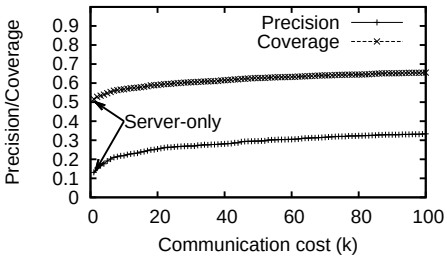
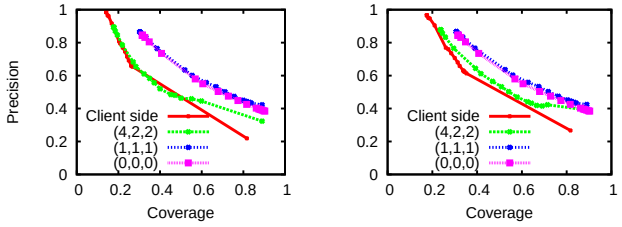
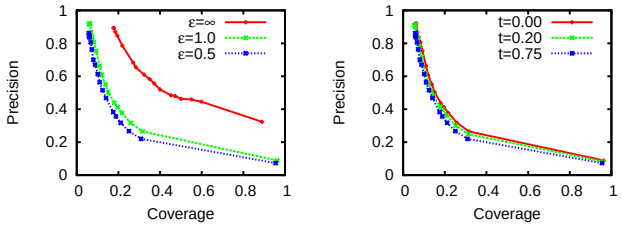


Fig. 6. Varying communication cost.



(a) $k = 10$ (b) $k = 50$

Fig. 7. Varying information disclosure.



(a) Effect of noise ϵ (b) Effect of robustness t

Fig. 8. Differentially-private estimates.

graph shows further that *increasing k has diminishing returns*, i.e., in the beginning the precision and coverage increase quickly with every additional ad being sent. However, with more and more ads being sent, the increase in precision and coverage becomes smaller and smaller.

Effect of Information Disclosure. Figure 7 shows the precision and coverage (for various CTR thresholds) of our framework with various levels of generalization. As expected, precision and coverage of our framework increases as more and more specific context information is sent to the server. Interestingly, *limited privacy does not hurt utility in a significant way*; as shown in the graph, precision and coverage values are very close for limited privacy (shown as $(1, 1, 1)$) and no privacy (shown as $(0, 0, 0)$).

Trading-off Constraints. To see how communication overhead affects the performance of our framework, we increase k from 10 to 50 in Figures 7(a) and (b). The graphs show that as we increase k from 10 to 50, the gaps between precision-coverage curves for various levels of information disclosure decreases. This shows that *privacy can be improved without hurting utility by a small increase in the communication cost*. For example, to have a precision and a coverage more than 0.8 and 0.3 respectively, a user needs to have no $(0, 0, 0)$ or very limited $(1, 1, 1)$ privacy when $k = 10$. However, she can improve her privacy level to $(4, 2, 2)$ and still have the same precision and coverage by increasing the value of k to 50. Overall we conclude from our experiments that reasonable levels of limited information disclosure, efficiency, and relevance can be achieved simultaneously.

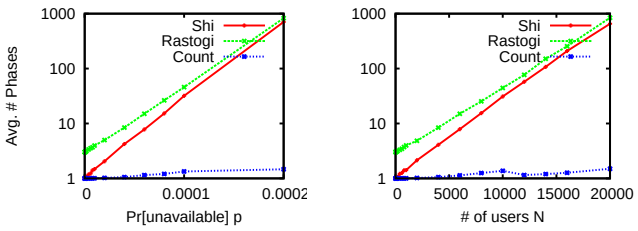
C. Comparison with Other Strategies

Server-only Personalization. Here, the server performs personalization based on the limited private information it has and sends one single ad to the client. As shown in Figure 6, this strategy gives a precision of 0.28. We can do much better with our optimization: When instead sending 5 ads and letting the client pick the most relevant one the precision rises by 35%.

Client-only Personalization. Here, the client sends only the query to the server, which then sends k ads matching the query to the client. The client then chooses the best ad based on the exact user context. Precision and coverage of this strategy are also shown in Figure 7 with the label "Client-side". As shown, our optimization can provide better utility that the client-only strategy. For example, for a target precision of 0.75, the client-side strategy can achieve a coverage of 0.2, while our framework with $(1, 1, 1)$ generalization can achieve a 0.4 coverage, an increase by $2\times$.

D. Privacy-Preserving CTRs

In the following experiments, we fix the number of contributions per user $m = 4$. Moreover, we found that it is beneficial



(a) $N = 10{,}000$ (b) $p = 0.0001$

Fig. 9. Varying user population.

to introduce a limit as towards how far TopDown goes down in the hierarchy. This improves the privacy guarantee by changing the factor depth(H) to this limit. The results here present a very aggressive limit of 1 which we found adequate for such a small set of training data.

Efficiency. When we run Estimates on our log, a user has to send roughly 1MB on average. Many of the count queries can be batched: On average a user participates in 2 batches. This communication cost is acceptable.

Accuracy. Figure 8(a) shows how precision and coverage of our framework degrades when we increase the differential privacy guarantee (by decreasing ϵ). We fixed $\delta = 0.01$. As a point of comparison the figure also draws the precision and coverage curve when using the exact, non-private statistics ($\epsilon = \infty$). We can see that when we want to achieve a precision of 0.6 then the coverage of our framework using the non-private statistics is much higher than the coverage of our framework using the ϵ-differentially private statistics (i.e., 0.3 vs 0.1). This is the price we have to pay for a privacy guarantee. The exact value of the privacy parameter ϵ (1 vs. 0.5) has a minor effect on the utility. We expect the cost of privacy to decrease with a larger user population. Moreover, we can avoid such negative impact on utility by paying the price of privacy in terms of communication overhead k—as shown in Figure 7, the utility can be improved by using a higher value of k.

Robustness. Figure 8(b) shows the effect of varying t (fraction of malicious/unavailable users) on precision and coverage for $\epsilon = 1.0$. We see that the parameter t only mildly affects the utility. Even when half the users could be unavailable or malicious (i.e., $t = 0.5$) the utility is almost the same as when all users are guaranteed to be available and honest.

To compare the robustness of our Count protocol with existing works, we model users' unavailability as a simple random process. Suppose a *phase* denotes the time it takes for the server to send a message to all users or for all users to send messages to the server. Let p denote the probability that a user is available to participate in our protocol in a given phase. We compare various protocols in terms of the average number of phases required to complete a query, as it indicates the latency and communication complexity of a protocol.

Figure 9 illustrates the effects of unavailability on communication complexity. We compare our algorithm with two existing protocols: RASTOGI [29] and SHI [31]. We run 1000 queries and report the average number of phases per query for different protocols. As shown, all the protocols cost close to their optimal number of phases when the probability of being unavailable (p) and the number of users (N) are small. However, *unlike our protocol, the costs for* SHI *and* RASTOGI *increase exponentially with* N *and* p (note the log scale of the graphs). For $p \geq 0.0001$ (corresponding to less than only 10 seconds a day) in (a) or $N \geq 1000$ (much fewer than users of popular online services) in (b) the protocols become impractical. This shows that SHI and RASTOGI are impractical for online services with dynamic users, while our protocol is efficient for them.

VII. CONCLUSION

We have addressed the problem of personalizing ad delivery to a smart phone, without violating user privacy. We proposed a flexible framework with tuning knobs on privacy, efficiency, and utility. We showed that the problem of selecting the most relevant ads under constraints on privacy and efficiency is NP-hard and proposed a solution with a tight approximation guarantee. We also proposed the first differentially-private distributed protocol to compute various statistics required for our framework even in the presence of a dynamic and malicious set of participants. Our experiments on real click logs showed that reasonable levels of privacy, efficiency, and ad relevance can be achieved simultaneously.

REFERENCES

[1] Gagan Aggarwal, Mayank Bawa, Prasanna Ganesan, Hector Garcia-Molina, Krishnaram Kenthapadi, Rajeev Motwani, Utkarsh Srivastava, Dilys Thomas, and Ying Xu 0002. Two can keep a secret: A distributed architecture for secure database services. In *CIDR*, 2005.

[2] Thorben Burghardt, Klemens Böhm, Achim Guttmann, and Chris Clifton. Anonymous search histories featuring personalized advertisement - balancing privacy with economic interests. *Transactions on Data Privacy*, 4(1):31–50, 2011.

[3] Georg Buscher, Susan T. Dumais, and Edward Cutrell. The good, the bad, and the random: an eye-tracking study of ad quality in web search. In *SIGIR*, 2010.

[4] Bee-Chung Chen, Daniel Kifer, Kristen LeFevre, and Ashwin Machanavajjhala. Privacy-preserving data publishing. *Foundations and Trends in Databases*, 2(1-2):1–167, 2009.

[5] Maria Luisa Damiani, Elisa Bertino, and Claudio Silvestri. The probe framework for the personalized cloaking of private locations. *Transactions on Data Privacy*, 3(2):123–148, 2010.

[6] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, volume 4004, 2006.

[7] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[8] Uriel Feige. A threshold of ln n for approximating set cover. *JOURNAL OF THE ACM*, 45:314–318, 1998.

[9] Matthew Fredrikson and Benjamin Livshits. Repriv: Re-envisioning in-browser privacy. Technical Report MSR-TR-2010-116, Microsoft Research, August 2010.

[10] William Gasarch. A survey on private information retrieval. *Bulletin of the EATCS*, 82:72–107, 2004.

[11] Gabriel Ghinita. Private queries and trajectory anonymization: a dual perspective on location privacy. *Transactions on Data Privacy*, 2(1):3–19, 2009.

[12] Gabriel Ghinita, Maria Luisa Damiani, Claudio Silvestri, and Elisa Bertino. Preventing velocity-based linkage attacks in location-aware applications. In *GIS*, 2009.

[13] Michaela Götz, Ashwin Machanavajjhala, Guozhang Wang, Xiaokui Xiao, and Johannes Gehrke. Publishing search logs - a comparative study of privacy guarantees. *TKDE*, 99(PrePrints), 2011.

[14] Saikat Guha, Bin Cheng, and Paul Francis. Challenges in Measuring Online Advertising Systems. In *Proceedings of Internet Measurement Conference (IMC)*, Melbourne, Australia, Nov 2010.

[15] Saikat Guha, Alexey Reznichenko, Kevin Tang, Hamed Haddadi, and Paul Francis. Serving Ads from localhost for Performance, Privacy, and Profit. In *HotNets*, 2009.

[16] Moritz Hardt and Guy Rothblum. A multiplicative weights mechanism for interactive privacy-preserving data analysis. In *FOCS*, 2010.

[17] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 3(1):1021–1032, 2010.

[18] Dorit Hochbaum and Anu Pathria. Analysis of the Greedy Approach in Problems of Maximum k-Coverage. *Naval Research Logistics*, 45(6):615–627, 1998.

[19] Christian S. Jensen, Hua Lu, and Man Lung Yiu. *Location Privacy Techniques in Client-Server Architectures*, pages 31–58. 2009.

[20] Ari Juels. Targeted advertising ... and privacy too. In *CT-RSA*, 2001.

[21] Aleksandra Korolova. Privacy violations using microtargeted ads: A case study. In *Workshop on Privacy Aspects of Data Mining (PADM)*, 2010.

[22] Andreas Krause and Eric Horvitz. A utility-theoretic approach to privacy and personalization. In *AAAI*, 2008.

[23] Ashwin Machanavajjhala, Daniel Kifer, John M. Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: Theory meets practice on the map. In *ICDE*, 2008.

[24] Frank McSherry and Ratul Mahajan. Differentially-private network trace analysis. In *SIGCOMM*, 2010.

[25] Frank McSherry and Ilya Mironov. Differentially private recommender systems: building privacy into the net. In *KDD*, 2009.

[26] Emiliano Miluzzo, Cory T. Cornelius, Ashwin Ramaswamy, Tanzeem Choudhury, Zhigang Liu, and Andrew T. Campbell. Darwin phones: The evolution of sensing and inference on mobile phones. In *ACM MobiSys*, 2010.

[27] Mohamed F. Mokbel, Chi-Yin Chow, and Walid G. Aref. The new casper: A privacy-aware location-based database server. In *ICDE*, 2007.

[28] Linda Pareschi, Daniele Riboni, Alessandra Agostini, and Claudio Bettini. Composition and generalization of context data for privacy preservation. In *PerComm*, 2008.

[29] Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *SIGMOD Conference*, 2010.

[30] Xuehua Shen, Bin Tan, and ChengXiang Zhai. Implicit user modeling for personalized search. In *CIKM*, 2005.

[31] Elaine Shi, T-H. Hubert Chan, Eleanor Rieffel, Richard Chow, and Dawn Song. Privacy-preserving aggregation of time-series data. In *NDSS*, 2011.

[32] Vincent Toubiana, Helen Nissenbaum, Arvind Narayanan, Solon Barocas, and Dan Boneh. Adnostic: Privacy preserving targeted advertising. In *NDSS*, 2010.

[33] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. Differential privacy via wavelet transforms. In *ICDE*, 2010.

[34] Yabo Xu, Ke Wang, Benyu Zhang, and Zheng Chen. Privacy-enhancing personalized web search. In *WWW*, 2007.

[35] Jun Yan, Ning Liu, Gang Wang, Wen Zhang, Yun Jiang, and Zheng Chen. How much can behavioral targeting help online advertising? In *WWW*, 2009.