Auctions in Private Internet Advertising

Alexey Reznichenko, Saikat Guha, Paul Francis
MPI-SWS, Microsoft Research and MPI-SWS
saikat@microsoft.com, {areznich,francis}@mpi-sws.org

ABSTRACT

Several recent research projects have designed systems that address the problem of user privacy in online advertising systems. The primary goal of these systems is to allow for targeted advertising without revealing user profiles to the ad network. None of these designs, however, adequately consider the role of the auction. This paper looks at the problem of running auctions that leverage user profiles for ad ranking while keeping the user profile private. We define the problem, broadly explore the solution space, and describe the pros and cons of these solutions in light of the proposed private advertising systems. We analyze the performance of our solutions using data from Microsoft Bing advertising auctions. We conclude that, while none of our auctions are ideal in all respects, they appear to be quite feasible in practice.

1. INTRODUCTION

Online advertising is a key economic driver in the Internet economy, funding a wide variety of websites and services. Internet *advertisers* increasingly wish to provide more personalized advertising. Unfortunately, personalized online advertising comes at the price of individual privacy [11]. To address this problem, recently a number of researchers have proposed systems that can do personalized advertising while protecting user privacy [10, 12, 14]. None of these systems, however, specify how to operate the advertising auction in a way that exploits the personalization. This leaves unanswered what revenue the *broker* (i.e. an ad network like Google) can earn, thereby reducing the likelihood that a private advertising system will be of commercial interest. This paper explores the problem of running auctions in private advertising systems that exploit personalization information while maintaining user privacy.

Although the private advertising systems cited above differ in many respects, they all share the following key design components: personalization information is stored in a *user profile* kept at the user's computer (the *client*). Multiple ads are transmitted to the client, not all of which match the user profile. The client then selects from among these ads which to display to the user.

What does this have to do with auctions? The most common pricing model for online advertising systems today is Pay Per Click (PPC): the advertiser does not pay the broker for showing an ad to a user, rather it pays only if the user clicks on an ad. The broker selects which ads to show through an auction whereby advertisers bid against each other. In a PPC system, the broker maximizes revenue by ranking the competing ads according to the $Bid \times ClickProbability$ product, and transmitting the highest ranking ads to the client where they are displayed in rank order. Of course, the broker doesn't know the precise click probability for every ad. Rather, the broker tries to predict the click probability as best it can. This prediction is based on a number of inter-related factors such as the ad keywords, the landing page keywords, the user search terms or keywords associated with the web-page being browsed, stored user characteristics, and so on. Microsoft incorporates at least seven and perhaps many more such factors in its Bing search advertising auctions.

The user profile has an effect on click probability. To give a simple example, say a user searches for "running shoe". Whether the user is a man or a woman, or prefers brand-name products or discount products, plays an important role in which running shoe ad he or she is more likely to click on. In a private advertising system, the broker does not know the user profile: if the auction takes place at the broker in the same way that it does today, then the user profile will not be factored into the result. Therefore the highest $Bid \times ClickProbability$ ads won't be selected, leading to less revenue than would otherwise be possible.

This paper proposes three basic solutions to this problem, and in part using auction traces from Microsoft Bing, and explores their pros and cons in terms of privacy (both user and advertiser), revenue, overhead, and vulnerability to attack. The three approaches are *rank-at-client*, *rank-at-broker*, and *rank-at-3rd-party*. The second approach was previously partially described in a tech report describing the Privad advertising system [9].

Altogether, this paper makes the following contributions:

- It is, to our knowledge, the first paper to explore the problem of user privacy in online advertising auctions.
- It explores the solution space, provides three basic solutions and several variants, and analyzes the tradeoffs between them in terms of privacy properties, auction properties, and click-fraud.

- It analyzes the effect of bid churn and auction timing on revenue and ad ranking using a trace of Bing search advertizing auctions, and uses this analysis to argue for the feasibility of the solutions.

2. BACKGROUND

This section describes how current online advertising systems such as Google and Microsoft work, and then describes the three currently proposed alternatives for private advertising. In the process, we establish terminology and define the basic components.

2.1 Current Ad Systems

In current ad systems [2, 7, 13], *advertisers* submit *ads* to a *broker*. Associated with each ad is a *bid*, one or more *keywords*, and optionally some targeting information like demographics (location, age, gender) or interests. When a *client* computer does a search or receives a web page with *adboxes* (space to place an ad), the broker identifies the ads that match the search terms or keywords associated with the web page, and runs an auction. The auction ranks the selected ads in order of highest expected revenue ($Bid \times ClickProbability$), and transmits some number of ads to the client. As already discussed, many factors are considered in estimating click probability. In this paper, we refer to all of these factors taken together as a *quality score* Q, where a higher value means higher expected click probability. Denoting bid as B, the ranking then is in order of the product ($B \times Q$).

When a user clicks on an ad, the ad ID is transmitted to the broker. The broker computes Cost per Click (CPC), that is, the price that the advertiser must pay, using a *second-price auction* [3]. In this approach, the price paid is pegged to the bid of the ad ranked immediately below the ad clicked. To give a simplistic example, suppose that advertiser A is willing to pay as much as $5 for a click, and advertiser B is willing to pay $10. In a second-price auction, A could go ahead and bid $5 and B could bid $10. B would win, but would only pay incrementally more than A's (2nd-price) bid, say $5.01.

In general, second-price auctions allow bidders to bid the maximum that they are willing to pay, rather than frequently modify their bid in search of the value incrementally higher than the next lower bidder. Specifically, the CPC is computed as:

$$CPC = B_n \left(\frac{Q_n}{Q_c} \right)$$

where B_n and Q_n are the bid and quality score of the next lower ranked ad, and Q_c is the quality score of clicked ad. This CPC formula captures the minimum amount the advertiser would have had to bid to beat the next-ranked ad in a first-price auction. Note that it prevents the advertiser from paying more than it bid, even when the next-ranked in fact bid more.

2.2 Private Ad Systems

This section describes three proposed private ad systems, Adnostic [14], Nurikabe [12], and Privad [10]. For the sake of brevity, we limit our descriptions here to those aspects of these systems that pertain to the auction problem. While these do not represent the complete possible design space for private advertising systems, they differ in significant respects and so offer a rich set of examples against which to evaluate private auction systems.

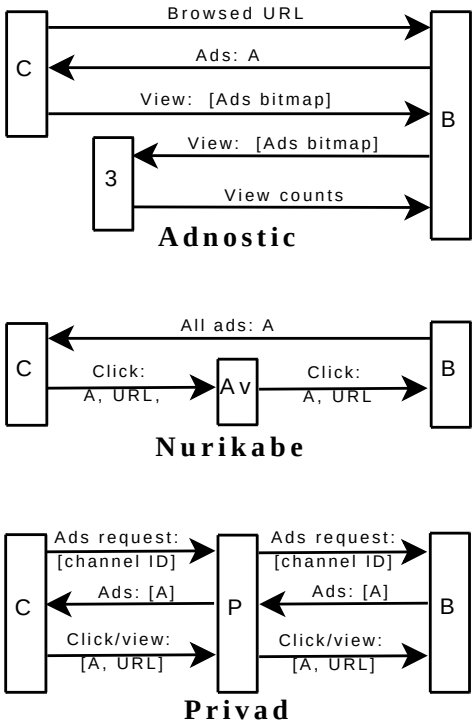

Figure 1: The three private advertising proposals. B is the broker, C is the client, P is the proxy, Av is the advertiser, and 3 is a third party system. [x] denotes encryption of x.

All three systems propose a *profiler* that runs at the client and generates a user profile. All three systems at least share the privacy goal that this profile not be revealed. All three systems propose that multiple ads be transmitted to the client, and that the client selects from among these ads which to show to the user. As Table 1 shows, the three systems differ in terms of which ads are sent to the client, when the ads are sent, and how the views and clicks are anonymized to protect user privacy. In addition, Nurikabe and Privad have stronger privacy goals than Adnostic. Adnostic hides only which ads have been viewed (besides hiding the user profile). Nurikabe and Privad additionally hide which pages the user has visited as well as the user's clicks.

Nurikabe and Privad both deliver ads to the client in advance of when adboxes arrive at the client (Figure 1). The ads are stored at the client, and placed in adboxes when the adboxes arrive. In the case of Nurikabe, all ads are transmitted to the client. As a result, the Nurikabe broker can learn nothing about the client in the act of delivering ads. Privad argues that sending all ads doesn't scale, and so allows the client to subscribe to anonymous pub-sub channels defined by a single interest, rough location, and language (for instance, running shoes, Brooklyn area, Spanish speaking). Privad argues that as long as there are many members of each such channel, the privacy of any individual member is not compromised. In order to prevent the Privad broker from correlating multiple channels across the same user, a proxy hides the client's network address from the broker. Encryption prevents the proxy from learning the clients' channels.

	Adnostic	Nurikabe	Privad
Which ads?	Those related to a browsed web page	All	Those related to an interest + location + language + gender channel
When?	At adbox arrival	Before adbox	Before adbox
Ad delivery anonymization	None	None needed	Via proxy
View anonymization	3rd-party using homomorphic crypto	None needed	Via proxy
Click anonymization	None	Via advertiser	Via proxy
User privacy (from Broker)	User profile, viewed ads	User profile, pages browsed, viewed and clicked ads	

Table 1: Characteristics of three proposed private advertising systems

Like current advertising systems, Adnostic delivers ads to the user when an adbox arrives. The URL of the page containing the adbox, or if search advertising, the search terms, are transmitted to the broker. Unlike current advertising systems, which would look at the user's profile and deliver only the ads that best match the user, Adnostic sends many ads across multiple demographics (for instance, 30 ads for each adbox).

Adnostic and Privad report views and clicks, while Nurikabe reports only clicks. Privad reports them anonymously through a proxy (encrypted so that the proxy does not learn the contents of the reports). Nurikabe reports clicks through the advertiser itself. In both cases, the broker does not learn anything about individual clients (as identified by their network address). Adnostic does not protect click reports — the broker learns what the user clicks on. Adnostic protects view reports by encrypting which of the multiple delivered ads were viewed, and having a 3rd-party system count the number of views for every given ad using homomorphic encryption.

3. GOALS

The primary goal of a private auction system is to achieve the $Bid \times ClickProbability$ ideal ranking and second-price auction, while fully leveraging the information in the user profile, and maintaining the privacy of the user profile. For today's centralized broker models, leveraging the user profile is straightforward, since the broker itself maintains this information, but it is of course not private. In a private advertising system, the broker does not have this information. In particular, we define a third auction component, the *user score* U, as the user information that is available to the client and not available to the broker.

We therefore define our ideal ranking and CPC functions as:

$$Rank \Rightarrow B \times Q \times U \tag{1}$$

$$CPC = B_n(\frac{Q_n \times U_n}{Q_c \times U_c}) \tag{2}$$

In the ideal case, the B, Q, and U used for computing CPC are the same as those used for ranking, and furthermore are the current values as of the time of the click. This is not possible for Nurikabe and Privad, where ads are sent to the client in advance of when they are displayed in adboxes. Therefore, we set the following goals with respect to rank and CPC:

- The B, Q, and U used for CPC calculation are the same as the B, Q, and U used for ranking. Note in particular that if they are not the same, then it is possible for instance for the CPC to be higher than the bid of the clicked ad.

- The delay between ranking and CPC calculation is small enough that the churn in B, Q, or U does not have a significant impact on rankings, CPC values, and broker revenue.

What information constitutes user score U and what information constitutes quality score Q depends on the privacy goals of the private advertising system itself. On one extreme, Nurikabe reveals nothing to the broker about the client except its IP address (and therefore broad location, although by design this is ignored). On the other extreme, Adnostic tells the broker which web page is being browsed and broad location (again via the IP address). Privad tells the broker a single interest and broad demographics like broad location and language. This information provided to the broker can be used along with other information available to the broker to derive the quality score Q.

What comprises user score U is an open question and depends on the client profiler. None of the three approaches nail this down in any detail, and in any event how the profiler operates can evolve over time. We can, however, classify user information into three time frames. At the time frame of months or even years are user demographics like gender, location, language, age, salary, and so on. User interests can also last years (e.g. coin collecting), but more typically last weeks (a new car), days (a new pair of shoes), or minutes (a pizza). If we assume that matching ads to the content of a web page or search page increases click probability, then user score can change in seconds or less. For instance, a user might be interested in tennis and music, but the user score for tennis ads may increase while the user is looking at a tennis website, and vice versa for music ads.

We do not make any assumptions about the relative importance of B, Q, or U. An ideal auction design allows for this flexibility.

Besides the basic goal of running an auction that leverages the user profile while maintaining user privacy, there are a few additional related goals that are important:

- to maintain the privacy offered to the advertisers themselves. In particular, to prevent advertisers from learning each other's bids and budgets.

- to maintain the level of click-fraud defense established by each advertising system.

- to minimize the overhead of the auction.

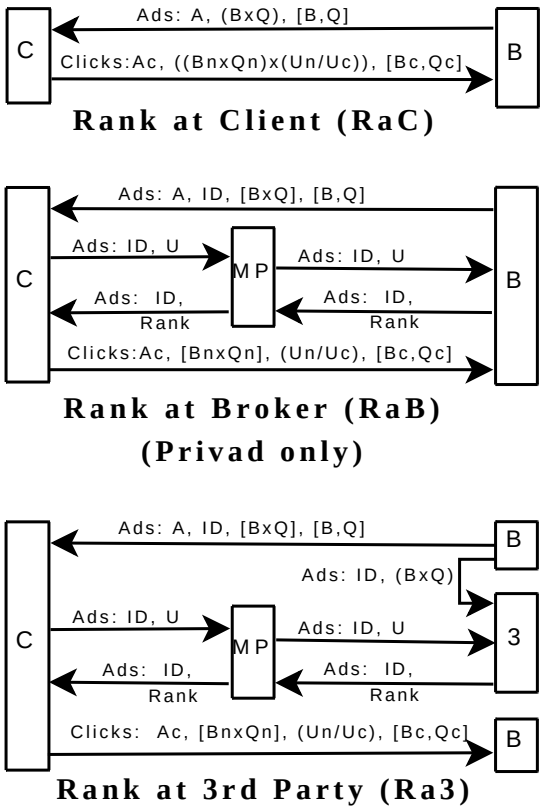

Figure 2: Three basic auction schemes. The device labeled MP is a mixing proxy. A is an ad ID unique to the ad, while ID is an Ad ID unique to the combination of ad and client. The subscripts 'c' and 'n' refer to the clicked ad and the next ranked ad respectively. [x] denotes symmetric key encryption of x such that neither the client nor the MP can decrypt. Though not shown here, the ads and the click reports would be protected (or not) as specified by the private advertising approach (Figure 1).

As will become apparent in subsequent sections, our designs do not perfectly achieve all of these goals. Rather, our designs offer tradeoffs between these goals. Furthermore, the tradeoffs differ for the different private advertising systems.

4. DETAILED DESIGNS

To run the auction specified in Section 3, the system doing the ranking must have access to the bid B, the quality score Q, and the user score U. This means that either 1) B and Q are sent to the client, 2) U is sent to the broker, or 3) B, Q, and U are all sent to a 3rd party (Figure 2).

4.1 Rank-at-Client (RaC)

In this approach, the following information is transmitted along with the ad to the client[1] along with everything else required by the advertising system (e.g. targeting information, not shown):

A: The ad ID.

$(B \times Q)$: A single value which is the product of $(B \times Q)$.

$[B, Q]$: The values B and Q, encrypted with a symmetric key known only to the broker.

When a collection of ads arrive at the client, it ranks all ads using $(B \times Q \times U)$. Note that in this case U is current, while $(B \times Q)$ is out of date. If the user clicks on an ad, then the client computes the following values and transmits them to the broker:

A_c: The ad ID of the clicked ad.

$((B_n \times Q_n) \times (U_n/U_c))$: A single value which is the $(B_n \times Q_n)$ product of the next-ranked ad times the ratio (U_n/U_c) of the user score of the next-ranked ad U_n and the user score of the clicked ad U_c.

$[B_c,Q_c]$: The encrypted B and Q for the clicked ad as earlier received from the broker.

Upon reception of this message, the broker decrypts $[B_c,Q_c]$. It uses Q_c and $((B_n \times Q_n) \times (U_n/U_c))$ to compute the CPC as shown in Equation 2. The broker also compares the resulting CPC with the decrypted B_c. If $CPC > B_c$, then the broker knows that the client is engaged in click fraud, and the broker can ignore the message. If $CPC \leq B_c$, then the broker can accept the message, although this doesn't mean that the client is not engaged in click-fraud. Other mechanisms, such as statistical analysis, must be used to detect it as is done today.

Variation: It may not be necessary to transmit the encrypted values $[B_c,Q_c]$. This is because B_c and Q_c can be looked up using the ad ID A_c. The danger here is that the looked up values may be very different from the other B and Q values used to compute CPC. This, however, to a large extend depends on the level of churn in B and Q values, which we found to be meager in the Bing auction trace. Therefore, it may well suffice to use looked-up values rather than values stored along with the ad at the client. Note that this variation applies to RaB and Ra3 as well.

4.2 Rank-at-Broker (RaB), (Privad only)

The RaB scheme presented here is essentially the approach proposed by Privad in [9], and indeed is only suitable for Privad. We present it here for completeness.

RaB requires a new component, the mixing proxy. The mixing proxy may be the same device as the proxy shown in Figure 1.

Along with the ad, the broker transmits the following to the client:

A: The ad ID.

ID: An identifier unique to this specific transmission of this ad (among all other transmissions). In other words, the same ad transmitted to other clients would have a different value of ID.

$[B \times Q]$: A single value which is the product of $(B \times Q)$, encrypted with a symmetric key known only to the broker.

$[B, Q]$: The values B and Q, encrypted with a symmetric key at the broker which is unknown to the client.

The client computes a user score U for each ad (in the absense of knowledge of what web page the ad may be shown

[1]Note that in the case of Privad, the message would be transmitted through a proxy which is not shown in Figure 2 for the sake of simplicity. We assume that the private advertising components from Figure 1 are used in addition to those shown in Figure 2.

on). In order to obscure the user profile, the client assigns a random value for U for those ads for which the client has a very low user score (for instance because the demographic doesn't match that of the user).

Clients transmit the ID and U tuples to a mixing proxy. The mixing proxy is trusted not to collude with the broker. The proxy mixes the tuples of multiple clients before passing them on to the broker. The mixing proxy also remembers which IDs were received from which clients.

For each received ID, the broker looks up the current values of B and Q. It then uses B, Q, and U to rank a large number of ads (say, those received over the last hour), and associates a rank number $Rank$ to each ad. The broker transmits the ID,$Rank$ tuple back to the mixing proxy.

The mixing proxy looks up which client is associated with the ID, and forwards the message on to the client. The client disregards the ads related to the low user scores, and uses the remaining ranking for selecting ads to put in ad boxes.

When a user clicks on an ad, it transmits the following information to the broker:

A_c: The ad ID of the clicked ad.

$[B_n \times Q_n]$: The encrypted $(B \times Q)$ for the next-ranked ad received earlier.

(U_n/U_c): A single value which is the ratio of the user score of the next-ranked ad U_n and the user score of the clicked ad U_c.

$[B_c,Q_c]$: The encrypted B and Q for the clicked ad received earlier.

Upon reception of this message, the broker decrypts $[B_n \times Q_n]$ and $[B_c,Q_c]$. It uses Q_c, B_n, Q_n, and (U_n/U_c)) to compute the CPC as shown in Equation 2. As with RaC, the broker also compares the resulting CPC with the decrypted B_c for click fraud.

The mixing proxy (combined with the normal Privad proxy shown in Figure 1) is meant to prevent the broker from knowing the identity of the client whose ads are being ranked. The per-client-per-ad unique ID is meant to prevent the mixing proxy, which does know the client identity (network address), from knowing which ad, and therefore what targeting information, is being referred to.

The reason that RaB is not suitable for Adnostic or Nurikabe is because, since the broker directly transmits ads to the client, the broker knows the identity of the clients from which user scores are received. Even with the noise added to low user scores, we are concerned that the non-noise user scores can be interpreted at the broker as a kind of fingerprint over the set of ads (i.e., ads targeted to men should have uniformly higher scores for men, and lower scores for women). In this way, the broker could tease out the profile of users. The alternative would be to introduce a Privad-style proxy to Adnostic and Nurikabe, but this then changes the design of those schemes, which is not our intent.

4.3 Rank-at-3rd-Party (Ra3)

This approach is similar to RaB, but can work for all three private advertising systems. The main difference between Ra3 and RaB is that in Ra3, the broker additionally sends the unique ad IDs and $(B \times Q)$ products to a third party system which is trusted not to collude with the broker. This 3rd party also receives the user scores from the clients, and based on this information, ranks ads in the same way the broker does in the RaB approach.

4.4 Homomorphic Encryption Variant (RaC, RaB, and Ra3)

A variation on all three auction designs is to use homomorphic encryption (e.g., ElGamal [4]), which allows for multiplication operations on encrypted data. This may be used to defend against certain attacks by the broker as described in Section 5.3.3. When a user clicks on an ad, the client encrypts (U_n/U_c) with the broker's public key. In the case of RaC, it also encrypts $(B_n \times Q_n)$ with the broker's public key. In the case of RaB and Ra3, the broker provides the encrypted $[B_n \times Q_n]$, but using its public key instead of a symmetric key. For all three schemes, the broker provides $[1/Q_c]$, again encrypted with the broker's public key. Using homomorphic multiplication, the client calculates:

$$[B_n \times Q_n] \times \left[\frac{U_n}{U_c}\right] \times [1/Q_c] = \left[B_n \left(\frac{Q_n \times U_n}{Q_c \times U_c}\right)\right]$$

and transmits the resulting value in the click report. Upon receiving a click report, broker decrypts the value to obtain the CPC. Although homomorphic encryption is expensive, there is no need to do the operations in real-time. Rather, the client can do the operations when it has spare CPU cycles before transmitting, and the broker can likewise run the operations later on as batch processing.

5. AUCTION ANALYSIS

This section analyzes the three types of auctions in terms of privacy, auction quality, and attacks on the auction systems.

5.1 Privacy Properties

In this section, we discuss the various shortcomings of each of the approaches with respect to privacy properties, on the assumption that the components are "honest but curious". In other words, they will operate as defined, but they will exploit any information that they gain through this correct operation.

5.1.1 *Broker sees* $((B_n \times Q_n) \times (U_n/U_c))$ *(RaC)*

As far as we can tell, exposing this value does not allow the broker to reverse-engineer user profiles. This is certainly the case for Nurikabe and Privad, where the next-ranked ad can be virtually anything. For Adnostic, the next-ranked ad is necessarily one of those just transmitted to the client, but as long as the ratio (U_n/U_c) is a fine-grained number for which an extremely large number of values may be generated, the Adnostic broker cannot confidently guess $(B_n \times Q_n)$, and therefore cannot guess the second-ranked ad[2].

5.1.2 *Client sees* $(B \times Q)$ *(RaC)*

In transmitting the product $(B \times Q)$ in the RaC approach, information is revealed about the quality score Q assigned by the broker to the ad. In particular, since the advertiser knows its bid B, and can receive the ad by simply running

[2]Note that the ability to link the clicked ad with a known client will, in our opinion, allow the broker to reverse-engineer the contents of the user profile to a greater or lesser extent. Our point here is that the RaC auction does not make this any worse.

a client, the advertiser learns Q. Whether this is a problem needs to be decided by the broker, though we point out that today Google reveals a course-grained quality score to its advertising customers.

Transmitting the product $(B \times Q)$ to the client also reveals the overall ranking of an ad to anyone running a client, including the advertiser's competitors. It is not clear how much of a problem this is, given that in today's advertising systems, an advertiser can see how its competitors rank relative to itself simply by observing how ads are displayed. RaC makes it easier and cheaper to obtain this ranking information, but does not materially change an advertiser's ability to do so.

If exposing the product $(B \times Q)$ to the client is an acceptable privacy loss, then RaC should be the preferred auction method. If it is not acceptable, then RaB and Ra3, which both avoid exposing this information, may be preferred.

5.1.3 Broker sees (U_n/U_c) (RaB and Ra3)

The reason the ratio (U_n/U_c) is exposed, rather than the individual values U_n and U_c, is because knowledge of U_c, when coupled with the targeting information in the associated clicked ad, would reveal additional user profile information [3]. This is particularly important for Adnostic and Nurikabe, where the broker and advertiser respectively know which client clicked. Some care must be taken in the creation of this ratio to prevent reverse engineering the separate values of U_n and U_c. In particular, if the ratio (U_n/U_c) is very fine-grained (specified to many decimal places) relative to the separate values, then the individual values can more easily be deduced from the separate values (i.e. if the ratio is known to be 0.2706131, and the individual values are integers between 1 and 500, then we know that the individual scores are 128/473). We therefore require that the U used for ranking be very fine-grained (4 or 5 digit accuracy), while the reported ratio be less fine-grained (say 2-digit accuracy, rounded to the nearest value). This minimizes the cases where the individual values can be deduced.

5.2 Auction Properties

In this section, we discuss the various shortcomings of each of the approaches with respect to the auction properties, especially ranking results and revenue.

5.2.1 System delays

For Nurikabe and Privad, there are delays in the system that can change both the rankings and the computed CPC. With RaC, there is a delay between when the ad was transmitted and when the ranking takes place (at adbox time). With RaB and Ra3, there is a delay between when the ranking occurs and when the ranking is actually used (at adbox time). In either case, the bid B or the quality score Q used for ranking may no longer be correct, and an out-of-date ranking takes place.

During the design of the auction approaches, these delays were a major concern. As it turns out, at least for the auction data from Bing search advertising auctions (Section 6), the delays have only a minor impact on both broker revenue and advertiser costs, even when the delay is several hours or a day. Nevertheless, this may not be the case for other systems or future systems, and so it remains important that these delays are minimized through good engineering.

[3]Of course, the simple act of clicking even without U_c reveals some profile information to the advertiser [8], although this can be mitigated in some cases by placing a proxy between the client and the advertiser.

5.2.2 Ad pre-selection

Recall from Section 2.2 that in all three private advertising schemes, more ads are transmitted to the client than will be displayed, and the client selects from among these ads. Transmitting these ads to the client necessarily requires that the broker select from among its universe of ads those that will be transmitted. In this "pre-selection" process, the broker cannot consider user score U, and so may select a non-optimal set of ads, resulting in lower CTR overall compared to where the broker has complete information.

Several criteria are used to make this initial pre-selection. For all three schemes, the broker does not want to select ads for advertisers whose budget has expired or has a high chance of expiring if the selected ad is clicked. In the case of Adnostic, since ads are pre-selected at adbox-time, the broker need only check to see if the budget of each ad has expired at that moment in time. In the case of Nurikabe and Privad, since ads are transmitted in advance, the broker must estimate the number of clicks that are likely to result from some number of transmissions. This can be done for the most part by measuring past click-to-transmission ratios. In any event, the broker must send a given ad to some fraction of clients that could otherwise legitimately receive the ad. For instance, say that the broker estimates that an advertiser's budget will expire if the ad is sent to 50% of the clients (all clients in the case of Nurikabe, clients that have subscribed to the associated interest channels in the case of Privad). The broker has no basis for choosing which fraction of clients the ad should go to other than random selection, and so may not select the set of clients with the highest click probability.

In the case of Adnostic, a relatively limited number of ads are sent to the client (30 is suggested in [14]). The only criteria for selecting these 30 is the value of $(B \times Q)$, where Q can be based on how well the ad matches the web page the user is accessing. If the web page matches the user's interest, then the ads selected by the broker will likely have a high user score as well. If the web page is rather more general (a news article, for instance), then the ads selected may well not match the user's interest, and may have a low user score even though the broker holds other ads that have a high user score for that user. In general, the more ads there are that are not transmitted, the higher the likelihood of missing high-ranked ads. From the Bing trace described in Section 6, we found that the average number of ads per auction is 137. Using Adnostic's suggestion of sending 30 ads, this means that almost 80% of the ads don't participate in the full auction. How damaging this is depends on the relative weight given to the user score U, and on how much U varies from ad to ad. This is because over time the average user score gets reflected in the CTR, which is accounted for in the quality score Q.

5.2.3 Overhead and Latency

RaB and Ra3 add stress to both Adnostic and Nurikabe, to the point where neither are very attractive options. In the case of Adnostic, the extra round trip to the broker (RaB) or third-party (Ra3) has to happen in real-time because the ads must be delivered in time to render the web page. This

is compounded by the fact that many ads must be sent.

In the case of Nurikabe, the sheer volume of ads that must be transmitted to the broker or 3rd-party for ranking will be a scaling challenge. The Bing trace contained 15M unique ads for a single day, with an average ad lifetime of roughly 9 days. This translates into a little over 1.7M new ads per day per client. Each of these ads is given a user score by each client, which is then transmitted to a broker or 3rd party for ranking. The Bing trace also counted 14M unique clients. This then translates into 24 tera-ads per day that must be ranked. Fortunately, this ranking function can be split over many machines (with each taking some fraction of the total number of ads to rank) without hurting the accuracy of the ranking significantly (assuming that each machine has a representative sample of ads). Therefore, while challenging, this sort is doable.

Though this is not related to the auction per se, note that each ad is roughly 250 bytes of text including the URL. Even ignoring client updates to B and Q values over the lifetime of an ad, this still requires 425MB of ad download per day per client (uncompressed), or about 43MB compressed (in bulk) for Nurikabe.

5.2.4 Auction Scope

An important aspect of the auction is the scope of the auction, by which we mean the set of ads that compete in any given auction. As a general rule, the more ads that compete, the higher the CPC. This is simply because the more ads there are, the more probabilistically likely the next-ranked ad will have a $(B \times Q \times U)$ closer to that of the clicked ad. On the other hand, the larger the auction scope, the less fair it is in the sense that very different types of ads must compete. A local pizza store may not wish to compete with Mercedes for ad boxes.

The auction scope for systems like Bing and Google is the set of ads whose keywords match that of the search or web page. The auction scope for Adnostic is the same, with the caveat that some ads are filtered out do to the ad limit (Section 5.2.2). The auction scope for Nurikabe is all ads. Privad falls somewhere in between, and the scope is somewhat tunable. Privad can choose, as a matter of policy, to set the scope to be all ads resident at a given client. It could also set the scope to be all ads within an interest channel at a given client. Since Privad can make interest channels more general or more specific, there is a way to adjust the auction scope.

5.3 Attacks

In this section, we discuss various forms of attacks that can exploit the auction systems.

5.3.1 Client click fraud

The client can commit a form of click fraud by lying about the value of $((B_n \times Q_n) \times (U_n/U_c))$ (RaC) or (U_n/U_c) (RaB or Ra3). By inflating or deflating these values, the client can cause advertisers to pay more or less, and cause publishers to earn more or less. At a high level, this is very similar to normal forms of click fraud that occur today, and in this sense our auctions do not allow fundamentally new forms of click fraud.

Of the three private advertising schemes, Adnostic is in the best position to deal with this click fraud. This is because Adnostic does not try to protect the click, and so the Adnostic broker can monitor each client's values and detect irregularities. Nurikabe, on the other hand, is in the worst position to deal with this click fraud. This is because the Nurikabe Broker has no visibility into the client at all. Rather, all it can do for click fraud is limit the number of clicks that any client can make. Privad falls somewhere in between, because the Privad broker is able to report irregularities to the Privad proxy, which is then able to identify misbehaving clients.

5.3.2 Mixing Proxy fingerprints client user scores and resulting ranking

It is difficult but conceivable that the mixing proxy could determine user profiles through observation of the client user scores and rankings. For instance, the mixing proxy could establish a number of fake clients that pretend to have various profile attributes, and establish fingerprints of the resulting user scores and rankings. One way to do this might be to determine $(B \times Q)$ given user scores and corresponding ad ranks, and use these values as the fingerprint. The mixing proxy could then compare these fingerprints with the corresponding fingerprints of real clients. It could be that the signal-to-noise ratio is high enough to successfully pull off this attack. One way to prevent this would be to encrypt user scores and rankings. The user scores could be encrypted using the brokers (RaB) or 3rd-party's (Ra3) public key, and the rankings could be encrypted using symmetric keys created by the clients and conveyed securely to the broker or 3rd-party. These symmetric keys would be frequently modified to prevent the broker or 3rd-party from linking user scores with the same client, and possibly launching a similar fingerprint attack.

5.3.3 Broker manipulates $[B_n \times Q_n]$ (RaB, Ra3)

An attack that a malicious broker could launch on Nurikabe or Privad is to identify clients by inserting unique IDs into the encrypted fields $[B_n \times Q_n]$. In the case of Nurikabe, when click reports are received with these fields copied back, the broker knows which clients (as identified by IP address) sent the click report. In the case of Privad, where both ads and reports are proxied, the broker can still link reports within an interest channel or across interest channels, to build up user profiles of otherwise anonymous clients. If these profiles are detailed enough, they may themselves be unique and linkable to identifiable users through external means. [4]

The homomorphic encryption variant described in Section 4.4 defends against this attack. Because the client multiplies the received encrypted fields with other fields, the values generated by the broker are obscured.

5.4 Discussion

Except for the fact the RaC reveals the $(B \times Q)$ product, RaC is superior to RaB and Ra3 in all other respects. It is simpler and has less overhead. It also avoids the need for the mixing proxy, a component that must not collude with the broker, and therefore requires a distinct organization to operate it. Most importantly, RaC conveys less information about the user than RaB or Ra3, which leak user scores and

[4]One might argue that the same attack can be launched simply be creating unique Ad IDs transmitted in the clear. However, this attack can at least be detected by third parties, for instance running honey-farms of clients.

(U_n/U_c) ratios. While we do not show conclusively that this information can be exploited, we outline ways in which it might be exploited. The downside of RaC of course is that it exposes more information about the advertiser $(B \times Q)$. This information, however, is even today indirectly revealed through the ad rankings returned in ad boxes. RaC only lowers the cost of obtaining this information.

While we believe that all three design schemes are viable, on balance we find RaC to be the best.

6. EFFECT OF CHURN

Section 5.2 describes how various delays in all three auction systems, when used with Nurikabe or Privad, may distort rankings and CPC computation. How detrimental this delay is depends on how much churn there is in $(B \times Q)$, and how this churn affects ranks, CPC values, and broker revenue. In this section, we use trace data from Microsoft's Bing advertising platform to study in depth the effect of these auction delays from both the advertiser and broker perspective. We find that, while churn exists, it has only a negligible impact on broker revenue and advertiser costs.

6.1 What Causes Churn?

$B \times Q$ for an ad changes when either B or Q changes. B can change in one of three ways: first, the advertiser can manually update the bid; second, the ad network can automatically update the bid (as directed by the advertiser); third, a 3rd-party may update the bid on the advertiser's behalf. Each of these has different churn characteristics:

Advertiser: Manual updates, we believe, cause very little churn since they are reactive over a long feedback cycle. Advertisers receive updated campaign information (i.e. how many clicks, actual amount charged, budget left) at fairly coarse intervals (few times a day). This limits the number of informed changes to their campaigns.

Ad Network: The advertiser can invoke functionality provided by the ad network to optimize his bidding strategy. For example, the ad network may allow the advertiser to set a preferred rank (e.g. position 4), and the ad network automatically lowers or raises the bid to satisfy the request based on the market. Other examples may include automatically modifying bids to meet a target number of impressions per day (while still being charged only for clicks), or modifying bids based on time of day etc. Some of this functionality (e.g. modify bids based on time-of-day) can be implemented in the client and would therefore not result in any added churn. Other functionality (e.g. preferred rank) tends to be implemented today as a periodic update (once every few hours).

3rd Party: Search Engine Optimization (SEO) companies optimize their client's bidding strategy in real-time [1] e.g. based on trending terms, real-time click-through rates, etc. This could potentially result in high bid churn, however, due to the premium nature of these SEO services, only a small number of ads would be affected.

Aside from changes in B, Q can also change. Recall Q in our model is a function of what the broker knows. In case of Nurikabe, this is based solely on the ad (past CTR, landing page quality, etc.). In case of Privad, Q is additionally based on the interest channel (e.g. relevance of the ad to that channel). Either way, Q is largely a property of the ad itself, which we don't expect to change quickly or dramatically. In any event, our Bing auction trace does not allow us to study changes in Q, since it does not isolate user-derived components of Q from other components.

6.2 How Does Churn Affect Auctions?

Today auctions take place at the time when an ad is displayed to the user; ranking and CPC calculations can immediately reflect any changes in B or Q. Privacy preserving auctions described in Section 4 are limited in terms of how fast new B and Q information can be incorporated. Since Q does not rapidly change over time or can be engineered to remain relatively stable (e.g., using U to reflect short-term changes in click probability), the main source of churn is the changes in B. To understand the effects of churn in B values, we simulate auctions that use stale B information for ranking and CPC computation, and then compare the resulting ranking and CPC computation with auctions that use up-to-date B information.

6.3 Dataset

For our trace driven simulations, we sampled around 2TB of log data from Bing's auction engine spanning a 48 hour period starting September 1, 2010. The data covers over 150M auctions for over 18M unique ads shown to North American Bing search users across all search topics. The trace record for an auction lists all the ads that participated in it (whether the ad was ultimately shown or not), the bids corresponding to each ad, the corresponding quality scores, and which if any of the ads were ultimately clicked by the user.

6.4 Methodology

We re-compute auction rankings and the CPC for each auction in our dataset using stale bid information; we vary staleness from 1 minute to 2 days.

Auction rankings are re-computed using bid and quality data from the trace. Since our trace does not show when the advertiser updated the bid, we infer the time based on multiple auctions that a given ad participates in. If the bid for the ad is the same for two consecutive auctions, we infer that the bid did not change during that interval. If the bid is different, we infer that the bid changed sometime between the two auctions; we use the mid-point as the time of change. To simulation an auction at time T with stale information from d minutes ago, we simply use the bids current as of time $T-d$ in our trace. The quality score in the trace is based on user features (e.g. search query), which correspond to U in private auctions; since the client always has the current value of U we use the same quality score for simulated auctions as in the trace.

CPCs are re-computed based on the re-computed auction rankings (using the second-price formula of Equation 2). In other words, for an adbox at time T in the trace, we compute the ranking based on bid values recorded at time $T-d$ and populate the adbox using resulting ranking. If the user clicked on an ad in this adbox, the bid of the next lower ranked ad B_n that we use in the CPC computation is the stale B_n taken at time $T-d$.

One limitation we face is that we cannot predict the change in user behavior when auction rankings change. Consider, for example, two ads A_1 and A_2 where in the trace they are ranked 1 and 2, while in the simulated stale auction they are ranked 2 and 1 respectively. If the user clicked A_2 in the trace, what might we expect the user to click in our

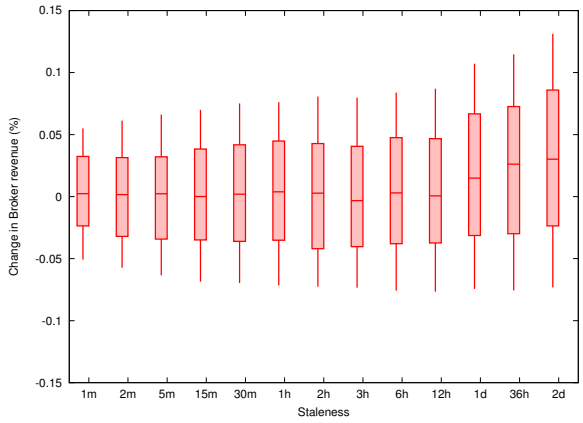

Figure 3: Change in broker revenue

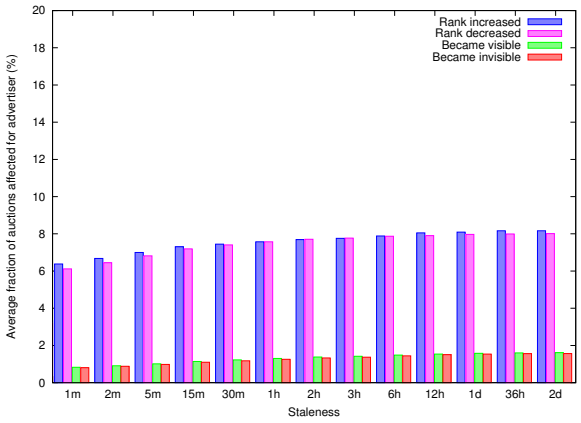

Figure 4: Fraction of auctions with modified rankings

simulation? One option is to model the user as clicking the same ad he clicked in the trace; thus in this case the user clicks A_2 in the simulation. Another option is to model the user as clicking the same position he did in the trace; in this case the user clicks position 2 (A_1 in the simulation). In reality, the user model is neither of these two extremes — it is well-known that both ad content and rank effect click-through rates [6]. To account for this, we simulate 5 user models: 1) same position, 2) 75% same position and 25% same ad, 3) 50%-50%, 4) 25% same position and 75% same ad, and 5) same ad. Thus we establish an envelop of possible user behavior to get a sense of the upper- and lower- bounds of our simulation results. Note that always clicking on the same ad is a strictly conservative estimate. This is because an ad that was clicked in the trace but is not shown to the user in our simulation (due to being ranked too low) would not get clicked; at the same time, under the same-ad model, an ad that was not shown in the trace (due to being ranked too low) and was therefore not clicked would have no chance of getting clicked even if it were to be shown to the user in the simulation. This asymmetry biases the simulation towards fewer clicks (and therefore lower revenues). The only user model immune to this limitation is the same-position model.

A second limitation we face is that we cannot predict how advertisers would change their bidding strategy in response to auctions being based on stale information. Enterprising advertisers or SEOs, may for instance, attempt to predict what bid they might want to make 1hr hence, and enter it into the system well in advance. Advance bidding would reduce the effective staleness of information. For our purposes, we assume the bidding strategy does not change.

6.5 Simulation Results

Overall our simulations show that there is no appreciable change in broker revenue for using stale bid information; even in the most conservative cases, the revenue is within $\pm 0.1\%$ of today. For advertisers, while stale bids affect their auction rankings, they do so in a balanced manner with cases of higher-than-today rank canceling out cases of lower-than-today rank resulting in zero net change.

Figure 3 plots the change in broker revenue compared to today as a function of the staleness of information used and the user model. The x-axis varies the stateless of bids from 1 minute to 2 days. The box-and-whisker plot varies the user model with the top whisker showing the outcome where the user clicks the same position, and the bottom whisker showing when the user clicks the same ad; the top edge of the box shows 75% same position and 25% same ad, and vice versa for the bottom edge of the box; the line in the middle shows the 50%-50% case.

The first observation we make from Figure 3 is that under a 50-50 user model, change in revenue is practically 0% even with bid information as stale as up to 12 hours. Under the 75-25 and 25-75 models, the change is almost always between $\pm 0.05\%$, and only in the extreme cases 100-0 or 0-100 does it pass $\pm 0.1\%$. More importantly, the change increases very gradually. This is good news since it means a private advertising system would not have a hard delay deadline beyond which there would be disproportionate change in revenue. Instead the system can strive to do the best it can, and reduce revenue change proportionally. The extremely gradual rate of change also means that system design trade-offs can be biased towards scalability and other engineering goals without much concern to revenue since it changes very little in the first place.

At first blush the effect of the "same-ad" user-model appears to be to reduce the revenue, but this is deceptive. As mentioned earlier, the more the user clicks on the same-ad (going from 0% to 100% from the top whisker to bottom), the more biased the simulation is towards fewer clicks and therefore less revenue. Recall that only the top-whisker is unaffected by this simulation bias.

The second observation we make from Figure 3 is the slight upwards trend of the top-whisker signifying higher revenues as more stale information is used. This suggests a consistent trend of advertisers (as a whole) reducing their bids over time. We don't know the cause of this trend.

Next we turn to the advertiser perspective. We compute for each ad the fraction of auctions where the user-visible simulated ranking increased or decreased compared to the trace, and whether the ad became visible or invisible due to being ranked high-enough or too-low as compared to the trace. Figure 4 plots the average of these numbers across all ads as a function of the staleness of bid information used.

We first observe that both increased ranks and decreased ranks are roughly equal, and so average to nearly zero. The same is true for ads becoming visible or invisible. While this is consistent with the revenue change in Figure 3 averaging out to zero, we note that there are other ways the revenue

could average out to zero while being unfair to advertisers. For instance, fewer increased ranks could have been compensated by more cases where the ad became visible thus still resulting in zero revenue change while being unfair to the advertiser; luckily, this is not the case.

We observe next that there is a very small impact of staleness on change in ranks; it begins with around 12% of auctions for 1 minute stale data, and quickly converges to around 16%. The reason this number is high is because of the cascade effect — if a single ad jumps from a low rank to a high rank, it causes all the ads in between to register a "change" in rank; thus a single change in bid can affect up to ten ads. The impact, however, is very little; the ad jumping from low to high might register a change of 10 ranks, however, the other 10 ads would register a change of only 1 rank each (not captured in the graph). Overall we found a median *net change* of 1 rank for every 820 auctions the ad participates in.

To summarize, based on extensive simulations across varying degrees of staleness and different user-models, there is little impact on broker revenue as compared to today, and little impact on advertiser fairness as compared to today.

7. RELATED WORK

There is a substantial body of work on private auctions where the primary goal is keeping bids secret from the broker and from other auction participants ([5] provides a comprehensive overview). Most of solutions proposed in this space are based on combinations of such cryptographic mechanisms as secure function computation, oblivious transfer and homomorphic encryption. High computational and communication complexity imposed by these systems make them impractical for our problem. To the best of our knowledge, we are not aware of any prior published work that addresses the problem of user privacy in online advertising auctions.

8. SUMMARY AND FUTURE DIRECTIONS

This paper addresses the challenge of designing an online advertising auction for a private advertising system that leverages the user profile information while keeping the user profile private. We broadly explore the design space, proposing three types of auctions, and analyzing them in the context of three previously defined private advertising systems. Overall, we find that one of the systems, Rank-at-Client, is the simplest, most efficient, exposes the client to the least privacy threats, and works well with all of the advertising systems. On the other hand, it exposes some additional advertiser information. Finally, noting that our auction designs suffer delays that cause out-of-date bid information to be used in rankings, we use Bing advertising system auction trace to determine the effect of these delays. We find the effect to be very minimal, and so conclude that our auction designs are viable.

As future work, we plan to implement the auction system to operate with a medium-scale deployment of Privad planned for next year (several 10's of thousands of users). We also plan to do a measurement study of ads served by Bing to determine to what extent advertisers can reverse-engineer each other's bids in today's systems. This will quantify how much advertiser privacy loss is incurred by the Rank-at-Client scheme. Each of the private advertising schemes so far proposed makes the tacit assumption that there is only a single broker, and a single profiler operating at each client. We are interested in exploring what happens if there are multiple profilers in each client. In particular, the profilers may be able to compete for ad boxes, thus adding a new element to the auction that is not unlike the way ad exchanges operate today.

9. REFERENCES

[1] S. Clifford. Instant Ads Set the Pace on the Web. *The New York Times*, 2010. `http://tinyurl.com/yl8dt29`.

[2] DoubleClick. DART for Advertisers. `http://www.doubleclick.com/products/dfa/index.aspx`, 2009.

[3] B. Edelman, M. Benjamin, and M. Schwarz. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. *American Economic Review*, 97(1):242–259, Mar. 2007.

[4] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *Information Theory, IEEE Transactions on*, 31(4):469–472, 2002.

[5] Felix Brandt. *Handbook of Financial Cryptography and Security*, chapter Auctions, pages 49–58. CRC Press, 2010.

[6] J. Feng, H. Bhargava, and D. Pennock. Implementing sponsored search in web search engines: Computational evaluation of alternative mechanisms. *INFORMS Journal on Computing*, 19(1):137, 2007.

[7] Google, Inc. AdWords: Advertise Your Business on Google. `http://adwords.google.com/`.

[8] S. Guha, B. Cheng, and P. Francis. Challenges in Measuring Online Advertising Systems. In *Proceedings of IMC '10*.

[9] S. Guha, B. Cheng, A. Reznichenko, H. Haddadi, and P. Francis. Privad: Rearchitecting Online Advertising for Privacy. Technical Report TR-2009-4, Max Planck Institute for Software Systems, Kaiserslautern-Saarbrücken, Germany, 2009. `http://mpi-sws.org/tr/2009-004.pdf`.

[10] S. Guha, A. Reznichenko, K. Tang, H. Haddadi, and P. Francis. Serving Ads from localhost for Performance, Privacy, and Profit. In *Proceedings of HotNets '09*.

[11] B. Krishnamurthy and C. E. Wills. Cat and Mouse: Content Delivery Tradeoffs in Web Access. In *Proceedings of WWW '06*.

[12] D. Levin, B. Bhattacharjee, J. R. Douceur, J. R. Lorch, J. Mickens, and T. Moscibroda. Nurikabe: Private yet Accountable Targeted Advertising. Under submission. Contact `johndo@microsoft.com` for copy, 2009.

[13] Microsoft, Inc. Start advertising on Yahoo! Search and Bing. `https://adcenter.microsoft.com/`.

[14] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas. Adnostic: Privacy Preserving Targeted Advertising. In *Proceedings of NDSS '10*.