Managing the Quality of CPC Traffic

Bobji Mungamuru
Stanford University
bobji@i.stanford.edu

Hector Garcia-Molina
Stanford University
hector@cs.stanford.edu

November 4, 2008

Abstract

We show how an online advertising network can use filtering, predictive pricing and revenue sharing together to manage the quality of cost-per-click (CPC) traffic. Our results suggest that predictive pricing alone can and should be used instead of filtering to manage organic traffic quality, whereas either method can be used to deter click inflation.

1 Introduction

Advertisers have been moving online in increasing numbers over the past decade. The online medium has gained popularity because it can potentially reach a very targeted audience, often at a lower cost than print or broadcast media, yielding a higher return on investment (ROI).

In any advertising medium, advertisers will be willing to pay more for higher-quality traffic. Here, "quality" is related to the likelihood that a "visitor" (i.e., viewer, listener, reader or web surfer, depending on the medium) will eventually be acquired as a "customer" of the service being advertised (e.g., buying a product, making a campaign contribution, or signing up for a mailing list). And as in traditional media, traffic quality can vary greatly depending on the channel (in the online context, a "channel" is simply a website from which traffic originates).

Broadly speaking, there are three techniques that online advertising networks can use to influence the quality of traffic (i.e., impressions and click-throughs) on their network: *filtering*, *predictive pricing* and *revenue sharing*. Filtering refers to distinguishing between valid and invalid traffic. Examples of invalid traffic include unintentional click-throughs, web-crawler traffic and fraudulent traffic. The implication is that the advertiser will not be billed (and the publisher will not be paid) for traffic that is deemed invalid. Predictive pricing [10] is concerned with a different aspect of traffic quality, namely how targeted it is. For example, it may be more effective for a university to advertise its MBA program in the *Economist* magazine rather than *Communications of the ACM*, since the former is presumably more targeted toward potential students than the latter. Examples of predictive pricing programs include Google's "Smart Pricing" [6] and Yahoo's "Quality-Based Pricing" [17]. Finally, revenue sharing is the practice of paying out to publishers a fraction of revenue earned from advertisers. Revenue sharing is the reason why publishers display advertisements in the first place. In this paper, we show how an advertising network can use these three techniques together to optimally manage traffic quality – that is, in a way that maximizes the profits of the advertising network. Our focus will be on *cost-per-click* (CPC) advertising, as it currently accounts for the majority of online advertising traffic.

1.1 Motivation: A Lemons Market

Managing traffic quality is a problem of immense practical significance because the CPC market has the two signature characteristics of a “lemons market” [1]: asymmetric information (publishers know more about their audience that an advertiser does, and can engage in click fraud) and products (i.e., clicks) of varying quality. Without a reputable, long-lived intermediary to counteract an advertiser’s uncertainty about traffic quality, the market may collapse: high-quality publishers would not receive fair value for their traffic, and advertisers get saddled with mostly low-quality traffic.

Advertising networks act as this intermediary, by pricing traffic differently based on quality. For a network to remain in business, it must deliver sufficiently high-quality traffic to advertisers. If substantially higher quality is offered by a competing network, advertisers will switch over to that network. So, although filtering, predictive pricing and revenue sharing can help advertising networks attract and retain lucrative traffic, applying these tools suboptimally can mean that a network is “leaving money on the table”. And in a market that, by most estimates, is worth several billions of dollars, losses due to a suboptimal policy can be tremendous.

As we alluded to earlier, there are two distinct-but-related aspects of traffic quality: *validity* and what we call *targetedness*. Validity refers to whether a click is valid or invalid. Valid clicks have a strictly positive probability of becoming a conversion, whereas invalid clicks have zero probability. Targetedness refers to the likelihood that a valid click will become a conversion. Targetedness is measured by the *conversion rate*, defined in Section 2.

Most existing research on traffic quality is concerned with validity, and click fraud in particular (e.g., [4, 5, 8, 11]). There has also been much discussion of traffic quality in the media and online forums, mostly centered on click fraud [13, 14, 15, 16]. However, traffic quality involves more than just validity, and validity is a broader issue than just click fraud. Relatively little attention has been paid to the targetedness aspect of traffic quality. To the authors’ knowledge, our earlier work [10] is the only published study to date on predictive pricing as a tool to manage targetedness. However, in [10], targetedness is considered independently i.e., it is assumed that all traffic is valid. In this paper, filtering, predictive pricing and revenue sharing are all used together to shape the quality of CPC traffic flowing through an online advertising network.

1.2 Overview

We begin by presenting an economic model of the CPC advertising market. The model is a simplification of what happens in practice – our intent is to hone in on decisions that affect traffic quality. Although our focus is on CPC, an analogous model can be developed for the *cost-per-mille* (CPM) and *cost-per-action* (CPA) pricing models. We present an algorithm for computing an advertising network’s best-response function. That is, if a network knows the filtering, predictive pricing and revenue sharing policies of its competitors, what actions should the network choose in response?

We then study the properties of a network’s best-response, particularly as they relate to the management of traffic quality. Our main conclusions can be summarized as follows:

- Due to the underlying incentives, it is important to distinguish between the two sources of click traffic: *organic* traffic and publisher-initiated *click inflation*. All valid traffic is organic, by definition, although organic traffic can also be invalid. All click inflation is, by definition, invalid (it is a form of click fraud).

Symbol	Description
I, J, K	Number of publishers, networks and advertisers
i, j, k	Index over publishers, networks and advertisers
u_j	Network j's aggressiveness at filtering
g_{ij}	Predictive price applied to i's traffic by j
h_j	Revenue share paid out by network j
c_{ij}	Fraction of publisher i's clicks sent to ad network j
v_{kj}	Advertiser k valuation of ad network j's clicks
$\mathbf{g}_j$	$= \{g_{ij}\ \forall i\}$, predictive prices chosen by network j
r_i	Fraction of publisher i's traffic that is valid
V_i	Volume of clicks on publisher i's site
B_i	$= (1 - r_i)V_i$, volume of invalid ("bad") clicks
G_i	$= r_i V_i$, volume of valid ("good") clicks
β_i^{Pub}	Targetedness of publisher i's traffic
β_j^{Net}	Network j's skill at matching
β_k^{Adv}	Effectiveness of advertiser k's ads
β_{ijk}	Conversion rate of clicks going from i to j to k
A_i	Nominal number of conversions due to publisher i
A_{ijk}	Number of converted clicks going from i to j to k
ξ_{ijk}	Fraction of i's traffic that j forwards on to k
$\hat{\beta}_i$	Network j's estimate of β_i^{Pub}
$\hat{r}_i$	Network j's estimate of r_i
σ	Standard error in network j's estimate of r_i
γ_j	Network j's effectiveness at filtering
$f_j(u_j)$	$= u_j^{\gamma_j}$, network j's false-negative rate
θ_j	Network j's auction revenue per click
κ_j	Network j's nominal auction revenue per click
E_{ij}	Effective number of clicks that i is paid for by j
π_{ij}	Publisher i's revenue from clicks sent to network j
η_j	Network j's total revenue

Table 1: Summary of notation.

- To manage the quality of organic traffic, it is unnecessary (and, in some cases, suboptimal) to use both predictive pricing and filtering simultaneously. Predictive pricing alone is enough. (Section 3)

- Filtering can, however, be (indirectly) useful in fighting click inflation, as long as the performance of the filtering algorithm can be accurately characterized. Otherwise, predictive pricing can be used to fight click inflation. (Sections 4.2 and 4.3)

2 CPC Advertising

In this Section, we present a model of the CPC advertising market. Our model is a generalization of both [11] and [10], so we remain notationally consistent with these papers whenever possible. Table 1 is a summary of the notation used in this paper.

We model the CPC market as a one-shot dynamic game between three classes of players: content publishers, advertising networks and advertisers. *Content publishers* (or, *publishers*) publish websites and display advertisements alongside their content. *Advertisers* design advertisements (or, *ads*) and bid at auction on *keywords* that best describe the interests of their target market. *Advertising networks* (or, *networks*) act as intermediaries between publishers and advertisers, by first judging which keywords best describe each publisher's content, and then delivering ads to the publisher from advertisers that have bid on those keywords.

If a user visits a publisher's site and clicks on an ad related to a given keyword, we say that a click-through (or, *click*) has occurred on that ad. If the click is deemed valid by the network, the advertiser pays the network a small amount. The network then pays out a fraction of this amount to the publisher where the click originated. Filtering is the process of detecting invalid clicks. Predictive pricing affects how much the advertiser is billed by the network. The revenue share determines what fraction of this billed amount the network will pay out to the publisher. A small fraction of valid clicks become *conversions* for the advertiser e.g., a product purchase, or a sign-up to an e-mail list. The advertiser earns some revenue each time a click becomes a conversion.

2.1 Model

Consider a single keyword. Suppose there are I publishers whose content is relevant to the keyword, K advertisers interested in buying clicks on this keyword, and J networks. Typically, $I \gg K \gg J$.

Each publisher i receives V_i clicks on his website, of which only a fraction r_i are valid. That is, publisher i receives $r_i V_i$ valid clicks and $(1 - r_i)V_i$ invalid clicks in total. For now, we assume that r_i and V_i are fixed parameters that describe the validity of publisher i's traffic (we will relax this assumption in Section 4, when we discuss click inflation).

In our model, each publisher i decides how to allocate its volume, V_i, of clicks across the J networks. Note that in practice, publishers allocate ad impressions (or, "page views"), rather than clicks. Under some reasonable assumptions (not discussed here), however, it is equivalent to model clicks (rather than impressions) as the objects being bought and sold. Let c_{ij} be the fraction of publisher i's clicks that are sent to network j. Then, $V_i c_{ij}$ is the total number of clicks that publisher i sends network j, of which

$$r_i V_i c_{ij} \tag{1}$$

are valid clicks.

The algorithms used by networks to filter out invalid clicks are prone to error. In particular, the algorithms may produce false positives (or, "Type I" errors) by marking valid clicks invalid. They may also produce false negatives (or, "Type II" errors) by marking invalid clicks valid.

Let u_j be the fraction of valid clicks that network j's filtering algorithm correctly identifies as valid (i.e., *true negatives*). We assume in our model that $f_j(u_j) \equiv u_j^{\gamma_j}$ will be the fraction of invalid clicks that network j mistakenly marks valid (i.e., *false negatives*), where $\gamma_j \in [1, \infty)$. The function f_j reasonably approximates the relationship between the true-negative rate and false-negative rate in many real-world binary-decision tasks. For example, f_j corresponds to a concave *receiver operating characteristic*, or "ROC curve". The fraction u_j is a measure of how aggressively network j is filtering for invalid clicks (lower u_j means more aggressive). The parameter γ_j is a measure how effective network j is at filtering (higher γ_j means more effective, since it leads to fewer false negatives for a given u_j).

Of all the clicks that publisher i sends to network j,

$$N_{ij} \equiv u_j r_i + u_j^{\gamma_j}(1 - r_i) \tag{2}$$

is the fraction that is marked valid. Thus,

$$N_{ij} V_i c_{ij} \tag{3}$$

is the number of publisher i's clicks that network j marks valid. Marking a click invalid only means that the network will not charge the advertiser for the click, and will consequently not pay the publisher. The user is forwarded to the advertiser's site, irrespective of whether the click is marked valid or invalid.

For each click coming from publisher i that is marked valid, network j bills advertisers for only a fraction g_{ij} of a click i.e., advertisers receive a $(1 - g_{ij})$ discount. The fraction g_{ij} is the *predictive price* that network j applies to publisher i's traffic. The *effective* number of clicks that publisher i is paid for by network j is then:

$$E_{ij} \equiv N_{ij} V_i c_{ij} g_{ij} \tag{4}$$

Suppose θ_j is the expected auction revenue per click on network j. Of each dollar of revenue from advertisers, network j pays out a fraction h_j to publishers. The fraction h_j is referred to as the *revenue share*. Then, the total revenue to publisher i from network j is:

$$\pi_{ij} \equiv N_{ij} V_i c_{ij} g_{ij} h_j \theta_j \tag{5}$$

Let $\mathbf{g}_j \equiv \{g_{ij}\ \forall i\}$. We refer to the pair $(\mathbf{g}_j, h_j)$ together as network j's *pricing policy*. We refer to the triple $(u_j, \mathbf{g}_j, h_j)$ together as network j's *traffic policy*.

2.2 Conversion Rates

The clicks sent by the publishers to the networks are, in turn, distributed amongst the K advertisers (in proportions related to the advertisers' bids). Of all the clicks sent from publisher i to advertiser k via network j, let β_{ijk} be the fraction that become conversions. The fraction β_{ijk} is referred to as a *conversion rate*. We assume that conversion rates are *separable* i.e., that each β_{ijk} is a product of three factors:

$$\beta_{ijk} \equiv \beta_i^{\text{Pub}} \beta_j^{\text{Net}} \beta_k^{\text{Adv}} \quad \forall (i, j, k) \tag{6}$$

Each factor in (6) has a different interpretation. β_i^{Pub} measures how targeted publisher i's traffic is with respect to the keyword in question. β_j^{Net} measures how good network j is at matching publishers' content with advertisers' ads. β_k^{Adv} measures the quality and effectiveness of advertiser k's ads.

2.3 Sequence of Events

Our one-shot dynamic game is comprised of two stages:

1. In the *first stage*, each network j selects and announces its traffic policy (i.e., its filtering aggressiveness, u_j, predictive prices, $\mathbf{g}_j$, and revenue share, h_j).

2. In the *second stage*, each publisher i decides which networks to sell its clicks on (i.e., its *allocations*, $\{c_{ij}\ \forall j\}$). Simultaneously, each advertiser k decides how much it is willing to pay for clicks from each network j (i.e., its *valuations*, $\{v_{kj}\ \forall j\}$).

After the second stage, payoffs are realized: a) publishers sell clicks (i.e., display ads) on their chosen networks, b) networks mark a subset of these clicks valid, c) advertisers pay the networks (possibly a discounted price) for the clicks marked valid, and d) networks pay out a fraction of earned revenues to publishers. Recall that users are forwarded to the advertiser's site even if a click is marked invalid.

Publishers, advertisers and networks decide on their allocations, valuations and traffic policies (respectively) with the objective of maximizing their own revenues. Network j's revenues are a function of the decisions taken by advertisers and publishers in the second stage. Advertisers' and publishers' decisions, in turn, are a reaction to the traffic policies chosen by all the networks in the first stage. Thus, each network's profits will depend on the traffic policies of all of its competing networks.

2.4 Best-Response Traffic Policies

Following the derivation in [10], we can compute the best-response traffic policy of network 1, holding the policies of all other networks fixed, and assuming an equilibrium is played in the second stage. Network 1's best-response is a solution to the following optimization problem:

$$\begin{array}{ll} \text{maximize} & \eta_1 \equiv \beta_1^{\text{Net}} \left(\sum_i r_i V_i \beta_i^{\text{Pub}} c_{i1} \right) (1 - h_1) \kappa_1 \\ \text{subject to} & X_{ij} = N_{ij} V_i g_{ij} h_j \theta_j \ \ \forall (i,j) \\ & \sum_j c_{ij} X_{ij} = \max_j X_{ij} \ \ \forall i \\ & \sum_j c_{ij} = 1 \ \ \forall i \\ & \theta_j = \kappa_j a_j \ \ \forall j \\ & a_j = \beta_j^{\text{Net}} \dfrac{\left(\sum_i r_i V_i c_{ij} \beta_i^{\text{Pub}}\right)}{\left(\sum_i N_{ij} V_i c_{ij} g_{ij}\right)} \ \ \forall j \\ & N_{ij} = u_j r_i + u_j^{\gamma_j} (1 - r_i) \ \ \forall (i,j) \\ & 0 \le u_1, g_{i1}, h_1, c_{ij} \le 1 \ \ \forall (i,j) \end{array} \tag{7}$$

The objective in (7) is an expression for network 1's profit. The first three constraints encode the assumption that each publisher chooses allocations optimally in the second stage. The fourth and fifth constraints say that advertisers also choose valuations optimally. Thus, the first five constraints together imply that there is an equilibrium in the second stage between publishers and advertisers. The sixth constraint is an expression for the fraction of clicks network j marks valid (see (2)). The final constraint gives ranges for the decision variables we are interested in.

The optimization problem (7) is highly non-convex, making it difficult to even find feasible points. In Appendix A we describe an iterative algorithm, TrafficQuality, for finding approximate solutions to (7). The key step in TrafficQuality involves solving a sequence of convex optimization problems, each of which is a relaxation of (7).

2.5 Remarks

The model we have presented in this Section is strictly more general than the models presented in [11] and [10]. Fixing $g_{ij} = 1\ \forall(i,j)$ and $h_j = h\ \forall j$ (i.e., no predictive pricing and equal revenue shares), assuming $\beta_{ijk} = \beta\ \forall(i,j,k)$ (equal conversion rates) and treating u_j as a decision variable places us in the setting of [11]. On the other hand, fixing $u_j = 1$, assuming $r_i = 1$ and treating $\mathbf{g}_j$ and h_j as decision variables gives us the model used in [10]. In this sense, TrafficQuality is a generalization of the PricingPolicy algorithm presented in [10], since the optimization is done over the space of traffic policies $(u_j, \mathbf{g}_j, h_j)$, rather than pricing policies $(\mathbf{g}_j, h_j)$.

In [11], filtering aggressiveness was measured as the rate of false positives, x_j i.e., the fraction of valid clicks that are mistakenly marked invalid. Thus, $x_j = 1 - u_j$. The true-positive rate was then $x_j^{\alpha_j}$, for $\alpha_j \in (0, 1]$. A lower value of α_j indicated that network j was more effective at filtering, as opposed to a higher value of γ_j in our current model.

3 Filtering And Predictive Pricing

In this Section, we study the relationship between filtering and predictive pricing. Given the option, should a network choose one over the other, or use both together?

The main result in [11] was that, if predictive pricing is disallowed and all networks offer the same revenue share, then it is optimal for network j to filter aggressively. In particular, if network 1 is the most skilled at filtering (i.e., $\gamma_1 > \gamma_j\ \forall j \neq 1$), all publishers (even low-quality ones) will prefer to send their traffic to network 1, provided it is filtering aggressively enough (i.e., $u_1 \leq u^*$ for some known threshold value u^*).

As a concrete example, consider a market with $I = 20$ publishers and $J = 2$ networks. Both networks offer publishers a revenue share of $h_1 = h_2 = 50\%$ and set $g_{ij} = 1\ \forall i$ (i.e., no predictive pricing). The networks are equally skilled at selecting ads ($\beta_1^{\text{Net}} = \beta_2^{\text{Net}} = 1$), but network 1 is more skilled at filtering ($\gamma_1 = 10$ and $\gamma_2 = 8$). Network 2 is marking valid $u_2 = 80\%$ of valid clicks (thus, it also marks valid $u_2^{\gamma_2} = 17\%$ of invalid clicks). To simulate a wide variation in traffic quality, we assume that $r_i = 0.05i$ and $\beta_i^{\text{Pub}} = 0.0025i$ i.e., publishers are sorted in increasing order of traffic validity and targetedness, with r_i ranging between 5% and 100%, and β_i^{Pub} ranging between 0.25% and 5% (a 5% conversion rate would be considered very high in practice).

For this scenario, we used TrafficQuality (with $\mathbf{g}_1$ and h_1 disabled) to compute the best-response traffic policy for network 1. As predicted, TrafficQuality recommends that network 1 filter aggressively (i.e., send $u_1 \to 0$), causing all publishers (including the low-quality ones) to send their traffic to network 1 (in this example, $u_1 \leq u^* = 81\%$ was sufficient for network 1 to win over the market).

Now, consider the same scenario, except that network 1 is allowed to use predictive pricing. That is, we allow $\mathbf{g}_1$ to be a decision variable for network 1 (note that the revenue share, h_1, is still fixed at 50%). Using TrafficQuality with $\mathbf{g}_1$ enabled this time, we get a very interesting outcome: the optimum is now $u_1 = 100\%$. That is, TrafficQuality is recommending that network 1 stop filtering altogether, and just use predictive pricing instead.

Figure 1 shows the effective number of clicks, E_{i1}, that each publisher is paid for (as a fraction of $V_i c_{i1}$), with and without predictive pricing (see (4)). With filtering alone, network 1 is restricted to a linear E_{i1} profile (recall that N_{i1} is linear in r_i, and in this example r_i is linear in i), and all publishers choose to send traffic to network 1. With both filtering and predictive pricing enabled, TrafficQuality outputs a non-linear profile

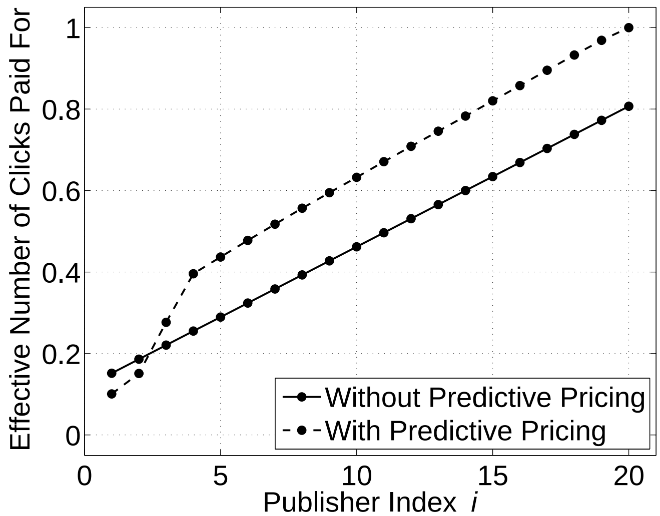

Figure 1: Effective number of clicks paid for, E_{i1}.

of predictive prices that achieves exactly the same market outcome as the filtering-only case (i.e., all publishers choose network 1), but it does so without needing to using filtering at all.

This outcome is not peculiar to the scenario described above – it is actually a general phenomenon:

Theorem 1. *Suppose* $(\tilde{u}_1, \tilde{\mathbf{g}}_1, h_1^*)$ *is a solution to (7), with* $\tilde{u}_1 < 1$*. Then, there exists another solution* $(u_1^*, \mathbf{g}_1^*, h_1^*)$ *to (7) where* $u_1^* = 1$*. The converse is not necessarily true.*

Proof. In (7), u_j only appears in the sixth constraint (i.e., the definition of N_{ij}), and in turn N_{ij} only appears as the product $N_{ij}g_{ij}$. Also, note that if $u_j = 1$, then $N_{ij} = 1$ irrespective of r_i (i.e., all clicks are being marked valid).

Therefore, if $(\tilde{u}_1, \tilde{\mathbf{g}}_1, h_1^*)$ is feasible for (7), then $(1, \mathbf{g}_1^*, h_1^*)$ will also be feasible, where $g_{i1}^* = \tilde{g}_{i1}\tilde{N}_{i1}$ and $\tilde{N}_{i1} = \tilde{u}_1 r_i + \tilde{u}_1^{\gamma_1}(1 - r_i)$. Moreover, all other variables are left unchanged. Neither N_{ij} nor g_{ij} appear in the objective of (7). Therefore, if $(\tilde{u}_1, \tilde{\mathbf{g}}_1, h_1^*)$ is optimal, then $(1, \mathbf{g}_1^*, h_1^*)$ will also be optimal.

Conversely, suppose $(1, \mathbf{g}_1^*, h_1^*)$ is a solution to (7), and let $\tilde{u}_j < 1$. If there is an i such that $r_i < 1$ and $g_{i1}^* = 1$, then $\tilde{g}_{i1} = \frac{g_{i1}^*}{\tilde{N}_{i1}} > 1$, which is infeasible. Therefore, an optimal $(\tilde{u}_1, \tilde{\mathbf{g}}_1, h_1^*)$ where $\tilde{u}_j < 1$ does not necessarily exist. □

Theorem 1 says that there is always a best-response for network 1 (i.e., a solution to (7)) that involves using predictive pricing alone (i.e., setting $u_j = 1$). Therefore, in most settings, predictive pricing can and should be used for managing traffic quality instead of filtering.

Theorem 1 holds irrespective of the numbers of publishers and advertisers and their traffic qualities. An immediate implication is that a network seemingly gains no competitive advantage from having superior algorithms for filtering, since competing networks can simply respond by implementing predictive pricing. This conclusion is a partial contradiction of the result established in [11].

Whereas filtering effectively requires careful development of algorithms for detecting specific traffic patterns, predictive pricing only requires aggregate measurements of click- and acquisition traffic and the selection of a set of coefficients (i.e., predictive prices). Of course, the challenge in predictive pricing is being able to accurately estimate these required traffic statistics (more on this issue in Section 3.2).

Combining Theorem 1 with the result from [11], we can also deduce the following:

Corollary 1. *In a subgame perfect equilibrium, each network uses either filtering or predictive pricing, but not both.*

Proof. In equilibrium, each networks uses a best-response traffic policy. From Theorem 1, if a network uses predictive pricing, it is best not to filter. From Theorem 2 in [11], if a network is not predictive pricing, its best response is to filter aggressively. □

Due to the richness of the networks' decision spaces in our game, general properties of subgame perfect equilibria are difficult to derive. Corollary 1 is one of the few general statements we can make.

3.1 Practical Implications

Theorem 1 may not seem very surprising, since predictive pricing does allow for much finer-grained publisher-level control (i.e., a factor g_{ij} for each publisher i) compared to filtering, which provides a single control u_j for the entire population of publishers. However, Theorem 1 has very significant practical implications.

3.1.1 No need to measure or control u_j

In most practical settings, u_j, the true-negative rate (i.e., aggressiveness) of network j's filtering algorithm, is difficult to measure since such a measurement typically requires a "ground truth" data set, which may not be available. In our context, ground truth would be a "representative" sample of traffic where clicks are labelled valid and invalid. Even if u_j could be measured, network j may not be able to select arbitrary values for u_j strictly between 0 and 1.

On the other hand, it is always possible to set $u_j = 0$ or $u_j = 1$ by simply marking all clicks invalid or valid, respectively (of course, with $u_j = 1$, the false-negative rate $u_j^{\gamma_j}$ would also be 1 i.e., all invalid clicks would also be marked valid). Theorem 1 guarantees that there is always a best-response where $u_j = 1$.

3.1.2 No need to know r_i and β_i^{Pub}

Even if accurate measurement and control of u_j were possible, there is a more serious practical issue concerning parameter estimation. In order for a network to use TrafficQuality to find an optimal traffic policy, it needs to know of r_i and β_i^{Pub} for each publisher i (see (7)). However, in practice, these quantities are not observed – if r_i could be observed, filtering would be unnecessary! Typically, networks are only able to measure the volume of clicks (e.g., using click logs) and numbers of conversions (e.g., using conversion tracking code installed on the advertisers' sites).

Fortunately, Theorem 1 allows a network to run TrafficQuality even without knowing r_i and β_i^{Pub}. We know that it is sufficient for network 1 to search for a traffic policy where $u_j = 1$. From (2), setting $u_j = 1$ implies that $N_{ij} = 1\ \forall i$, irrespective of r_i and γ_j. Upon substituting $N_{ij} = 1$ into the first, fifth and sixth constraints of (7), we observe that the only remaining places that r_i and β_i^{Pub} appear in (7) are as the product $r_i\beta_i^{\text{Pub}}$. As we describe in Section 3.2, it is possible to estimate the product $r_i\beta_i^{\text{Pub}}$ by observing the volumes of clicks and conversions alone.

Therefore, network j can run TrafficQuality even without knowing r_i and β_i^{Pub}, by simply setting $u_j = 1$ and estimating the product $r_i\beta_i^{\text{Pub}}$ instead. Theorem 1 guarantees that there is no loss in profits from calculating a best response in this way.

3.2 Parameter Estimation

We now demonstrate how network j can estimate the product $r_i\beta_i^{\text{Pub}}$ for each publisher i, as well as other parameters required to run TrafficQuality, using data that can be readily observed in practice.

3.2.1 Estimating $r_i\beta_i^{\text{Pub}}$

Suppose a user visits publisher i's website, on which advertiser k's ad is displayed, and the ad is delivered by network j. If the user then clicks on the ad, network j redirects the user from publisher i's site to advertiser k's site, and records the click-through in its click logs. So, by simply analysing its click logs, network j can compute V_ic_{ij} $\forall i$, which is the total number of clicks sent to network j by publisher i. Network j can also compute ξ_{ijk} $\forall(i,k)$, where ξ_{ijk} is the fraction of publisher i's traffic sent to advertiser k by network j.

Let A_{ijk} denote the number of clicks originating on publisher i's site that eventually become conversions for advertiser k, where the ad is delivered by network j (the letter A stands for "actions" or "acquisitions"). Conversion tracking software installed on the advertiser's site typically gives the total number of clicks that advertiser k converts, along with the times and dates of those clicks and conversions (alternatively, the advertiser can "self-report" the number of conversions, although the advertiser may not self-report truthfully – see [9] and [12] for a discussion). Cross-referencing this data with the click logs, a network can infer which publisher each converted click originated from. Thus, as long as the required conversion-tracking infrastructure is in place, each network j can observe A_{ijk} $\forall(i,k)$.

Using the quantities defined in Section 2, A_{ijk} can be decomposed into a product as follows:

$$A_{ijk} = V_ic_{ij}r_i\xi_{ijk}\beta_{ijk} \tag{8}$$

Equation (8) holds since $V_ic_{ij}r_i\xi_{ijk}$ is the number of valid clicks sent from publisher i to advertiser k via network j, and β_{ijk} is the fraction of these clicks that become conversions.

Therefore, using its observations of V_ic_{ij}, ξ_{ijk} and A_{ijk}, network j can compute the product $r_i\beta_{ijk}$ $\forall(i,k)$ as follows:

$$r_i\beta_{ijk} = \frac{A_{ijk}}{V_ic_{ij}\xi_{ijk}} \tag{9}$$

Then, applying the separability assumption (6), we get:

$$\left(r_i\beta_i^{\text{Pub}}\right)\beta_k^{\text{Adv}} = \frac{A_{ijk}}{V_ic_{ij}\xi_{ijk}\beta_j^{\text{Net}}} \tag{10}$$

Observe that the right-hand side of (10) can be computed directly from available data for every (i,k), whereas the left-hand side is comprised of parameters which network j needs to estimate. In particular, network j has IK (possibly noisy) data points with which to estimate the $I+K$ parameters $\{r_i\beta_i^{\text{Pub}}\ \forall i\}$ and $\{\beta_k^{\text{Adv}}\ \forall k\}$. Since there will typically be many more data points than parameters (i.e., $IK \gg I+K$), a data-fitting technique (e.g., least-squares) can be used to to do the estimation.

In practice, some advertisers may not be willing to install conversion tracking software on their sites. However, network j is most interested in estimating the product $r_i\beta_i^{\text{Pub}}$ (for use in TrafficQuality), as opposed to β_k^{Adv} for specific k. So, as long as "enough" advertisers do install conversion tracking software, data-fitting techniques can be used to compute good estimates of $r_i\beta_i^{\text{Pub}}$. Real-world deviations from, say, the separability assumption can also be (partially) compensated-for using fitting techniques.

3.2.2 Other parameters

In many cases, network j can assume $V_i = V_i c_{ij}$ whenever publisher i sends it any traffic at all (i.e., that $c_{ij} = 1$ whenever $c_{ij} > 0$). Publisher i's optimization problem (not discussed here) is such that, in most cases, its optimal allocation c_{ij} will be either 0 or 1. That is, if publisher i sends network j any traffic at all, it will send network j all of its traffic. Fractional allocations of traffic across multiple networks (i.e., $c_{ij} \in (0, 1)$) are only optimal for publisher i when there a "tie" between those networks in terms of profitability, which happens infrequently in practice.

In such cases, network j can use (10) to also infer A_i, the *nominal number of conversions* for publisher i, as follows:

$$A_i \equiv r_i \beta_i^{\mathrm{Pub}} V_i = \frac{A_{ijk}}{\xi_{ijk} \beta_j^{\mathrm{Net}} \beta_k^{\mathrm{Adv}}} \tag{11}$$

Intuitively, A_i measures the "potential" number of conversions that can result from publisher i's traffic, before adjusting for the matching algorithms of network j (i.e., β_j^{Net}) and the ad quality of advertiser k (i.e., β_k^{Adv}). In the case where $\beta_j^{\mathrm{Net}} = \beta_k^{\mathrm{Adv}} = 1 \ \forall (j, k)$, we get $A_i c_{ij} \xi_{ijk} = A_{ijk}$ and $\sum_{j,k} A_{ijk} = A_i$, both of which are consistent with the interpretations of c_{ij} and ξ_{ijk} as fractional traffic allocations. We will use A_i in Section 4, when we discuss click inflation.

Finally, we assume in (10) and (11) that β_j^{Net} is known to each network j. More generally, β_j^{Net} should be known for each network j to within a constant factor. Relative values of β_j^{Net} can be estimated from historical competitive data, or by other comparative means.

4 Click Inflation

Theorem 1 states that, for most sources of traffic, predictive pricing can and should be used for managing traffic quality instead of filtering. In fact, there are cases where lower profits can result from filtering.

However, Theorem 1 paints an incomplete picture. An underlying assumption in the proof was that r_i and V_i are fixed parameters that describe publisher i's traffic. In particular, they were not considered decision variables for publisher i. Stated differently, we assumed that all of the traffic on publisher i's site is *organic*, in the sense that the traffic is not generated or caused by the publisher itself. All valid traffic, by definition, is organic. Most forms of invalid traffic can also be considered organic, including the various forms of non-click-fraud invalid traffic (e.g., double clicks, unintentional clicks, web crawlers), as well as click fraud due to "competitor clicking" [3].

Unfortunately, r_i and V_i can indeed be manipulated by publisher i in practice. In particular, a publisher can inject a stream of fraudulent clicks into the traffic it sends to network j – this practice is known as *click inflation* [3]. Engaging in click inflation increases the total volume of clicks while leaving the amount of valid clicks and the number of resulting conversions unchanged (since none of the fraudulent clicks become conversions).

In Section 4.1, we discuss why click inflation might occur, and how to account for it in our model. In Sections 4.2 and 4.3, we present a pair of approaches that networks can use to compensate for click inflation. Roughly, one approach is to use filtering to estimate r_i, and then pay publisher i for its valid, organic traffic

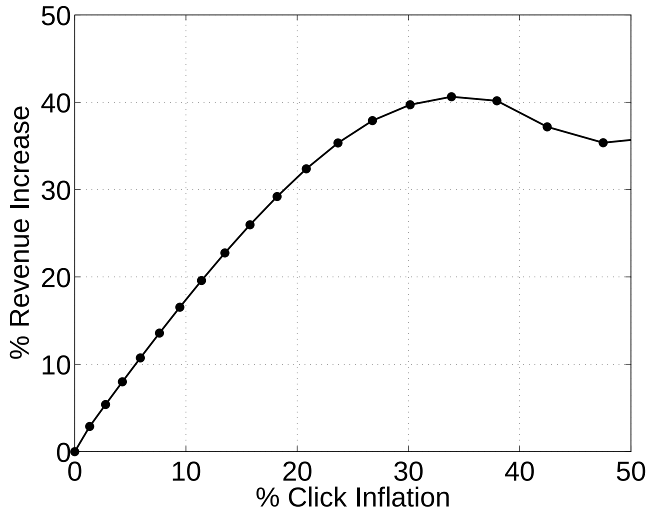

Figure 2: Increase in revenues due to click inflation.

only. Another is to find predictive prices such that the incentive for click inflation is eliminated altogether. In Section 4.4, we compare these approaches.

4.1 Why Click Inflation Occurs

It is easy to see why publishers might have an incentive to engage in click inflation. Define, for convenience, $G_i \equiv r_i V_i$ and $B_i \equiv (1 - r_i)V_i$ (the letters G and B stand for "good" and "bad"). Click inflation causes B_i to increase without a change in G_i. For example, publisher i might pay users to visit its site and click on ads, even though the users are uninterested in the product being advertised. The problem for network j is that it pays for $N_{ij}V_i c_{ij} = \left(u_j G_i + u_j^{\gamma_j} B_i\right) c_{ij}$ clicks, so for any $u_j > 0$, the number of clicks that publisher i is paid for will be increasing in B_i.

As an illustration, we computed the best-response traffic policy for network 1 in a scenario where $r_i = 1\ \forall i$ (i.e., $G_i = V_i$) and β_i^{Pub} is linear in i. We then chose the highest-quality publisher, and computed by how much its revenues would increase if it inflated its click volume by various amounts. The results are shown in Figure 2. For example, inflating click volumes by 11.5% (i.e., injecting $B_i = 0.115 G_i$ fraudulent clicks) results in a 19.6% increase in revenues.

There are two features that stand out in Figure 2. First, publishers can increase their revenues significantly by generating fraudulent traffic. Second, it not optimal for a publishers to generate an arbitrarily large amount of fraudulent traffic. In our example, the publisher's revenues are maximized (a gain of 41%) when it inflates its traffic by just 34%. Any further inflation causes its revenues to decrease, since network 1 would apply a very low predictive price to its traffic. Intuitively, this example suggests that even high-quality publishers may try to slip a small amount of fraudulent traffic through the networks' filters.

To model click inflation, we slightly modify the dynamic game described in Section 2.3 as follows:

- In the first stage, networks select traffic policies based on forecasts of organic traffic quality.
- In the second stage, publishers decide on allocations as well as whether (and by how much) to inflate B_i.
- After the second stage (i.e., after receiving the publishers' traffic, but before any payments are made), networks adjust their traffic policies to account for any perceived click inflation in each publisher's traffic.

In other words, r_i and V_i are treated as decision variables for publisher i. Recall that click inflation causes an increase in total volume V_i, but leaves G_i unchanged. In Section 2, the fixed set of parameters that described publisher i's traffic (i.e., its "type") was the triple $(r_i, \beta_i^{\text{Pub}}, V_i)$. In this Section, publisher i's organic traffic is described by $(G_i, \beta_i^{\text{Pub}}, A_i)$.

4.2 Solution 1: Estimate r_i

One approach for network j to fight click inflation is to somehow estimate r_i, so that it can pay publisher i for only $r_i V_i c_{ij} = G_i c_{ij}$ valid clicks. One way to derive an estimate of r_i is to: a) run the clicks through a filtering algorithm to observe N_{ij}, b) use labeled "synthetic" traffic to measure u_j and γ_j, and then c) use N_{ij}, u_j and γ_j to invert (2). In this Section, rather than detailing an estimation procedure, we study the effect of estimation errors on a network's profits (since network j's estimation procedure may be highly dependent on the specifics of its ad-serving mechanism).

As described in Section 4.1, the incentive for click inflation arises because network j usually pays each publisher i for $N_{ij} V_i c_{ij} = \left(u_j G_i + u_j^{\gamma_j} B_i\right) c_{ij}$ clicks. The expression for N_{ij} captures the fact that r_i is unknown to network j, and that its filtering algorithms are prone to error. Hypothetically, suppose network j knew the exact value of r_i. It could then simply set $N_{ij} = r_i \ \forall (i, j)$ i.e., pay each publisher i for exactly $r_i V_i c_{ij}$ clicks. Then, (4) would be $E_{ij} = G_i g_{ij} c_{ij}$, which is independent of B_i, implying that publisher i would gain nothing from engaging in click inflation.

Operationally, network j would then compute its best-response traffic policy by simply replacing the sixth constraint in (7) with:

$$N_{ij} = r_i \ \forall (i, j) \tag{12}$$

With this motivation in mind, network j can try to estimate r_i for each publisher i.

The extent to which network j can deter click inflation will depend on how accurately it can estimate r_i. Inaccurate estimates mean that network j solves (7) with incorrect coefficients, and so the policy output by TrafficQuality may be suboptimal. We ran an experiment to quantify the sensitivity of network 1's profits to errors in estimating r_i. For concreteness, suppose network 1 uses filtering to derive a noisy estimate, $\hat{r}_i$, of r_i for each publisher i:

$$\hat{r}_i = r_i + \sigma Z_i \tag{13}$$

Each Z_i is an independent, zero-mean, unit-variance normal random variable ($\hat{r}_i$ is truncated so that it is between 0 and 1). Smaller (larger) values of the standard error, σ, mean that network 1 is more (less) accurate at estimating r_i.

The value of σ was varied between 0% and 40%, and several trials were run at each value. In each trial, network 1 first uses filtering to derive a set of estimates $\{\hat{r}_i \ \forall i\}$ (the estimation error is given by (13)). Network 1 then sets $N_{ij} = \hat{r}_i$ in (7), and uses TrafficQuality to compute a best-response pricing policy $(\hat{\mathbf{g}}_1, \hat{h}_1)$. Of course, $(\hat{\mathbf{g}}_1, \hat{h}_1)$ may be suboptimal since $\hat{r}_i \neq r_i$ (although $r_i \beta_i^{\text{Pub}}$ can still be estimated accurately, as discussed in Section 3.2). Network 1 assumes that network 2 is setting $g_{i2} \propto r_i \beta_i^{\text{Pub}}$. We then compute the actual profit to network 1 resulting from using $(\hat{\mathbf{g}}_1, \hat{h}_1)$ and $N_{ij} = \hat{r}_i$. When computing actual profits, network 2 is assumed to know r_i and β_i^{Pub} exactly.

Figure 3 shows network 1's profits in each trial, as well as the average profit across the trials for each value of σ. Profits have been normalized by network 1's profit in the $\sigma = 0\%$ case. As expected, from Figure 3 we

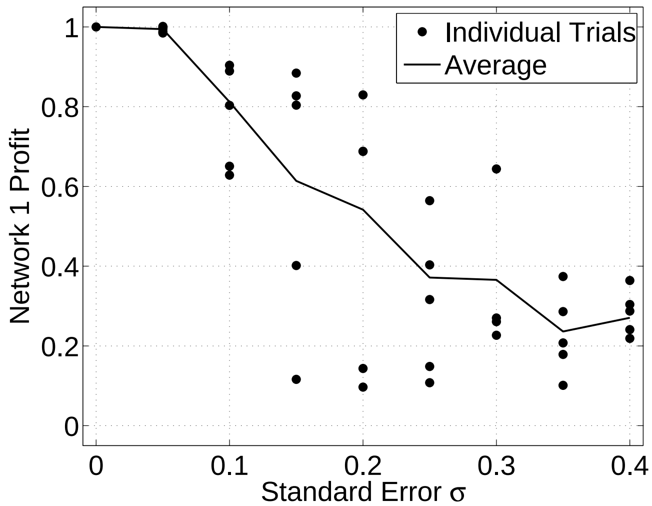

Figure 3: Sensitivity of profits to accuracy of $\hat{r}_i$.

see that lower values of σ (i.e., higher estimation accuracy) result in higher profits for network 1. For $\sigma \leq 10\%$, the losses are relatively small (less than 10% loss in some trials). For $\sigma > 10\%$, however, the variance in the outcome increases greatly, due to estimation errors. The average losses also steadily increase, exceeding 60% at $\sigma = 30\%$.

We conclude, in this example, that the estimates $\hat{r}_i$ are useful only when $\sigma \leq 10\%$ or so. There is one more issue: in practice, networks can only refuse to pay publishers for clicks that their algorithm marks invalid. That is, networks cannot refuse payment for a given click-through, without providing the publisher a justification for doing so. To utilize its estimates of r_i, network j would take the following steps:

1. Use filtering to derive estimates $\hat{r}_i$, but mark all clicks valid irrespective of what the filter decides.
2. Use TrafficQuality with $N_{ij} \leftarrow \hat{r}_i$ to compute predictive prices $\mathbf{g}_j^*$ and revenue share h_j. Effectively, network j is computing a pricing policy for a setting where all clicks are valid.
3. Apply $g_{ij} \leftarrow g_{ij}^* \hat{r}_i$ to each publisher i's traffic.

Observe that network j does not actually mark any clicks invalid. The desired effect of paying for only $G_i c_{ij}$ clicks is achieved indirectly using predictive prices $g_{ij} = g_{ij}^* \hat{r}_i$.

To summarize, if network j could determine r_i exactly, it would use TrafficQuality with $N_{ij} \leftarrow r_i$ to compute its traffic policy. There is zero loss in network j's profits due to click inflation since no new restrictions are placed on $\mathbf{g}_j$ (compare this situation to Section 4.3). Therefore, when fighting click inflation using estimates $\hat{r}_i$ of r_i, any and all losses are purely due to inaccurate estimates. As we discuss in Section 5, estimating r_i accurately is very different (and perhaps much easier) that being skilled at discerning the validity of individual clicks.

4.3 Solution 2: Quasi-CPA

An alternate approach to fighting click inflation is to directly constrain the search for predictive prices in a way that eliminates the incentive, as we describe in this Section. The main advantage of doing so is that no estimates of r_i are needed. As we will demonstrate, this approach is closely related to *cost-per-action* (CPA) pricing schemes. It is widely agreed-upon that CPA schemes are resistant to click inflation. They are susceptible to other forms of fraud, however – we refer the reader to [3, 9, 12] for more details.

Suppose network j simply assumes that all clicks are valid i.e., that $r_i = 1\ \forall i$. The only other measure of publisher quality would then be the conversion rate, β_i^{Pub}. Using the approach discussed in Section 3.2, network j could simply compute an estimate $\hat{\beta}_i$ of β_i^{Pub} as $\hat{\beta}_i = r_i\beta_i^{\text{Pub}}$. Clearly, $\hat{\beta}_i$ would be an underestimate of β_i^{Pub}, since $r_i \in [0, 1]$. Let us assume in this Section that the publishers are sorted by $\hat{\beta}_i$ (i.e., $\hat{\beta}_i$ is increasing in i).

We can interpret a vector of predictive prices $\mathbf{g}_j$ as samples of a continuous function $g_j(\beta)$. More specifically, $\mathbf{g}_j$ is simply the continuous function $g_j(\beta)$ sampled at the I points $\{\hat{\beta}_i, i \in 1, \ldots, I\}$. These I points could then be interpolated smoothly to reconstruct the function $g_j(\beta)$ over the domain $[0, 1]$. Figure 1 in Section 3 is an example.

Let $u_j = 1$ (due to Theorem 1), and define $e_{ij} \equiv c_{ij}h_j\theta_j$ for convenience. Interpreting g_{ij} as the function $g_j(\beta)$ evaluated at the point $\hat{\beta}_i$, we can then rewrite (5) as follows:

$$\begin{aligned} \pi_{ij} &= N_{ij}V_i c_{ij} h_j \theta_j g_{ij} \\ &= e_{ij} V_i g_j(\hat{\beta}_i) \end{aligned} \tag{14}$$

Recall that π_{ij} is the revenue earned by publisher i from traffic sent to network j.

To simplify our discussion, let us assume in this Section that $\beta_j^{\text{Net}} = \beta_k^{\text{Net}} = 1\ \forall (j, k)$ (the results that follow do not depend on this assumption). Then, as discussed in Section 3.2, network j can compute $\hat{\beta}_i = r_i\beta_i^{\text{Pub}}$ by just counting up the number of conversions, A_i, and dividing by the volume of clicks, V_i (see (11)). Therefore, we can write (14) as:

$$\pi_{ij} = e_{ij}\frac{A_i}{\hat{\beta}_i}g_j(\hat{\beta}_i) \tag{15}$$

Clearly, the incentive for click inflation would be eliminated if π_{ij} were non-increasing in B_i. Since $V_i = G_i + B_i$ and $\hat{\beta}_i$ is inversely proportional to V_i, it is sufficient that π_{ij} is non-decreasing in $\hat{\beta}_i$. We can therefore differentiate (15) and impose a non-negativity condition:

$$\frac{\partial \pi_{ij}}{\partial \hat{\beta}_i} = e_{ij}A_i\left(\frac{g_j'(\hat{\beta}_i)}{\hat{\beta}_i} - \frac{g_j(\hat{\beta}_i)}{\hat{\beta}_i^2}\right) \geq 0 \tag{16}$$

Simplifying, we get:

$$g_j'(\hat{\beta}_i) \geq \frac{g_j(\hat{\beta}_i)}{\hat{\beta}_i} \tag{17}$$

To eliminate the incentive for click inflation, network j should select predictive prices g_{ij} such that (17) holds for all $\hat{\beta}_i$. Intuitively, a high-quality publisher i should not feel tempted to "masquerade" (i.e., by engaging in click inflation) as any other lower-quality publisher just because the latter is "getting a better deal" than the former. We can achieve this effect by approximating the derivative in (17) with a backward difference between publishers i and $i - 1$:

$$g_j'(\hat{\beta}_i) \approx \frac{g_{ij} - g_{i-1,j}}{\hat{\beta}_i - \hat{\beta}_{i-1}} \tag{18}$$

Simplifying again, we arrive at:

$$\frac{g_{i-1,j}}{g_{ij}} \leq \frac{\hat{\beta}_{i-1}}{\hat{\beta}_i} \tag{19}$$

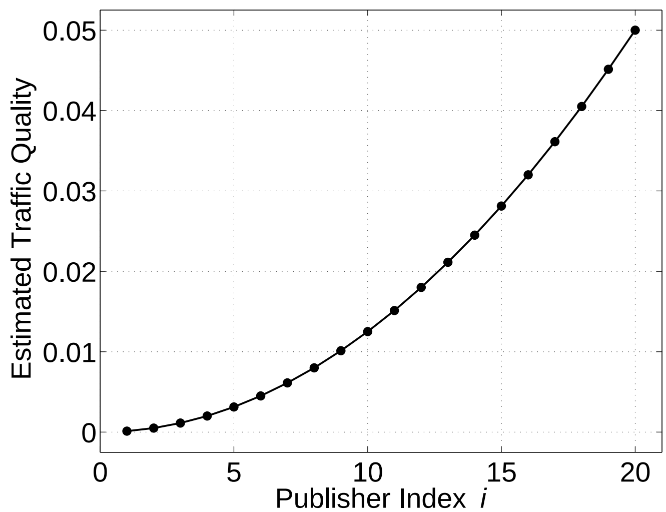

Figure 4: Estimated traffic quality, $\hat{\beta}_i = r_i \beta_i^{\text{Pub}}$.

Therefore, we can simply add the constraint (19) to the optimization problem (7), for each pair of publishers i and $i-1$. It is very convenient that (19) can be added as-is to a geometric program (i.e., without using an approximation), meaning we can enforce these constraints exactly using TrafficQuality (see Appendix A).

The constraint (19) has a very interesting form. At equality, it forces g_{ij} to be proportional to $\hat{\beta}_i$ i.e., $g_{ij} = \delta\hat{\beta}_i$ for some constant δ. Substituting $g_{ij} = \delta\hat{\beta}_i$ into (4), and using (11), we find that the effective number of clicks publisher i is paid for by network j is simply $E_{ij} = \delta c_{ij} A_i$. That is, E_{ij} is directly proportional to the number of clicks that become conversions. We have, essentially, a CPA pricing scheme. It can even be shown that the expected revenues to each player in the market are equal to those that would be obtained from a CPA pricing scheme (we omit the details here). Therefore, what the constraint (19) tells us is that CPA is just a special case within the set of traffic policies that eliminate the incentive for click inflation. For this reason, we refer to traffic policies that enforce (19) as *quasi-CPA* policies.

Consider a scenario with $I = 20$ publishers and $J = 2$ networks. The fraction of valid traffic and the conversion rate for each publisher i is linear in i ($r_i = 0.05i$ and $\beta_i^{\text{Pub}} = 0.0025i$). This scenario is the same as the one considered in Figure 1, except that we will now allow h_1 to be a decision variable. Figure 4 is a plot of the product $\hat{\beta}_i = r_i \beta_i^{\text{Pub}}$ for each publisher i in this example – there are many low-quality publishers and relatively few high-quality publishers. Figure 5 shows the predictive prices that are recommended by TrafficQuality with and without the quasi-CPA constraint (19) included in the optimization problem (7). For example, the predictive price chosen for publisher 15 in the quasi-CPA case is $g_{ij} = 0.5625$, compared to $g_{ij} = 0.9843$ for the unconstrained case. From Figures 4 and 5, we see that the quasi-CPA constraint has yielded predictive prices that are (almost exactly) proportional to $\hat{\beta}_i$, whereas the unconstrained profile has a very different shape.

The unconstrained policy tries to dissuade low-quality publishers 1 through 7 by applying drastically lower predictive prices to them compared to medium- and high-quality publishers (8 through 20). The quasi-CPA policy only allows network 1 to capture traffic from publishers 12 through 20 – the medium-quality publishers 8 through 11 are penalized too harshly due to (19), so they choose network 2 instead.

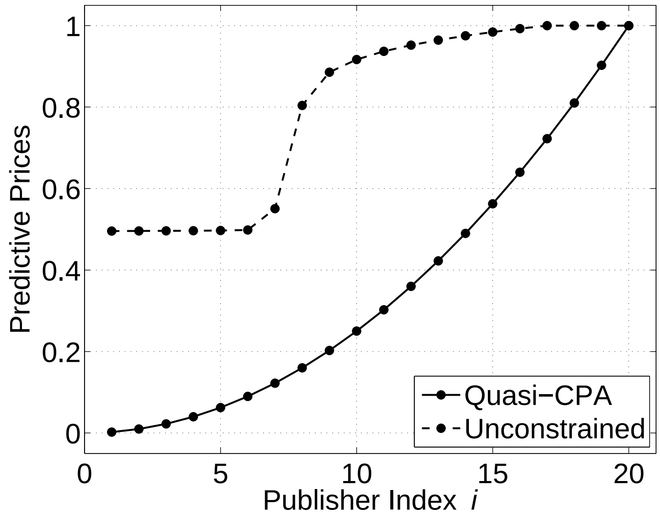

Figure 5: Effect of (19) on predictive prices.

4.4 Comparison

In Sections 4.2 and 4.3, we presented two different approaches for removing the incentive for click inflation. Which approach should a network use?

The answer depends completely on how accurately the network is able to estimate r_i. In the example in Section 4.3, network 1's profit using a quasi-CPA policy turns out to be 79% of the profit using the unconstrained policy (which could be used if there was no click inflation, or if r_i could be estimated exactly). From the experiment in Section 4.2 (see Figure 3), we see that average profits exceed 80% when the standard estimation error $\sigma \leq 10\%$. Therefore, in this instance, network 1 should use its estimates of r_i if $\sigma \leq 10\%$. Otherwise, a quasi-CPA traffic policy should be used.

In general, there will be a threshold value, say σ^*, such that if $\sigma \leq \sigma^*$, estimating r_i directly would be more profitable on average for network 1. The actual value of the threshold, of course, will be need to be identified by each network in practice.

5 Discussion

We conclude this paper by briefly revisiting some issues that were raised in Sections 1-4.

Yahoo! and Google. There is anecdotal evidence that Yahoo! and Google employ predictive pricing profiles qualitatively similar to the unconstrained case in Figure 5, i.e., a few low-quality publishers being punished severely, and the rest not even noticing the effects of predictive pricing. Just search the web for "quality-based pricing" or "smart pricing", and read the complaints made by publishers about "getting smart priced".

Filtering. The results of Section 4.2 suggest that when it comes to filtering, a network's competitive advantage arises from accurately characterizing the *aggregate* performance of its filtering algorithm (i.e., knowing u_j and γ_j), rather than correctly deciding on the validity of individual clicks. Even though network j's filtering algorithm makes incorrect decisions on a large number of individual clicks, it can compensate for these errors in aggregate by simply adjusting its predictive prices. Therefore, networks should devote engineering resources to estimating the fraction r_i accurately, rather than trying to mark individual clicks valid or invalid.

Advantages of quasi-CPA over CPA. Since much of the online advertising industry (including some of the largest networks) currently operates on a CPC basis, there may be large costs and risks associated with switching to a CPA pricing scheme. Using a quasi-CPA traffic policy allows a network to reap the benefits of a CPA scheme (i.e., no click inflation) while maintaining a CPC infrastructure. Another very important reason publishers and networks shy away from (and advertisers prefer) CPA schemes is that they transfer the risk of organic non-converting traffic entirely away from the advertisers (and on to publishers and networks). Using a CPC-based quasi-CPA policy enables a more equitable distribution of risk.

6 Conclusion

In this paper, we discussed how filtering, predictive pricing and revenue sharing can be used together to influence the quality of traffic delivered by a CPC advertising network. Managing traffic quality is critical, since quality uncertainty and asymmetric information between publishers and advertisers can destroy value (the "lemons market" effect).

We drew an important distinction between organic traffic and publisher-initiated click inflation. If possible, predictive pricing should be chosen in favour of filtering to manage organic traffic quality. To fight click inflation, either filtering or predictive pricing can used, depending on how well a network can characterize the performance of its filtering algorithm. In either case, it is important to remember that eliminating the incentive for click inflation does not mean that invalid traffic will disappear – many forms of invalid traffic are, in fact, organic.

References

[1] G. Akerlof. The market for "lemons": Quality uncertainty and the market mechanism. *Quarterly Journal of Economics*, 84(3):488–500, 1970.

[2] S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi. A tutorial on geometric programming. *Optimization and Engineering*, 8(1):67–127, 2007.

[3] N. Daswani, C. Mysen, V. Rao, S. Weis, K. Gharachorloo, and S. Ghosemajumder. *Crimeware: Understanding New Attacks and Defenses*, chapter Online Advertising Fraud. Addison-Wesley Professional, 2008.

[4] N. Daswani and M. Stoppelman. The anatomy of clickbot.A. In *Hot Topics in Understanding Botnets (HotBots)*. Usenix, April 2007.

[5] M. Gandhi, M. Jakobsson, and J. Ratkiewicz. Badvertisements: Stealthy click-fraud with unwitting accessories. *Journal of Digital Forensic Practice*, 1(2):131–142, 2006.

[6] Google. The facts about smart pricing. *Google AdSense Blog*, October 2005.

[7] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming (web page and software), June 2008. `http://stanford.edu/~boyd/cvx`.

[8] N. Immorlica, K. Jain, M. Mahdian, and K. Talwar. Click fraud resistant methods for learning click-through rates. In *Internet and Network Econonomics*, volume 3828 of *Lecture Notes in Computer Science*, pages 34–45. Springer, 2005.

[9] M. Mahdian and K. Tomak. Pay-per-action model for online advertising. In *Internet and Network Econonomics*, volume 4858 of *Lecture Notes in Computer Science*, pages 549–557. Springer, 2007.

[10] B. Mungamuru and H. Garcia-Molina. Predictive pricing and revenue sharing. *Stanford Infolab Technical Report*, July 2008.

[11] B. Mungamuru, S. Weis, and H. Garcia-Molina. Should ad networks bother fighting click fraud? (Yes, they should.). *Stanford Infolab Technical Report*, 2008.

[12] H. Nazerzadeh, A. Saberi, and R. Vohra. Dynamic cost-per-action mechanisms and applications to online advertising. In *ACM Conference on the World Wide Web*, April 2008.

[13] A. L. Penenberg. Click fraud threatens web. *Wired News*, October 2004.

[14] E. Savitz. Eric Schmidt: Click fraud is self correcting. Mark Cuban: No, it isn't. *Barron's*, July 2006.

[15] B. Schneier. Google's click-fraud crackdown. *Wired News*, July 2006.

[16] S. Shankland. Yahoo to try using Google for search ads. *CNET News*, April 2008.

[17] Yahoo! A new pricing model rolls out today. *Yahoo! Search Marketing Blog*, June 2007.

A TrafficQuality Algorithm

In this Appendix, we describe the TrafficQuality algorithm for finding approximate solutions to (7). Let $\mathbf{u}_{-1} \equiv \{u_j \ \forall j \neq 1\}$, $\mathbf{G}_{-1} \equiv \{\mathbf{g}_j \ \forall j \neq 1\}$ and $\mathbf{h}_{-1} \equiv \{h_j \ \forall j \neq 1\}$ denote the actions of network 1's competitors. Let $\mathbf{C} \equiv \{c_{ij} \ \forall (i,j)\}$ denote the publishers' allocations in the second stage, and let $\mathbf{c}_i$ be column i of $\mathbf{C}$.

Algorithm 1 TrafficQuality

Require: $\mathbf{u}_{-1}$, $\mathbf{G}_{-1}$, $\mathbf{h}_{-1}$, T

Select arbitrary initializations $u_1^{(1)}$, $\mathbf{g}_1^{(1)}$ and $h_1^{(1)}$

Use fixed-point iteration to compute the second-stage equilibrium, $\mathbf{C}^{(1)}$, assuming other networks play $(\mathbf{u}_{-1}, \mathbf{G}_{-1}, \mathbf{h}_{-1})$ and network 1 plays $\left(u_1^{(1)}, \mathbf{g}_1^{(1)}, h_1^{(1)}\right)$

for $t \in 1, \ldots, T-1$ **do**

  Solve a geometric program relaxation of (7) to find an optimal point $(u_1', \mathbf{g}_1', h_1', \mathbf{C}')$ that is ϵ-"close to" $\left(u_1^{(t)}, \mathbf{g}_1^{(t)}, h_1^{(t)}, \mathbf{C}^{(t)}\right)$, assuming other networks play $(\mathbf{u}_{-1}, \mathbf{G}_{-1}, \mathbf{h}_{-1})$

  $\left(u_1^{(t+1)}, \mathbf{g}_1^{(t+1)}, h_1^{(t+1)}, \mathbf{C}^{(t+1)}\right) \leftarrow (u_1', \mathbf{g}_1', h_1', \mathbf{C}')$

end for

Use fixed-point iteration to recompute the second-stage equilibrium, $\mathbf{C}^{(T)}$, assuming other networks play $(\mathbf{u}_{-1}, \mathbf{G}_{-1}, \mathbf{h}_{-1})$ and network 1 plays $\left(u_1^{(T)}, \mathbf{g}_1^{(T)}, h_1^{(T)}\right)$

return $\left(u_1^{(T)}, \mathbf{g}_1^{(T)}, h_1^{(T)}, \mathbf{C}^{(T)}\right)$

Most of the computation in TrafficQuality is done inside the `for` loop. In each iteration t of the loop, we are given as input $(\mathbf{u}_{-1}, \mathbf{G}_{-1}, \mathbf{h}_{-1})$ and a point $\left(u_1^{(t)}, \mathbf{g}_1^{(t)}, h_1^{(t)}, \mathbf{C}^{(t)}\right)$. We first compute $N_{i1}^{(t)} = u_1^{(t)} r_i + \left(u_1^{(t)}\right)^{\gamma_1}(1-r_i)\ \forall i$ and $N_{ij} = u_j r_i + u_j^{\gamma_j}(1-r_i)\ \ \forall (i, j \neq 1)$. We then define:

$$\mathbf{w}_j^{(t)} \equiv \{w_{ij}^{(t)}\} \qquad \text{where } \ w_{ij}^{(t)} \equiv r_i V_i \beta_i^{\text{Pub}} \kappa_j \beta_j^{\text{Net}} c_{ij}^{(t)}$$

$$\mathbf{z}_j^{(t)} \equiv \{z_{ij}^{(t)}\} \qquad \text{where } \ z_{ij}^{(t)} \equiv \begin{cases} N_{i1}^{(t)} V_i c_{i1}^{(t)} g_{i1}^{(t)} & j = 1 \\ N_{ij} V_i c_{ij}^{(t)} g_{ij} & j \neq 1 \end{cases}$$

Using these quantities, we solve the following geometric program [2], which is a relaxation of (7) around the input point $\left(u_1^{(t)}, \mathbf{g}_1^{(t)}, h_1^{(t)}, \mathbf{C}^{(t)}\right)$:

$$\begin{aligned}
\text{maximize} \quad & d_1 p_1 \\
\text{subject to} \quad & d_1 \leq 1 - h_1 \\
& X_{ij} = N_{ij} V_i g_{ij} h_j \theta_j \ \ \forall (i,j) \\
& c_{ij} \leq (1+\epsilon) \frac{X_{ij}^{\alpha}}{\sum_m X_{im}^{\alpha}} \ \ \forall (i,j) \\
& 1 - \epsilon \leq m\left(\mathbf{c}_i; \mathbf{c}_i^{(t)}\right) \ \ \forall i \\
& \sum_j c_{ij} \leq 1 + \epsilon \ \ \forall i \\
& \theta_j = \frac{p_j}{q_j} \ \ \forall j \\
& (1-\epsilon) p_j \leq m\left(\mathbf{w}_j; \mathbf{w}_j^{(t)}\right) \ \ \forall j \\
& \sum_i w_{ij} \leq (1+\epsilon) p_j \ \ \forall j \\
& (1-\epsilon) q_j \leq m\left(\mathbf{z}_j; \mathbf{z}_j^{(t)}\right) \ \ \forall j \\
& \sum_i z_{ij} \leq (1+\epsilon) q_j \ \ \forall j \\
& w_{ij} = r_i V_i \beta_i^{\text{Pub}} \kappa_j \beta_j^{\text{Net}} c_{ij} \ \ \forall (i,j) \\
& z_{ij} = N_{ij} V_i c_{ij} g_{ij} \ \ \forall (i,j) \\
& (1-\epsilon) N_{i1} \leq m_u\left(u_1; u_1^{(t)}, r_i\right) \ \ \forall i \\
& u_1 r_i + u_1^{\gamma_1}(1 - r_i) \leq (1+\epsilon) N_{i1} \ \ \forall i \\
& u_1, g_{i1}, h_1, c_{ij} \leq 1 \ \ \forall (i,j)
\end{aligned} \tag{20}$$

In (20), $\epsilon > 0$ and $\alpha \gg 0$ are fixed parameters. Given a vector $\mathbf{x}^{(t)} \equiv \left\{x_i^{(t)}\right\}$ and a generic point $\mathbf{x} \equiv \{x_i \ \forall i\}$, the function $m\left(\mathbf{x}; \mathbf{x}^{(t)}\right)$ is a monomial approximation of $\sum_i x_i$ about the point $\mathbf{x}^{(t)}$. Similarly, $m_u\left(u; u^{(t)}, r\right)$ is a monomial approximation of $ur + u^{\gamma_1}(1-r)$ about the point $\left(u^{(t)}, r\right)$. To solve (20) in our experiments, we used `CVX`, a package for solving convex programs [7].