Predictive Pricing and Revenue Sharing

Bobji Mungamuru
Stanford University
bobji@i.stanford.edu

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

July 24, 2008

Abstract

Predictive pricing (e.g., Google's "Smart Pricing" and Yahoo's "Quality-Based Pricing") and *revenue sharing* are two important tools that online advertising networks can use in order to attract content publishers and advertisers. We develop a simple model of the pay-per-click advertising market to study the market effects of these tools. We then present an algorithm, PricingPolicy, for computing an advertising network's best response i.e., given the predictive pricing and revenue sharing policies used by its competitors, what policy should an advertising network use in response? Using PricingPolicy, we gain insight into the structure of optimal predictive pricing and revenue sharing policies.

1 Introduction

Google's "Smart Pricing" [6] and Yahoo's "Quality-Based Pricing" [12] are examples of a practice we refer to as *predictive pricing*. The idea behind predictive pricing in pay-per-click advertising is to charge the same advertiser different prices for click-throughs, depending on which publisher the click-through originated from. For example, an advertiser who bid on the keyword "camera" might be charged less for a click-through from a travel website than one from a photographer's blog, since the latter would (ostensibly) be more targeted to potential camera purchasers than the former. Advertising networks use predictive pricing to attract publishers and advertisers to their network.

Revenue sharing, which is the practice of paying out a fraction of earned revenues to the publishers where click-throughs originate, is another tool used by advertising networks to attract traffic. Revenue sharing is the reason publishers display advertisements alongside their content in the first place. In this paper, we study how an online advertising network can apply predictive pricing and revenue sharing "optimally" – that is, in a manner that maximizes the advertising network's profits.

The sheer size of the online advertising market makes this problem interesting and important. Although 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" (either by paying out an unnecessarily large revenue share, or by attracting less- or lower-quality traffic than they could be). And in a market that, by most estimates (e.g., [5]), is worth several billions of dollars, the losses due to suboptimal pricing policies can be tremendous. Advertising networks that currently do not apply predictive pricing should feel compelled to start – our results suggest that they are yielding a significant advantage to their competitors.

Finding an optimal pricing policy is difficult because models of the online advertising market can quickly become very complex. The challenge is to capture just those aspects that impact pricing policy decisions. Our model is game-theoretic, necessitating the computation of equilibria (a task that is known to be difficult). Worse yet, the optimization problems involved in computing equilibria are highly non-convex. Thus, we face significant challenges from both modeling and computational perspectives.

The practice of predictive pricing in the pay-per-click advertising market is relatively new. To the authors' knowledge, there has been no formal analysis thus far of how to apply predictive pricing and share revenue optimally. We suspect that it is currently being applied in an ad hoc manner. Perhaps there has been no need for a principled approach – the fact that any predictive pricing was being done at all may have been sufficient to satisfy publishers and advertisers. However, as more networks adopt such programs, we feel that a principled approach will become necessary. Recent research on "click quality" has focused on a related (but orthogonal) problem i.e., *click fraud* [3, 4, 9, 10]. Click fraud relates to whether a given click-through is valid or invalid. Predictive pricing, on the other hand, focuses on the probability that a valid click-through becomes a conversion i.e., the conversion rate. Also, techniques for fighting click fraud are typically not applied on a per-publisher basis (apart from simple blacklisting). Predictive pricing, on the other hand, allows for very fine-grained publisher-level control.

1.1 Overview

We begin by constructing a model of the online advertising market as a game between content publishers, advertising networks and advertisers. The model is a simplification of what happens in practice – our intent is to hone in on the market effects of predictive pricing and revenue sharing decisions. We then derive an expression for an advertising network's *best-response function*. That is, if an advertising network knows the predictive pricing and revenue sharing policies of its competitors, what policy should the network choose in response, in order to maximize its profits? The expression we derive for the best-response function is implicit – it is the solution to a difficult optimization problem. We then present an algorithm, PricingPolicy, for solving this optimization problem, yielding a near-optimal predictive pricing and revenue sharing policy.

Finally, we apply PricingPolicy toward answering some qualitative questions about predictive pricing:

- Is it always optimal to charge less for lower-quality traffic? (Yes.)
- Should an advertising network always try to attract as much traffic as it can, regardless of traffic quality? (No.)
- If a network is better at targeting, can it offer a lower revenue share? (Yes.)
- Does predictive pricing harm publishers, as has been conjectured in online forums? (Yes and no – it harms low-quality publishers and helps high-quality publishers.)

In principle, the best-response function can be used as a "subroutine" for computing equilibrium policies for advertising networks (an equilibrium is, by definition, a fixed point of the networks' best-response functions). However, we believe that the practical value of our algorithm lies in computing best responses, rather than equilibria. It prescribes actions that networks can take "today" in response to their competitors, rather than waiting for equilibria to unfold. Thus, our focus will be on finding best responses.

2 Model

We model the *pay-per-click* (PPC) advertising 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 on keywords that describe the interests of their target market. *Advertising networks* (or, *networks*) act as intermediaries, auctioning off click-throughs (or, *clicks*) to advertisers and delivering relevant ads to publishers upon request.

If a user visits a publisher's site and clicks on an ad, 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. Predictive pricing affects how much the advertiser is billed by the network, whereas the revenue share determines what fraction of this revenue the network will pay out to the publisher. A small fraction of clicks become *conversions* e.g., a purchase, or a sign-up to an email list. The advertiser earns some revenue each time a click becomes a conversion.

Our dynamic game is comprised of two steps:

1. In the *first step*, networks select and announce their predictive pricing and revenue sharing policies.

2. In the *second step*, publishers decide which networks to sell their clicks on, and advertisers decide how much they are willing to pay for clicks from each network.

After the second step, payoffs are realized: a) publishers sell clicks (i.e., display ads) on their chosen networks, and b) advertisers pay the networks, who then pay the publishers. We consider a one-shot game, although the extension to a multi-period model is straightforward. Appendix B contains a summary of the notation used in this paper.

Consider the market for click-throughs on a single keyword. 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[1]. Let c_{ij} be the fraction of these clicks that publisher i sends to network j. Then,

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

is the total number of clicks that publisher i sends to network j.

For simplicity, we will ignore click fraud in this paper. That is, we assume that all clicks are valid and that networks mark all clicks valid ($r_i = 1\ \forall i$ and $N_{ij} = 1\ \forall (i, j)$, using the notation of [10]). Our results are in no way dependent on this assumption. For example, if we were to account for click fraud, (1) would be $N_{ij} V_i c_{ij}$, where N_{ij} is the fraction of publisher i's clicks marked valid by network j.

For each click coming from publisher i, 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 pricing factor*[2] that network j applies

[1]Each publisher in our model has an inventory of clicks to allocate across the networks. In practice, however, publishers allocate impressions (or, "page views"), not clicks. In [10], we include a detailed discussion of the conditions under which it is equivalent to model clicks (rather than impressions) as the objects being bought and sold.

[2]The term "predictive pricing" alludes to network j's prediction about the quality of publisher i's traffic (i.e., accounting for click-through rates, click fraud and conversion rates).

to publisher i's traffic. The *effective* number of clicks publisher i is paid for by network j is then:

$$V_i c_{ij} g_{ij} \tag{2}$$

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*. We refer to $\{g_{ij}\ \forall i\}$ and h_j together as network j's *pricing policy*.

Let θ_j be the expected auction revenue per click on network j. That is, if network j were to auction off Z clicks, its total expected revenue would be $Z\theta_j$. The value of θ_j depends on the auction mechanism used by network j, as well as all the advertisers' bids. Then, the revenue to publisher i from network j is:

$$\pi_{ij} \equiv V_i c_{ij} g_{ij} h_j \theta_j \tag{3}$$

The total revenue to i across all networks is:

$$\pi_i \equiv \sum_j \pi_{ij} \tag{4}$$

Of all the clicks sent to network j, a fraction ξ_{jk} is sent on to advertiser k. The fraction ξ_{jk} depends on network j's auction mechanism, as well as all of the advertisers' bids. We assume that ξ_{jk} does not depend on i i.e., the posterior probability that a given click originated on publisher i's site does not depend on the advertiser that received the click. As we demonstrate later, we will never need to actually compute the value of ξ_{jk}. Of the clicks going from publisher i to network j to advertiser k, let β_{ijk} be the fraction that become conversions i.e., the *conversion rate*. The number of clicks converted by advertiser k that came from publisher i via network j is then:

$$V_i c_{ij} \xi_{jk} \beta_{ijk} \tag{5}$$

Let y_k be the revenue that advertiser k earns from each conversion. The total revenue to advertiser k from conversions of clicks from network j, across all publishers, is then:

$$Y_{kj} \equiv \left(\sum_i V_i c_{ij} \xi_{jk} \beta_{ijk} \right) y_k = \left(\sum_i V_i c_{ij} \beta_{ijk} \right) \xi_{jk} y_k \tag{6}$$

Using (2), the effective number of clicks originating from publisher i that advertiser k is billed for by network j is:

$$V_i c_{ij} g_{ij} \xi_{jk} \tag{7}$$

The total number of clicks advertiser k is billed for by network j is then:

$$Z_{kj} \equiv \left(\sum_i V_i c_{ij} g_{ij} \right) \xi_{jk} \tag{8}$$

Let v_{kj} be advertiser k's valuation for network j's clicks i.e., v_{kj} is what advertiser k is willing to pay network j per click. The total amount that advertiser k is willing to pay network j is then:

$$Z_{kj} v_{kj} \tag{9}$$

Advertiser k's *return on investment* (ROI) on clicks from network j would therefore be:

$$R_{kj} \equiv \frac{Y_{kj}}{Z_{kj} v_{kj}} = \frac{\left(\sum_i V_i c_{ij} \beta_{ijk}\right) y_k}{\left(\sum_i V_i c_{ij} g_{ij}\right) v_{kj}} \tag{10}$$

Note that we are differentiating between bids and valuations here. Network j does not know v_{kj}, so it runs auctions to extract this information. Advertiser k's bid in this auction does not necessarily have to be v_{kj}. Network j's expected per-click auction revenue, θ_j, is a function of $\{v_{kj}\ \forall k\}$ and the auction mechanism used by j.

Finally, network j's total profit, η_j, is the amount collected from advertisers less the amount paid out to publishers. Therefore:

$$\eta_j \equiv \left(\sum_i V_i c_{ij} g_{ij}\right)(1-h_j)\theta_j = \frac{1-h_j}{h_j}\sum_i \pi_{ij} \tag{11}$$

2.1 Assumptions

Separable Conversion Rates. We assume that conversion rates are *separable* i.e., that each β_{ijk} is a product of three factors:

$$\beta_{ijk} = \beta_i^{\text{Pub}} \beta_j^{\text{Net}} \beta_k^{\text{Adv}} \quad \forall (i,j,k) \tag{12}$$

Each factor in (12) has a different interpretation[3]. 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. From (6), separability of conversion rates implies:

$$Y_{kj} = \beta_j^{\text{Net}} \left(\sum_i V_i c_{ij} \beta_i^{\text{Pub}}\right) \xi_{jk} \beta_k^{\text{Adv}} y_k \tag{13}$$

Linear Auctions. We assume that every network uses an auction that is *linear* in the following sense: if all agents' valuations are scaled by a factor γ, then the expected revenue from the auction is also scaled by γ. First-price, second-price, Dutch and English auctions can all be shown to have this property. The maximal and minimal equilibrium revenues for the position auction in [11] and the generalized second-price auction in [5] are also linear in this sense.

We are not assuming that all networks use the same auction mechanism, or even that the mechanisms are truthful – only that each auction is linear. The linearity assumption will allow us to derive an explicit expression for θ_j (see (19)).

2.2 Publishers' and Advertisers' Objectives

In the first step, network j chooses its pricing policy (i.e., h_j and $\{g_{ij}\ \forall i\}$) such that its profit, η_j, is maximized. We discuss network j's optimization problem in Section 3.

In the second step, publisher i chooses allocations $\{c_{ij}\ \forall j\}$ such that the total revenue generated from its

[3] A related "separability" assumption is made in [1] and [11], where the click-through rate (CTR) for an advertisement is assumed to be the product of an advertiser-specific factor and a position/"slot"-specific factor.

sites is maximized:

$$\text{maximize} \;\; \pi_i \;\; \text{subject to} \;\; \sum_j c_{ij} = 1 \tag{14}$$

At the same time, each advertiser k chooses valuations v_{kj} that maximize its revenue from each network j, subject to a lower bound R_k on ROI:

$$\text{maximize} \;\; Y_{kj} \;\; \text{subject to} \;\; R_{kj} \geq R_k \tag{15}$$

Here, R_k is advertiser k's *target ROI*. Intuitively, R_k is the ROI that advertiser k can achieve by advertising through channels other than PPC. Solving (15) is equivalent to maximizing advertiser k's combined profits from both online and "offline" advertising.

Publishers and advertisers know the networks' pricing policies when they make their decisions in the second step. Publisher i's type is $\left(V_i, \beta_i^{\text{Pub}}\right)$, network j's type is β_j^{Net} and advertiser k's type is $\left(y_k, R_k, \beta_k^{\text{Adv}}\right)$. Publishers' and networks' types are common knowledge, whereas each advertiser k's type is known only to k.

2.3 Publishers' and Advertisers' Best Responses

At the optimum, the constraint in (15) will be binding for each network j:

$$R_{kj} = R_k \;\forall j \tag{16}$$

To understand why (16) holds, assume for a moment that network j's auction is truthful i.e., that advertiser k's bid is v_{kj}. Recall that R_k is the ROI that advertiser k can achieve through channels other than PPC advertising. So, if $R_{kj} > R_k$, advertiser k will want to spend more money on network j i.e., it will want to buy more clicks from network j. To receive more clicks from network j, it must increase its bid v_{kj}. However, from (10), we know that R_{kj} is decreasing in v_{kj}. Therefore, advertiser k will keep increasing v_{kj} as long as $R_{kj} > R_k$, meaning (16) will hold at the optimum. This argument is informal, but can be made rigourous.

From (10) and (16), advertiser k's optimal valuation (i.e., its best response) is:

$$v_{kj} = \beta_j^{\text{Net}} \frac{\left(\sum_i V_i c_{ij} \beta_i^{\text{Pub}}\right)}{\left(\sum_i V_i c_{ij} g_{ij}\right)} \frac{\beta_k^{\text{Adv}} y_k}{R_k} = \bar{v}_k a_j \tag{17}$$

where $\bar{v}_k$ is defined as advertiser k's *nominal valuation*, and a_j is an *adjustment factor* applied to network j:

$$\bar{v}_k \equiv \frac{\beta_k^{\text{Adv}} y_k}{R_k} \qquad\qquad a_j \equiv \beta_j^{\text{Net}} \frac{\left(\sum_i V_i c_{ij} \beta_i^{\text{Pub}}\right)}{\left(\sum_i V_i c_{ij} g_{ij}\right)} \tag{18}$$

Intuitively, a_j is proportional to the ratio between the number of clicks converted on network j and the effective number of clicks billed for, after predictive pricing. The key point is that publishers' and networks' decisions affect v_{kj} only through the multiplicative factor a_j. Moreover, v_{kj} depends on the actions of all publishers, but does not depend on the actions of any other advertisers.

Let κ_j be network j's expected revenue per click assuming $v_{kj} = \bar{v}_k \;\forall k$ i.e., $\theta_j = \kappa_j$ when $v_{kj} = \bar{v}_k$. Note that $\bar{v}_k$ does not depend on j. Therefore, if $\kappa_1 > \kappa_2$, it would mean network 1 is extracting more revenue per click than network 2, from the same set of valuations i.e., network 1's auction is more "efficient" than network

2.

Now, suppose each advertiser k chooses his valuation optimally i.e., (17) holds for all k. Compared to the scenario where $v_{kj} = \bar{v}_k \ \forall k$, each advertiser k's valuation has been scaled up by a factor a_j. The assumption that network j's auction is linear would therefore imply that:

$$\theta_j = \kappa_j a_j \tag{19}$$

From (3) and (4), π_i is linear in publisher i's allocations $\{c_{ij} \ \forall j\}$. The lone constraint in (14) is also linear in c_{ij}. Thus, solutions to (14) have a simple and intuitive form. Let X_{ij} be publisher i's revenue assuming it sends all of its traffic to network j (i.e., $c_{ij} = 1$). From (3), we get:

$$X_{ij} = V_i g_{ij} h_j \theta_j \tag{20}$$

The optimal allocations $\{c'_{ij} \ \forall j\}$ for publisher i (i.e., its best response) satisfy:

$$\sum_j c'_{ij} X_{ij} = \max_j X_{ij} \quad \text{and} \quad \sum_j c'_{ij} = 1 \tag{21}$$

In words, it is optimal for publisher i to send all its traffic to the single network whose X_{ij} value is highest. If there is a tie between two or more networks, publisher i can split its traffic arbitrarily between these networks.

We emphasize that the networks act first and publishers and advertisers second. So, when publishers compute their optimal allocations and advertisers compute their optimal valuations, the networks' actions (i.e., their pricing policies) are known. Therefore, $\{g_{ij} \ \forall (i,j)\}$ and $\{h_j \ \forall j\}$ are treated as constants and not variables in (14) and (15).

For a given first-step outcome $\{g_{ij} \ \forall (i,j)\}$ and $\{h_j \ \forall j\}$, an equilibrium in the second step is defined as a scenario where every advertiser k chooses its valuations $\{v_{kj} \ \forall j\}$ optimally and every publisher i chooses its allocations $\{c_{ij} \ \forall j\}$ optimally i.e., (17), (19) and (21) hold simultaneously for all (i,j,k). Therefore, if an equilibrium is played in the second-step, we can substitute (17) and (19) into (11), and simplify:

$$\eta_j = \beta_j^{\text{Net}} \left(\sum_i V_i \beta_i^{\text{Pub}} c_{ij} \right) (1 - h_j) \kappa_j \tag{22}$$

Interestingly, the predictive pricing factors g_{ij} do not appear anywhere in (22), although they affect publisher allocations and advertiser valuations (see (17), (20) and (21)).

3 Optimal Pricing Policies

Network j's goal is to maximize its profit, η_j. From (22), η_j depends on the decisions made by publishers and advertisers. However, the networks act first in our game. Publishers and advertisers observe the networks' decisions in the first step before deciding on their allocations and valuations in the second step. In other words, the outcome in the second step (i.e., allocations and valuations) is the market's reaction to first-step outcome (i.e., networks' pricing policies). Therefore, to maximize revenue, each network j will: a) assume that an equilibrium will be played in the second step, and b) choose a pricing policy that induces the most profitable equilibrium in the second step.

The second-step outcome depends not only on network j's pricing policy, but also on the pricing policies chosen by competing networks in the first step. For example, if the revenue share h_j offered by network j is too low, then very few publishers may send traffic to j (i.e., $c_{ij} = 0$ for most i), leading to a low η_j. If h_j were too high, more publishers may send traffic to network j, but η_j might be low again since j would be paying out a large fraction of revenues to publishers. Therefore, network j must account for the actions of all other networks when choosing its own pricing policy.

We will now compute the *best response* of network 1, holding the actions of all other networks fixed, and assuming an equilibrium is played in the second step[4]. Combining (11), (18), (19), (20) and (21), network 1's best response is a solution to the following optimization problem:

$$\begin{aligned}
\text{maximize} \quad & \eta_1 \equiv \beta_1^{\text{Net}} \left(\sum_i V_i \beta_i^{\text{Pub}} c_{i1} \right) (1 - h_1)\kappa_1 \\
\text{subject to} \quad & X_{ij} = V_i g_{ij} h_j \theta_j \quad \forall (i,j) \\
& \sum_j c_{ij} X_{ij} = \max_j X_{ij} \quad \forall i \\
& \sum_j c_{ij} = 1 \quad \forall i \\
& \theta_j = \kappa_j a_j \quad \forall j \\
& a_j = \beta_j^{\text{Net}} \frac{\left(\sum_i V_i c_{ij} \beta_i^{\text{Pub}}\right)}{\left(\sum_i V_i c_{ij} g_{ij}\right)} \quad \forall j \\
& 0 \le g_{i1}, h_1, c_{ij} \le 1 \quad \forall (i,j)
\end{aligned} \tag{23}$$

The objective in (23) is an expression for network 1's profit (see (22)). The first three constraints encode the assumption that each publisher chooses allocations optimally in the second step (see (20) and (21)). The fourth and fifth constraints say that advertisers also choose valuations optimally (see (18) and (19)) i.e., that there is an equilibrium in the second step between publishers and advertisers. In particular, observe that y_k, R_k and β_k^{Adv} do not appear in (23). The constants $\{\kappa_j \ \forall j\}$ are *sufficient statistics* for the distribution of advertiser types in our problem. The final constraint gives ranges for the decision variables we are interested in.

Network 1's optimization problem (23) is highly non-convex, so even feasible points are not easy to find. One of our main contributions is an iterative algorithm, which we call PricingPolicy, for finding near-optimal solutions to (23).

Define $\mathbf{g}_1 \equiv \{g_{i1}\}_{I\times 1}$ i.e., $\mathbf{g}_1$ is an I-by-1 matrix (i.e., a length-I vector) whose i^{th} element is g_{i1}. Similarly, let $\mathbf{C} \equiv \{c_{ij}\}_{I\times J}$ be an I-by-J matrix of publisher allocations and let $\mathbf{G}_{-1} \equiv \{g_{ij} \ \forall j \neq 1\}_{I\times(J-1)}$ and $\mathbf{h}_{-1} \equiv \{h_j \ \forall j \neq 1\}_{(J-1)\times 1}$ denote the actions of the other networks. Recall that in (23), $\mathbf{G}_{-1}$ and $\mathbf{h}_{-1}$ are given as inputs i.e., we are finding network 1's best response to $\mathbf{G}_{-1}$ and $\mathbf{h}_{-1}$, so they are not variables.

We refer to triples $(h_1, \mathbf{g}_1, \mathbf{C})$ as *points*. We say that the point $(h_1, \mathbf{g}_1, \mathbf{C})$ is *feasible* if it satisfies the constraints in (23). If $(h_1, \mathbf{g}_1, \mathbf{C})$ is feasible, it means that if network 1 plays $(h_1, \mathbf{g}_1)$ and the other networks play $(\mathbf{h}_{-1}, \mathbf{G}_{-1})$ in the first step, then the publishers' equilibrium allocations in the second step will be $\mathbf{C}$ (recall that the corresponding advertiser valuations can be computed from (17)). We say that $(h_1^*, \mathbf{g}_1^*, \mathbf{C}^*)$ is *optimal* if

[4]Our choice of network 1 is without loss of generality. Obviously we can compute the best response for any network j in a similar manner.

it is feasible and network 1's profit is (weakly) the highest when $(h_1^*, \mathbf{g}_1^*, \mathbf{C}^*)$ is played, compared to any other feasible point i.e., it is a best response to $(\mathbf{h}_{-1}, \mathbf{G}_{-1})$.

In Appendix A, we describe a sequence of transformations that yield a *geometric programming (GP) relaxation* of (23) around a given point $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right)$. That is, we approximate (23) by a GP in the vicinity of the point $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right)$. GPs are log-convex [2], and therefore can be solved globally and efficiently. PricingPolicy works by solving a sequence of these GPs. It outputs a sequence of feasible (but not-necessarily optimal) points, where each point yields weakly higher profits for network 1 than the previous point. The sequence of solutions (hopefully) converge to an approximate solution to (23).

Algorithm 1 PricingPolicy

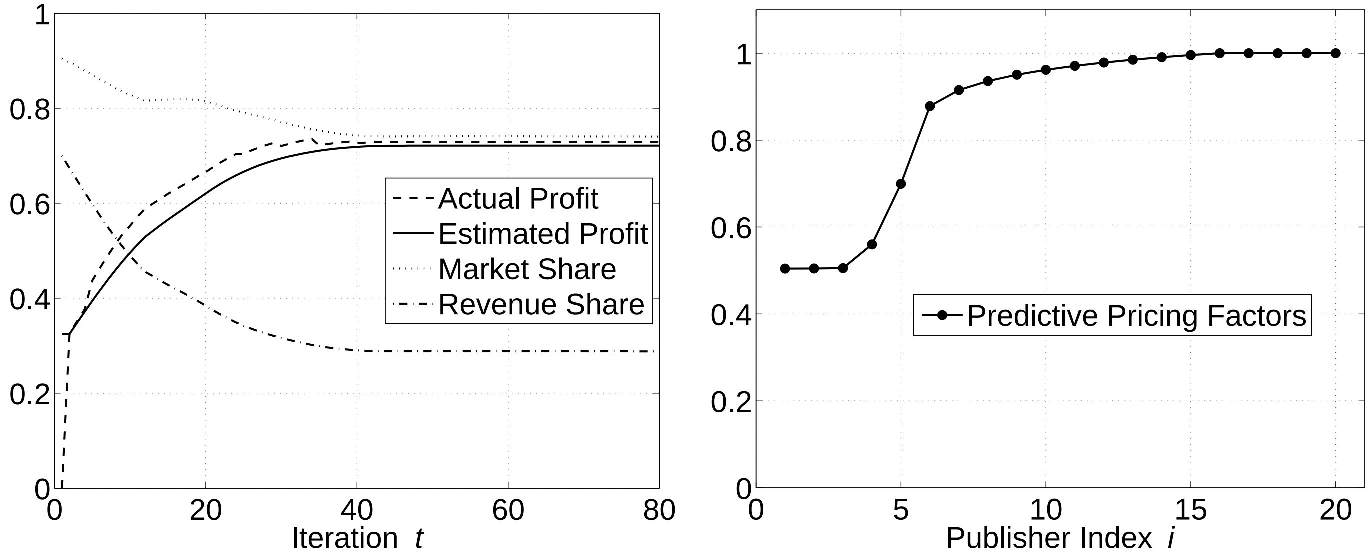
Require: $\mathbf{G}_{-1}$, $\mathbf{h}_{-1}$, T

1: Select arbitrary initializations $h_1^{(0)}$ and $\mathbf{g}_1^{(0)}$
2: Compute second-step equilibrium, $\mathbf{C}^{(0)}$, assuming other networks play $(\mathbf{h}_{-1}, \mathbf{G}_{-1})$ and network 1 plays $\left(h_1^{(0)}, \mathbf{g}_1^{(0)}\right)$
3: **for** $t \in 1, \ldots, T$ **do**
4: Solve GP-relaxation of (23) to find an optimal point $(h_1', \mathbf{g}_1', \mathbf{C}')$ that is "close to" $\left(h_1^{(t-1)}, \mathbf{g}_1^{(t-1)}, \mathbf{C}^{(t-1)}\right)$, assuming other networks play $(\mathbf{h}_{-1}, \mathbf{G}_{-1})$
5: $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right) \leftarrow (h_1', \mathbf{g}_1', \mathbf{C}')$
6: **end for**
7: Recompute second-step equilibrium, $\mathbf{C}^{(T)}$, assuming other networks play $(\mathbf{h}_{-1}, \mathbf{G}_{-1})$ and network 1 plays $\left(h_1^{(T)}, \mathbf{g}_1^{(T)}\right)$
8: **return** $\left(h_1^{(T)}, \mathbf{g}_1^{(T)}, \mathbf{C}^{(T)}\right)$

Given $\left(h_1^{(0)}, \mathbf{g}_1^{(0)}\right)$ and $(\mathbf{h}_{-1}, \mathbf{G}_{-1})$ (i.e., the first-step outcome), the second-step equilibrium allocations $\mathbf{C}^{(0)}$ in line 2 of PricingPolicy can be computed using fixed-point iteration on equations (19), (20) and (21). The purpose of line 2 is to provide a feasible starting point $\left(h_1^{(0)}, \mathbf{g}_1^{(0)}, \mathbf{C}^{(0)}\right)$ for the inner loop of PricingPolicy. Similarly, line 7 ensures that the final output $\left(h_1^{(T)}, \mathbf{g}_1^{(T)}, \mathbf{C}^{(T)}\right)$ is feasible for (23), since the solutions of the relaxed problem may be infeasible for the original problem (23).

Different initializations $\mathbf{g}_1^{(0)}$ and $h_1^{(0)}$ may lead to different local optima, so PricingPolicy should be executed several times with different initializations, keeping the best result. There are also a number of minor tweaks needed to ensure that PricingPolicy works well in practice. Some of these tweaks are related to "regularization", to ensure the numerical stability of the algorithm. It is also prudent to verify that the sequence of approximate solutions output by PricingPolicy corresponds to progress being made on the original problem. For brevity, we omit further details here.

In principle, PricingPolicy could be used as a subroutine to an compute equilibrium for the first-step i.e., for computing a subgame-perfect equilibrium for our one-shot dynamic game. We would run the algorithm (i.e., computing a best response) for each network j holding all other networks' actions fixed, and iterate until convergence. However, we feel that computing the best-response function is more useful in practice – what pricing policy should network j use in order to maximize its own profits?



(a) Progress of PricingPolicy.

(b) Optimal predictive prices, $g_{i1}^{(T)}$.

Figure 1: The effects of predictive pricing.

4 Experiments

Using PricingPolicy, we can gain some interesting insights into the structure of optimal pricing policies.

Our first experiment examines whether networks that apply predictive pricing gain a competitive edge, compared to networks that do not. Consider a market with $J = 2$ networks and $I = 20$ publishers. We assume $V_i = 100\ \forall i$ and $\beta_i^{\text{Pub}} = 0.0025i$. That is, each publisher receives 100 clicks, and β_i^{Pub} is linear in i with values ranging from 0.25% to 5%[5]. The networks are equally effective at matching up publishers and advertisers i.e., $\beta_1^{\text{Net}} = \beta_2^{\text{Net}} = 1.0$. We assume $\kappa_1 = \kappa_2 = 10$, so the auction mechanisms used by each network are also equally efficient.

We used PricingPolicy to compute the optimal pricing policy for network 1, assuming network 2 does not use predictive pricing (i.e., $g_{i2} = 1\ \forall i$) and offers publishers a revenue share of 50% (i.e., $h_2 = 0.5$). To solve the GP-relaxation of (23) in line 4 of PricingPolicy, we used `CVX`, a package for solving convex programs [7, 8].

We initialized the algorithm with random choices of $\mathbf{g}_1$ and h_1. Figure 1(a) shows the revenue share $h_1^{(t)}$ output at each iteration t, as well as the market share $\frac{1}{I}\sum_i c_{i1}^{(t)}$, estimated profit $\hat{\eta}_1^{(t)}$ and actual profit $\eta_1^{(t)}$ at each iteration[6]. The estimated profit is computed using the allocations $\mathbf{C}^{(t)}$ output by PricingPolicy in iteration t (recall that $\mathbf{C}^{(t)}$ may be infeasible for (23)), whereas the actual profit is computed using the actual second-stage equilibrium allocations resulting from $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}\right)$.

From Figure 1(a), we see that the algorithm converges after roughly $T = 50$ iterations. The estimated profit tracks the actual profit reasonably well – in this case it is an underestimate of the actual profit, but in other experiments we ran it was an overestimate. As iterations progress, $h_1^{(t)}$ steadily decreases – PricingPolicy recommends progressively better predictive prices $g_{i1}^{(t)}$, allowing network 1 to offer progressively lower revenue

[5]Such a range is realistic – 5% would be considered a high conversion rate in practice.

[6]From (22), note that $\eta_1 \le \left(\sum_i V_i \beta_i^{\text{Pub}}\right) \kappa_1 \beta_1^{\text{Net}} \equiv \eta_1^{\max}$, which is the maximum possible profit network 1 can attain in any outcome. Thus, in Figures 1(a) and 2, we normalize profits by $\eta_1^{\max}$.

shares.

Observe that the algorithm converges to a revenue share of 29%, which is much lower than the 50% being offered by network 2. Despite offering a lower revenue share, network 1 manages to attract 74% market share. Thus, the use of predictive pricing is giving network 1 a significant advantage.

It may seem surprising that the market share in Figure 1(a) is also falling across iterations. The lowest-quality (i.e., lowest β_i^{Pub}) publishers are essentially being driven from network 1 to network 2. Figure 1(b), which shows the final set of predictive prices $\mathbf{g}_1^{(T)}$, suggests why these publishers leave network 1. Advertisers are being charged very low prices (i.e., low g_{i1}) for traffic from low-quality publishers (i.e., low β_i^{Pub}). Consequently, network 1 offers to pay these low-quality publishers very little for their traffic, causing them to choose network 2 instead.

Observe that the optimal predictive prices in Figure 1(b) are increasing in i, and consequently in the conversion rate, β_i^{Pub}. That is, advertisers are being charged less for traffic from publishers whose conversion rate is lower. We ran several other experiments (not discussed here), and found the optimal g_{i1} was increasing in β_i^{Pub} in every case.

Essentially, a "lemons market" effect is avoided on network 1 as a result of predictive pricing. The lack of low-quality publishers on network 1 raises the average quality of network 1's traffic, causing advertisers' bids to increase. The high-quality publishers get paid more per click, and are willing to settle for a lower revenue share as a result.

Our second experiment considers the impact of targeting (i.e., β_j^{Net}) on market outcomes. In particular, if a network is more effective than its competitors at matching publishers with advertisers, does it translate to higher profits for that network? Consider a market with $J = 3$ networks and $I = 20$ publishers. We assume $\beta_i^{\text{Pub}} = 0.000125i^2$ i.e., β_i^{Pub} is quadratic in i, with values ranging from 0.0125% to 5% (there are many low-quality publishers and a few high-quality ones). Networks 2 and 3 are equally skilled at matching i.e., $\beta_2^{\text{Net}} = \beta_3^{\text{Net}} = 1.0$. We assume $\kappa_1 = \kappa_2 = \kappa_3 = 10$, so no network has an edge due to the auction mechanism they use. We assume that $g_{i2} = 20\beta_i^{\text{Pub}}$ (i.e., network 2 uses a predictive pricing rule that is linear in conversion rate), and network 3 sets $g_{i3} = 1\ \forall i$ (i.e., it does not use predictive pricing). Network 2 offers a lower revenue share than network 3, i.e., $h_2 = 0.5$ and $h_3 = 0.6$.

We computed optimal pricing policies for network 1, for various values of β_1^{Net} ranging from 0.7 to 1.3. Recall that β_1^{Net} greater than (less than) 1.0 means that network 1 is better (resp., worse) at matching than networks 2 and 3. Figure 2 shows network 1's optimal revenue share h_1^* and its resulting profits (normalized by $\eta_1^{\max}$). As we might expect, network 1 earns higher (lower) profits when β_1^{Net} is higher (resp., lower). From Figure 2, we see that network 1 is able to offer a lower revenue share when β_1^{Net} is higher, since network 1 is generating more conversions for advertisers, causing bids (and consequently publishers' revenues) to increase.

5 Conclusion

We have presented an economic model of the PPC advertising market that captures the effects of predictive pricing and revenue sharing. The model is simple, yet flexible enough to account not only for conversion rates, but also click-through rates and click fraud (although we did not discuss them in this paper). We derived an implicit expression for the optimal pricing policy for a network, as the solution to a difficult optimization problem. We then presented an iterative algorithm, PricingPolicy, which finds near-optimal solutions to

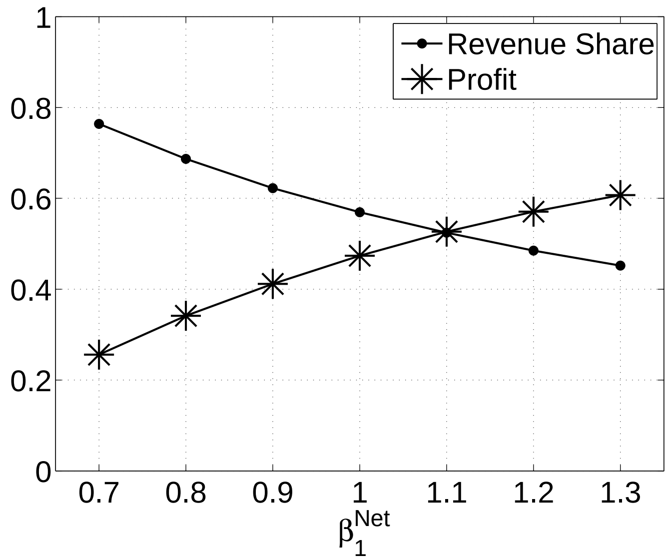


Figure 2: The effect of network 1's skill at matching publishers and advertisers (i.e., β_1^{Net}).

this problem.

Through experiments, we found that predictive pricing and revenue sharing can be very effective tools for advertising networks to attract publishers and advertisers, especially if their competitors are not using predictive pricing. It is not necessarily optimal to attract as much traffic as possible – quality can be just as important as quantity. Being more effective at matching publishers and advertisers can increase a network's profits, so improving their matching algorithms may be a worthwhile investment for networks.

References

[1] Aggarwal, G., Goel, A., and Motwani, R. Truthful auctions for pricing search keywords. In *ACM Conference on Electronic Commerce (EC06)* (June 2006).

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

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

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

[5] Edelman, B., Ostrovsky, M., and Schwarz, M. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review 97*, 1 (March 2007), 242–259.

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

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

[8] Grant, M., and Boyd, S. *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*. Springer, 2008, ch. Graph Implementations for Nonsmooth Convex Programs.

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

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

[11] Varian, H. Position auctions. *International Journal of Industrial Organization* (October 2006).

[12] Yahoo! A new pricing model rolls out today. *Yahoo! Search Marketing Blog* (June 2007).

A GP-relaxation of (23)

In this appendix, we show how to form a *geometric programming (GP) relaxation* of (23). GPs are log-convex [2], and can therefore be solved globally and efficiently. Instances of the relaxed problem are solved in each iteration of the `for` loop (lines 3 to 6) of PricingPolicy.

Begin by rewriting problem (23) as follows:

$$\begin{array}{ll} \text{maximize} & \left(\sum_i A_{i1}c_{i1}\right)(1-h_1) \\ \text{subject to} & u_{ij} = V_i g_{ij} h_j y_j \quad \forall (i,j) \\ & \sum_j c_{ij}u_{ij} = \max_j u_{ij} \quad \forall i \\ & \sum_j c_{ij} = 1 \quad \forall i \\ & y_j = \dfrac{(\sum_i A_{ij}c_{ij})}{(\sum_i V_i c_{ij} g_{ij})} \quad \forall j \\ & 0 \le g_{i1}, h_1, c_{ij} \le 1 \quad \forall (i,j) \end{array} \tag{24}$$

where we have defined $y_j \equiv \theta_j$, $u_{ij} \equiv X_{ij}$, and $A_{ij} \equiv V_i \beta_i^{\text{Pub}} \kappa_j \beta_j^{\text{Net}}$ for notational convenience. In (24), upper case quantities (i.e., A_{ij} and V_i) are constants, whereas lower case quantities are variables (except g_{ij} and h_j for $j \neq 1$, since $\mathbf{G}_{-1}$ and $\mathbf{h}_{-1}$ are given as input). The fourth and fifth constraints in (23) have been combined into one.

Replace max with softmax. Replace the second constraint in (24) by the following approximation:

$$c_{ij} = \frac{u_{ij}^{\alpha}}{\sum_n u_{in}^{\alpha}} \quad \forall (i,j) \tag{25}$$

where α is some large positive (but finite) constant. We are replacing the "hard" maximum in (24) with a "softmax" approximation i.e., if $u_{ij} > u_{in}\ \forall n \neq j$, then c_{ij} will be close to 1 (but not exactly 1). The softmax is continuous and differentiable whereas the maximum operator is not. Larger values of α yield better approximations, although choosing α too large may lead to numerical instability in practice.

Introduce linear equalities. Define $w_{ij} \equiv A_{ij}c_{ij}$ and $p_j \equiv \sum_i w_{ij}$. Define $z_{ij} \equiv V_i c_{ij} g_{ij}$ and $q_j \equiv \sum_i z_{ij}$. The fourth constraint in (24) then becomes $y_j = \frac{p_j}{q_j}\ \forall j$. Define $d_1 \equiv 1 - h_1$. The objective function in (24) can then be expressed as $d_1 p_1$.

We can then rewrite (24) as:

$$\begin{aligned} \text{maximize} \quad & d_1 p_1 \\ \text{subject to} \quad & d_1 = 1 - h_1 \\ & u_{ij} = V_i g_{ij} h_j y_j \quad \forall (i,j) \\ & c_{ij} = \frac{u_{ij}^{\alpha}}{\sum_m u_{im}^{\alpha}} \quad \forall (i,j) && (26) \\ & \sum_j c_{ij} = 1 \quad \forall i && (27) \\ & y_j = \frac{p_j}{q_j} \quad \forall j \\ & \sum_i w_{ij} = p_j \quad \forall j && (28) \\ & \sum_i z_{ij} = q_j \quad \forall j && (29) \\ & w_{ij} = A_{ij} c_{ij} \quad \forall (i,j) \\ & z_{ij} = V_i c_{ij} g_{ij} \quad \forall (i,j) \\ & 0 \leq g_{i1}, h_1, c_{ij} \leq 1 \quad \forall (i,j) && (30) \end{aligned}$$

The problem (30) is almost a GP, except for the linear equalities (27), (28) and (29), and the softmax equality (26), all of which are not GP-compatible[7]. Due to (27), the softmax equality can simply be replaced by a less-than inequality. To deal with the linear equalities, we replace them with *local monomial approximations* that are GP-compatible, as suggested by [2].

Aside: Local monomial approximations. Consider any non-negative length-L vector $\mathbf{x} \equiv \{x_i\}_{L\times 1}$ and a linear equality constraint $\sum_i x_i = 1$. The equality $\sum_i x_i = 1$ is equivalent to the following pair of inequalities:

$$\sum_i x_i \leq 1 \qquad (31)$$

$$\sum_i x_i \geq 1 \qquad (32)$$

[7] A constraint is said to be *GP-compatible* if it is permitted in a geometric program [2].

The less-than constraint (31) is GP-compatible, whereas the greater-than constraint (32) is not. To make (32) GP-compatible, fix a vector $\mathbf{x}_0$ and define $m : \mathbb{R}^L \to \mathbb{R}$ as follows:

$$m(\mathbf{x}; \mathbf{x}_0) \equiv d(\mathbf{x}_0) \prod_{i=1}^{L} x_i^{f_i(\mathbf{x}_0)} \tag{33}$$

where

$$f_i(\mathbf{x}) \equiv \frac{x_i}{\sum_{m=1}^{L} x_m}$$

and

$$d(\mathbf{x}) \equiv \left(\sum_{i=1}^{L} x_i \right) \prod_{i=1}^{L} x_i^{-f_i(\mathbf{x})}$$

The function $m(\mathbf{x}; \mathbf{x}_0)$ is a good monomial approximation of $\sum_i x_i$ around the point $\mathbf{x}_0$ in the sense that $\log m(\mathbf{x}; \mathbf{x}_0)$ is the first-order Taylor-series approximation of $\log \sum_i x_i$ about the point $\mathbf{x}_0$. Moreover,

$$m(\mathbf{x}; \mathbf{x}_0) \leq \sum_i x_i \ \forall \mathbf{x} \tag{34}$$

with equality iff $\mathbf{x} = \mathbf{x}_0$, which means m is a global under-approximation of $\sum_i x_i$.

Monomial greater-than inequalities are GP-compatible. Therefore, to approximate a a linear equality constraint $\sum_i x_i = 1$ in a GP, we replace the equality with the pair of inequalities:

$$\sum_i x_i \ \leq \ 1 + \epsilon \tag{35}$$

$$m(\mathbf{x}; \mathbf{x}_0) \ \geq \ 1 - \epsilon \tag{36}$$

The slack parameter $\epsilon > 0$ restricts our search space to a small ϵ-neighbourhood around $\mathbf{x}_0$, and ensures a non-singleton feasible set i.e., with $\epsilon = 0$, the only feasible solution to the inequalities (35) and (36) would be $\mathbf{x} = \mathbf{x}_0$.

Introduce monomial approximations. Given $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right)$, $(\mathbf{G}_{-1}, \mathbf{h}_{-1})$, $\epsilon > 0$ and $\alpha \gg 0$, define:

$$\mathbf{w}_j^{(t)} \equiv \{w_{ij}^{(t)}\}_{I\times 1} \qquad \text{where } \ w_{ij}^{(t)} \equiv A_{ij} c_{ij}^{(t)} \tag{37}$$

$$\mathbf{z}_j^{(t)} \equiv \{z_{ij}^{(t)}\}_{I\times 1} \qquad \text{where } \ z_{ij}^{(t)} \equiv \begin{cases} V_i c_{i1}^{(t)} g_{i1}^{(t)} & j = 1 \\ V_i c_{ij}^{(t)} g_{ij} & j \neq 1 \end{cases} \tag{38}$$

Also define $\mathbf{c}_i^{(t)} \equiv \{c_{ij}^{(t)}\}_{J\times 1}$. We can now replace each linear equality in (30) with a monomial approximation about the point $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right)$:

$$\begin{aligned}
\text{maximize} \quad & d_1 p_1 \\
\text{subject to} \quad & d_1 \leq 1 - h_1 \\
& u_{ij} = V_i g_{ij} h_j y_j \quad \forall (i,j) \\
& c_{ij} \leq (1+\epsilon) \frac{u_{ij}^{\alpha}}{\sum_m u_{im}^{\alpha}} \quad \forall (i,j) \\
& 1 - \epsilon \leq m\left(\mathbf{c}_i; \mathbf{c}_i^{(t)}\right) \quad \forall i \\
& \sum_j c_{ij} \leq 1 + \epsilon \quad \forall i \\
& y_j = \frac{p_j}{q_j} \quad \forall j \\
& (1-\epsilon) p_j \leq m\left(\mathbf{w}_j; \mathbf{w}_j^{(t)}\right) \quad \forall j \\
& \sum_i w_{ij} \leq (1+\epsilon) p_j \quad \forall j \\
& (1-\epsilon) q_j \leq m\left(\mathbf{z}_j; \mathbf{z}_j^{(t)}\right) \quad \forall j \\
& \sum_i z_{ij} \leq (1+\epsilon) q_j \quad \forall j \\
& w_{ij} = A_{ij} c_{ij} \quad \forall (i,j) \\
& z_{ij} = V_i c_{ij} g_{ij} \quad \forall (i,j) \\
& g_{i1}, h_1, c_{ij} \leq 1 \quad \forall (i,j)
\end{aligned} \tag{39}$$

Problem (39) is a relaxed version of (30), and it is a GP. We have eliminated all the non-negativity constraints from (30) since such constraints are implicit in any GP. The constraint $d_1 = 1 - h_1$ has been replaced by a less-than inequality – this constraint will be tight at the optimum since we are maximizing $d_1 p_1$. The slack parameter ϵ allows solutions of (39) to be infeasible for the original problem (24). That is why we recompute $\mathbf{C}^{(T)}$ in line 7 of PricingPolicy.

To summarize, in each iteration t of PricingPolicy, the point $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right)$ is given as input. We then search in the ϵ-vicinity of $\left(h_1^{(t)}, \mathbf{g}_1^{(t)}, \mathbf{C}^{(t)}\right)$ for a point that is more profitable for network 1. The optimum is then labeled $\left(h_1^{(t+1)}, \mathbf{g}_1^{(t+1)}, \mathbf{C}^{(t+1)}\right)$, and we iterate until convergence. Network 1's profit is monotonically increasing across iterations t, and is bounded above by $\eta_1^{\max}$, so convergence is guaranteed. Although there are far fewer variables in (24) than (39), the latter can be solved efficiently due to the log-convexity of GPs. Thus, the increased number of variables in (39) is acceptable.

B Notation

Table 1 is a summary of the notation used in our model.

Table 1: Summary of notation used in our model.

Symbol	Description
I, J, K	Number of publishers, networks and advertisers (respectively)
i, j, k	Index over publishers, networks and advertisers (respectively)
T	Number of iterations in `for` loop of PRICINGPOLICY
t	Index over iterations in `for` loop of PRICINGPOLICY
V_i	Volume of clicks on publisher i's site
β_i^{Pub}	Quality of publisher i's traffic
π_{ij}	Publisher i's revenue from clicks sent to network j
π_i	Publisher i's total revenue
c_{ij}	Fraction of publisher i's clicks sent to ad network j
y_k	Advertiser k's revenue per conversion
R_k	Advertiser k's target ROI
β_k^{Adv}	Effectiveness of advertiser k's ads
β_{ijk}	Conversion rate of clicks going from i to j to k
Y_{kj}	Advertiser k's revenue from network j's clicks
Z_{kj}	Number of clicks advertiser k is billed for by network j
R_{kj}	Advertiser k's ROI on ad network j
$\bar{v}_k$	Advertiser k's nominal valuation
v_{kj}	Advertiser k valuation of ad network j's clicks
β_j^{Net}	Network j's skill at matching publishers and advertisers
θ_j	Network j's expected auction revenue per click
κ_j	Network j's expected auction revenue per click when $a_j = 1$
a_j	Network j's adjustment factor
η_j	Network j's total revenue
$\eta_j^{\max}$	Network j's maximum possible revenue
h_j	Revenue share paid out by network j
g_{ij}	Predictive pricing factor applied to publisher i's traffic by network j