Optimizing Budget Constrained Spend in Search Advertising

Chinmay Karande* chinmayk@fb.com
Aranyak Mehta aranyak@google.com
Ramakrishnan Srikant srikant@google.com

Google Research
Mountain View, CA, USA

ABSTRACT

Search engine ad auctions typically have a significant fraction of advertisers who are budget constrained, i.e., if allowed to participate in every auction that they bid on, they would spend more than their budget. This yields an important problem: selecting the ad auctions in which these advertisers participate, in order to optimize different system objectives such as the return on investment for advertisers, and the quality of ads shown to users. We present a system and algorithms for optimizing such budget constrained spend. The system is designed be deployed in a large search engine, with hundreds of thousands of advertisers, millions of searches per hour, and with the query stream being only partially predictable. We have validated the system design by implementing it in the Google ads serving system and running experiments on live traffic. We have also compared our algorithm to previous work that casts this problem as a large linear programming problem limited to popular queries, and show that our algorithms yield substantially better results.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: Linear Programming; K.6.0 [**General**]: Economics

1. INTRODUCTION

Search ad auctions have emerged as the primary model for monetizing the value provided by search engines. Advertisers use phrases (keywords) to specify the set of queries they are interested in, and bid the cost they are prepared to pay per click on their ad. For each search query, the set of ads to show, the order in which they are shown, and the cost per click for each shown ad are determined via an auction.

Each advertiser also specifies a daily budget, which is an upper bound on the amount of money they are prepared to spend each day. While many advertisers use bids as the primary knob to control their spend and never hit their budget, there exists a significant fraction of advertisers who would spend more than their budget if they participated in every auction that their keywords match. Search engines often provide an option to automatically scale the advertiser's bids [1, 2], but a substantial fraction of budget constrained advertisers do not opt into these programs. For these advertisers, the search engine has to determine the subset of auctions the budget constrained advertiser should participate in. This creates a dependence between auctions on different queries, and leads to essentially a matching or an assignment problem of advertisers to auctions.

In this paper, we consider the problem of *optimized budget allocation*: allocating advertisers to queries such that budget constraints are satisfied, while simultaneously optimizing a specified objective. Prior work in this area has often chosen revenue as the objective to optimize. However, the long term revenue of a search engine depends on providing good value to users and to advertisers. If users see low quality ads, then this can result in ad-blindness and a drop in revenue. If advertisers see low return on investment (ROI), then they will reduce their bids and budgets, again resulting in a drop in revenue. Thus, we explore two other objectives in this paper: improving quality, and advertiser ROI.

Paper Outline We describe the problem of optimized budget allocation in Section 2, followed by related work in Section 3. We present our algorithms and system design in Section 4, and discuss certain key properties of the algorithm in Section 5. In Section 6, we show that our algorithms yield substantially better results than prior work and also describe the results of experiments on live traffic. We conclude with some closing thoughts in Section 7.

2. PROBLEM DEFINITION

Let A be the set of advertisers, and Q the set of queries. Each advertiser $a \in A$ comes with a daily budget B_a. Let $G(A, Q, E)$ be a bipartite graph such that for $a \in A$ and $q \in Q$, edge $(a, q) \in E$ means that an ad of a is eligible for the auction for query q (a's keywords match q). Let $\texttt{ctr}_{(a,q)}$ be the probability of a click on a's ad for q, and $\texttt{bid}_{(a,q)}$ be the amount a is willing to pay per click. (Note that $\texttt{ctr}_{(a,q)}$ is the position-independent CTR, i.e., the probability of a

*The author is currently at Facebook, Inc., Menlo Park, CA, USA. The work described in this paper was done while the author was at Google.

WSDM'13, February 4–8, 2013, Rome, Italy.

click at some chosen fixed position. In other words, $\texttt{ctr}_{(a,q)}$ does not depend on the position of the ad.)

When a query q arrives, the eligible ads a for q are ranked by $\texttt{bid}_{(a,q)}\texttt{ctr}_{(a,q)}$ and shown in that order. Denoting the jth ad in the order as a_j, the cost per click of a_j is set as

$$\texttt{cpc}(a_j) = \texttt{bid}_{(a_{j+1},q)}\texttt{ctr}_{(a_{j+1},q)}/\texttt{ctr}_{(a_j,q)}$$

This is known as the generalized second price (GSP) auction (see, e.g., [25, 14, 6]).

Let T_a denote the spend of the advertiser a if a participates in all the auctions for which a is eligible via the keyword match (ignoring a's budget). If $T_a > B_a$, the advertiser is budget constrained, and the search engine has to limit (or throttle) the set of auctions in which the advertiser participates.

Objectives. For budget constrained advertisers, the search engine can select to optimize different objectives such as:

- the quality of the ads shown, e.g., maximize the position-independent predicted click-through rate,
- reduce advertiser cost-per-click (maximize the number of clicks), or
- increase advertiser ROI.

We do not have full knowledge of the graph G, but only information from past data, i.e., the graphs from previous days. We also require practical algorithms which work online, i.e., given the next query, decide, in sub-second response time, which ad impressions to show. We can now define the problem as follows.

The Optimized Budget Allocation Problem: Given information about the past as $G'(A, Q', E')$ including the bids and CTR for each advertiser-query pair, the budget B_a, for each advertiser a, and a specified objective function to optimize: For each query arriving in an online stream of queries Q, decide which advertisers should participate in the auction.

3. RELATED WORK

There have been two broad approaches to optimizing budget constrained spend: allocation and bid modification. Allocation treats bids as fixed, and allows only decisions about whether the advertiser should participate in the auction. This is our setting, where we are constrained to not change bids, but only optimize allocations.

The second approach, bid modification, is in a setting where bids can be changed. This body of work typically considers the problem from the advertiser's perspective, and assumes full knowledge of the information that advertisers typically have (such as the value of clicks or conversions). However, this work can also be adapted to be applicable from a search engine's perspective, for advertisers who have opted in and allow the search engine to change their bids.

We next describe the related work in the allocation and bid modification approaches.

Allocation The paper by Abrams et al. [5] is the closest to our work. They solve the offline problem (with complete knowledge of future query frequency) for *head* queries (the most popular search queries) in the GSP auction setting using a linear program (LP), to optimize search engine revenue or advertiser clicks per dollar. Thus their approach yields an optimal solution to the click maximization problem in the general GSP setting. However, there are two reasons why this work is not the final word on the allocation approach. First, the LP can only run over head queries due to resource constraints, which brings up an interesting question: Can a non-optimal algorithm that runs over the entire query stream beat an optimal algorithm that is restricted to the head? Second, the LP formulation can yield solutions that are clearly unfair, in that the allocation for some advertisers is very different from what the advertiser would choose for themselves for that objective (see Section 5). Hence it is unclear whether LP solutions can be deployed by search engines.

A second stream of work focused on optimizing search engine revenue in a first price auction. Mehta et al. [22] and Buchbinder et al. [9] both provide an online approximation algorithm for optimizing revenue, with a best possible approximation guarantee of $1 - 1/e \simeq 63\%$, for the scenario in which we do not know anything about the future distribution of queries. In contrast, we assume that we can predict future distributions of queries, albeit noisily. Mahdian et al. [21] extended the algorithm from [22] to provide guarantees in the setting when we have unreliable estimates. [19, 13] analyzed revenue maximization in an average case setting where queries arrive in a random order, or are picked i.i.d from a distribution. All the above papers focus solely on search engine revenue, assume a first price auction, and do not consider multiple slots or positions. We focus on optimizing very different objectives, such as user experience and advertiser ROI, in the second price GSP auction, with multiple slots and positions.

There has been considerable related work in budget allocation for display ads, e.g., [11, 12, 15, 16, 26]. This work also does not consider second price auctions, or the objectives we study.

There has also been work on designing new incentive compatible auctions in the presence of bidders with budget constraints, initiated by [4, 8]. Our goal is to optimize budgets in the context of the GSP auction used by search engines, and hence we do not consider alternate auction designs.

Bid Modification The second approach [7, 10, 17, 20, 24] studies the advertiser's problem of bidding optimally in order to maximize ROI (or some notion of utility). Changing bids in such a manner is an alternate solution to dealing with budget constrained advertisers: if the advertiser permits, the search engine can scale bids down until the advertiser is no longer budget constrained. However, despite the availability of a free option to automatically scale bids [1], a substantial fraction of budget constrained advertisers have not opted in to use this product. For these advertisers, the search engine (as a policy decision) has to use the allocation approach, not bid modification. Hence the work on bid modification, while clearly an appealing alternative, is not applicable in our setting.

Despite the lack of applicability, it would be interesting to understand how optimized budget allocation fares against bid scaling. We compare our algorithms to bid scaling in Section 6.

4. ALGORITHMS

Given the problem of optimizing budget constrained spend, the first step is to neither over- nor under-spend.[1] The naive way to do this is to let advertisers participate in auctions until they hit their budget, and then make them ineligible for the rest of the day. Clearly, this will yield very biased traffic to the advertiser, and also skew auction competition towards the earlier part of the day. The next simplest approach, which does not have these drawbacks, is *Vanilla Probabilistic Throttling* (VPT).

For each advertiser a, define:

- B_a: the remaining budget for the day (or time period).
- T_a: the remaining maximum spend for the rest of the day, i.e., the total spend if the advertiser had unlimited budget.

We now define the **Vanilla Probabilistic Throttling** algorithm:

> For each arriving query q:
> For each budget constrained advertiser a:
> Flip a coin with $P[Heads] = B_a/T_a$.
> If heads, a participates in the auction.

Figure 1: Vanilla Probabilistic Throttling (VPT)

If our estimate of T_a is accurate, then each advertiser spends very close to her budget in expectation (and with high probability, by Chernoff bounds). Advertisers also receive a representative sample of the traffic they are eligible for.

4.1 Optimized Throttling

We now present algorithms for optimizing one or more of the following objectives:

- average quality of ads shown, represented by CTR.
- clicks per dollar,
- conversions per dollar,
- the advertiser's profit, using the difference between the bid and the cost per click as the estimate of profit.

For the chosen objective, given a candidate ad impression i (i.e., for a specific query and advertiser), we compute a metric $\theta(i)$ which tracks the desired objective. Given a choice between two impressions (for the same advertiser), we would prefer to show the impression with the higher value of the metric. For example, if the objective is quality, the metric could be the position-independent predicted click-through rate of the advertiser for this query, reflecting our desire to show higher-quality impressions. Conceptually, we would like to rank all the impressions for an advertiser by the desired metric, and choose the impressions with the highest metric score until the budget is filled.

[1]While most advertisers are clearly budget constrained or unconstrained, advertisers at the margin may switch back-and-forth between the two states, based on traffic. It is straightforward to handle these marginal cases. For ease of exposition, we ignore this issue in the paper; however our implementation does handle these cases gracefully.

To achieve this goal, our algorithm uses a third input, the rank $R_{\theta,a}$ of an impression for a given advertiser a and metric θ. Define

- $F_{\theta,a}(\mu)$: Estimated fraction of maximum spend T_a for which $\theta(i) \leq \mu$. In other words, F is the estimated cumulative distribution function of θ.
- $R_{\theta,a}(\mu) = 1 - F_{\theta,a}(\mu)$.

The lower the value of the rank R, the better the impression scores on our metric.

We now define the **Optimized Throttling** algorithm:

> For each arriving query q:
> For each budget constrained advertiser a:
> If $R_{\theta,a}(\theta(i)) \leq B_a/T_a$, then a participates in the auction.

Figure 2: Algorithm OT

While the algorithm appears to be a straightforward greedy algorithm, there are several subtleties:

- The algorithm yields a solution that is "fair" to each advertiser (formally defined, and proved in Section 5).
- The algorithm yields an optimal fair solution under certain constraints. While we can find many examples where the constraints don't hold, the constraints hold often enough that the algorithm is not too far from optimal in practice.
- We have transformed the domain from the space of queries to the distribution of some property of queries. Predicting the frequency of queries in the tail is intractable. Predicting the distribution of specific properties of the queries in the tail is very tractable (for properties we use in our algorithms).
- The choice of metric lets us optimize a wide range of objectives, or combinations of objectives (discussed next).

We now define five instantiations of OT, corresponding to the four objectives we listed earlier, and a fifth objective that combines quality and clicks:

	Objective	θ(i)
OT-CTR	Ad quality	$\texttt{ctr}_i$
OT-Clicks	Clicks	$1/\texttt{cpc}_i$
OT-Profit	Profit	$(\texttt{bid}_i - \texttt{cpc}_i)/\texttt{cpc}_i$
OT-Conversions	Conversions	$\texttt{cvr}_i\texttt{val}_i/\texttt{cpc}_i$
OT-CTR-Profit	Blend	$\frac{\texttt{ctr}_i(\texttt{bid}_i - \texttt{cpc}_i)}{\texttt{cpc}_i}$

Figure 3: Instantiations of OT

The first metric, $\texttt{ctr}_i$ is straightforward: we are using position-independent predicted CTR as a proxy for quality. (Of course, one could use any arbitrary quality metric instead of CTR.)

To understand the next three metrics, it is helpful to multiply the numerator and denominator by the position-dependent predicted CTR. For example, in OT-Clicks, the

numerator now becomes the expected number of clicks, and the denominator the expected cost, and the ratio is the clicks per dollar. Since the total spend is fixed for budget constrained advertisers, optimizing clicks is equivalent to optimizing clicks per dollar. [2]

For OT-PROFIT, assuming that $\texttt{bid}_i$ is the value to the advertiser, $\texttt{bid}_i - \texttt{cpc}_i$ is the expected profit to the advertiser if there is a click on this impression. So the metric for OT-PROFIT is the expected profit per dollar of spend.

For OT-CONVERSIONS the metric is the expected conversion value per dollar, given the value of a conversion on this impression $\texttt{val}_i$, and a model that predicts the conversion rate $\texttt{cvr}_i$. Building machine learning models for estimating $\texttt{cvr}_i$ is beyond the scope of the paper. However, we will note the existence of commercial systems that estimate $\texttt{cvr}_i$ given advertiser-specific conversion data, e.g., [2]. Advertisers do not necessarily need to provide conversion data in order to benefit from the techniques in the paper. One can build models for estimating conversion rate that are not advertiser-specific, e.g., we found that $\texttt{ctr}_i$ is correlated with $\texttt{cvr}_i$.

The final metric simply multiplies the metric for CTR and profit. The intuition is that for advertisers with a lot of variance on CTR but not much on profit, the algorithm will focus on CTR. Similarly for advertisers with more variance on profit than CTR, the algorithm will focus on profit. Thus if the search engine cares about both CTR and advertiser profit, the blended metric will likely yield better results than simply averaging the results of the individual metrics.

4.2 System design and implementation

We next describe our implementation of the algorithm in the production Google ads serving system. Our system has three primary components:

- estimating B_a/T_a,
- estimating and compressing $R_{\theta,a}$, and
- using the estimates at serving time.

We will use OT-CTR to illustrate the techniques used, and discuss any differences between OT-CTR and the other instantiations as they come up.

Estimating B_a/T_a: This component estimates the *impression probability* $\texttt{ip}$, where $\texttt{ip}$ is defined as B_a/T_a. In other words, $\texttt{ip}$ is the probability with which we should allow an impression of the advertiser to participate in an auction, in order to show her impressions uniformly through the remainder of the day, and exhaust her budget at the end of the day. We estimate $\texttt{ip}$ using traffic information from the past, and using the available budget. Given the inherent noise in traffic, a feedback loop is used at frequent intervals to adjust the probability. Clearly, the more accurate the estimate of $\texttt{ip}$, the more the gains from optimization.

Estimating and compressing $R_{\theta,a}$: We use historical data to compute the cumulative distribution function $F_{\theta,a}$. $R_{\theta,a}$ is a trivial transformation of $F_{\theta,a}$. In the rest of this section, we will drop the subscripts and refer to $R_{\theta,a}$ as R (purely for ease of exposition). Since a search engine has a large number of advertisers, we would like to compress the information in R at serving time. We compress R into a histogram H.

Recall that R is only used to answer the question of whether $R(\theta(i))$ is less than $\texttt{ip}$ $(= B_a/T_a)$. So if the estimate of $\texttt{ip}$ was very stable, we need just two buckets in the histogram H, with the boundary at the value c^* such that $R(c^*) = \texttt{ip}$. With two buckets, we would need just 8 bytes of data per budget constrained advertiser: the value c^* and the value of $R(c^*)$.

We may choose to create additional buckets around the threshold c^*, based on the tradeoff between increased gains from choosing the highest scoring impressions (see below) versus memory constraints. For each bucket m in H excluding the last bucket, we store the bucket boundary and the value of R at the bucket boundary. We refer to these histograms as *throttling parameters*.

Using the estimates at serving: When a query arrives, we need to determine, for each budget constrained advertiser a, whether a participates in the auction. The input consists of the histogram H, the current value of $\texttt{ip}$, and the value of $\theta(i)$ for the current impression i. Let $\theta(i)$ be in bucket m. Let $H_b(m)$ denote the upper boundary of bucket m, and let $H_r(m) = R(H_b(m))$. Then the advertiser participates in the auction with the following probability:

$$\begin{cases} 1, & \text{if } H_r(m) \leq \texttt{ip} \\ 0, & \text{if } H_r(m-1) \geq \texttt{ip} \\ \frac{\texttt{ip} - H_r(m-1)}{H_r(m) - H_r(m-1)}, & \text{otherwise} \end{cases}$$

The first two cases are straightforward, and follow directly from the goal that we (do not) show the impression if it is (not) in the top $\texttt{ip}$ fraction of spend. In the third case, the probability that the impression is in the top $\texttt{ip}$ fraction of spend is given by $(\texttt{ip} - H_r(m-1))/(H_r(m) - H_r(m-1))$, and hence we show the impression with that probability.

Implementation: We have built our data collection pipeline on top of Google's *sawzall* [23] infrastructure, which allows us to process historical query data in parallel. The throttling parameters generated by the data collection pipeline is written to the *Google File System (GFS)* [18]. The data is stored in the *protocol buffer* format [3], which reduces the storage requirement as well as make the transfer and processing of the data more efficient. From GFS, the throttling parameters data is picked up by the *ads data push* system, which writes it to one of its data channels. The ads serving system gets the updated throttling parameters through this channel.

Interactions between budget constrained advertisers: Out of the metrics in Figure 3, $\texttt{ctr}_i$, $\texttt{bid}_i$, $\texttt{cvr}_i$ and $\texttt{val}_i$ are all functions purely of the impression, and do not change based on the other ads in the auction. However, $\texttt{cpc}_i$ is a function of the runner-up, since we use a second-price auction. We analyzed the logs, and found that budget constrained ads are more likely to be next to budget unconstrained ads than budget unconstrained ads. However, we will have instances with consecutive budget constrained ads. We use an iterative technique for improving the performance of the online algorithms in these cases.

The intuition behind the iterative technique is to run sev-

[2]Some metrics like $\texttt{ctr}_i$ are purely a function of the impression. However, metrics like $\texttt{cpc}$ may change based on the other ads in the auction. We discuss this issue in Section 4.2.

eral (simulated) iterations of the auction. In each iteration, we compute the metric θ for each budget constrained advertiser (including those that do not participate in the auction in the current iteration), and based on the value of the metric, decide whether that advertiser participates in the next iteration. Note that an impression may be removed in one iteration and re-enter in a subsequent iteration. While we cannot prove convergence, in practice this often converges in a few rounds, or at least leads to an improved solution over simply using the value of θ from the first iteration.

5. KEY PROPERTIES OF ALGORITHM

The overall effectiveness of the algorithm depends on two factors: the accuracy of the prediction of future traffic (total traffic and the distribution of the metric), and the intrinsic effectiveness of the algorithm. To understand the latter, we analyze the algorithm on the offline version of the problem, in which the advertiser-query graph is known. We will focus on the instantiations with a linear objective: OT-Clicks, OT-Profit and OT-Conversions. For non-linear objectives such as CTR, optimality with even a single budget constrained advertiser may require allocation in a manner that is clearly against the advertiser's interest. So algorithms like OT-CTR which are designed to both be good for advertisers and improve quality cannot be optimal.

We first consider the special case of a single budget constrained advertiser, and show that our algorithm is optimal for linear objectives. We then discuss the issue of fairness when there are multiple budget constrained advertisers, and show by example that linear programming can yield solutions that are optimal but not fair. When we restrict the space of solutions to fair solutions, we show that our algorithm yields an optimal solution even with multiple budget constrained advertisers, as long as there are no adjacent budget constrained advertisers in a given auction. Obviously, we do get adjacent budget constrained advertisers in the real world – but the optimality result with that constraint suggests that the algorithm will perform well in practice.

For ease of exposition, we will choose the simplest instantiation, OT-Clicks as the representative algorithm in the proofs. It is straightforward to sketch out similar proofs for OT-Profit and OT-Conversions.

5.1 Optimal For A Single Budget Constrained Advertiser

We start by proving that given $G(A, Q, E)$ and a single budget constrained advertiser $a \in A$, OT-Clicks maximizes clicks per dollar for a. While the proof is obvious, it is useful as a building block to more interesting results.

Given an advertiser a, we define:

- E_a to be the set of impressions in queries where a is eligible to participate in the auction, and
- $I_a \subseteq E_a$ to be the set of impressions where a participates in the auction.

The total expected clicks is $\sum_{i \in I_a} \alpha_i \texttt{ctr}_i$, where α_i is the position normalizer. For fixed budget, maximizing clicks per dollar is the same as maximizing total clicks, which is captured by the following linear program: Given a, find

$$\underset{I_a \subseteq E_a}{\texttt{Max}} \sum_{i \in I_a} \alpha_i \texttt{ctr}_i \ , \quad s.t. \quad \sum_{i \in I_a} \texttt{spend}_i \leq B_a \qquad (1)$$

- Rank the impressions $i \in E_a$ in order of decreasing $1/\texttt{cpc}_i$.
- Pick the top impressions in E_a according to this ranking until the budget runs out, i.e. the largest prefix I_a, s.t. $\sum_{i \in I_a} \texttt{spend}_i \leq B_a$, and at most one additional fractional impression to finish the budget.

Figure 4: Offline-OT-Clicks-Single-Advertiser

Without fractional impressions, this is the integral knapsack problem. Since one click is a tiny fraction of advertiser spend, we allow the choice of one fractional impression, thereby converting the problem to a fractional knapsack problem. Observing that $\texttt{spend}_i = \alpha_i \texttt{ctr}_i \texttt{cpc}_i$ and therefore $\alpha_i \texttt{ctr}_i / \texttt{spend}_i = 1/\texttt{cpc}_i$, we get the algorithm in Figure 4, which is a simple greedy algorithm, using the ratio of the expected value from the click to the cost of the click. Theorem 1 follows from the well known optimality of the greedy strategy for the fractional knapsack problem, and its proof is omitted here.

Theorem 1. *Offline-OT-Clicks-Single-Advertiser computes an optimal solution to the ROI maximization problem for a single budget constrained advertiser.*

5.2 Fair Allocations

We begin with the following definition of an optimal allocation:

Definition 1. We call an allocation **optimal** if it maximizes

$$\frac{\sum_{a \in A} w_a \sum_{i \in I_a} \alpha_i \texttt{ctr}_i}{\sum_{a \in A} w_a}$$

over all possible allocations (given the $w_a \geq 0$, which are arbitrary advertiser specific weights).

However, this definition has the problem that in trying to maximize the weighted average of advertiser ROI, we may end up sacrificing the interests of some advertisers, as the following example illustrates.

Example 1. There are two budget constrained bidders, a and b, each with a budget of \$100. There are two different queries q_1 and q_2, each with 100 instances. q_1 has a minimum reserve cpc of \$1, and q_2 has a reserve of \$2 (one may replace the reserves by an unconstrained bidder, keeping the example unchanged). The bidders bid the following values for the queries (a does not bid on q_2):

	q_1	q_2
a	20	–
b	10	10
min	1	2

For ease of exposition, we assume that the CTR is equal for all advertisers and query pairs, in both positions. To maximize total clicks, or equivalently, clicks per dollar, the

optimal solution is to let a participate in q_1, and b in q_2. Then a gets 100 clicks and b gets 50, giving a total of 150 clicks at a cost-per-click of \$1.33. But this solution is not fair to b, who would rather show for q_1 and get a cpc of \$1 and hence 100 clicks instead of 50. In this scenario a would get only 10 clicks at a cpc of \$10, giving a total of 110 clicks at an average cpc of \$1.81. □

Example 1 motivates the following definition:

Definition 2. We define an allocation $I \subseteq E$ to be a **fair allocation** for the clicks objective, if $\forall\ a \in A$:

a. The expected spend of a is equal to B_a (a exhausts its budget), or $I_a = E_a$ (we show every impression of a it is eligible for).

b. Given the allocation of other advertisers (i.e. $I \backslash I_a$), I_a maximizes the total expected clicks that a can obtain within its budget.

We call an allocation I an **optimal fair allocation** if it is a fair allocation, and it maximizes

$$\frac{\sum_{a \in A} w_a \sum_{i \in I_a} \alpha_i \mathtt{ctr}_i}{\sum_{a \in A} w_a}$$

among all fair allocation (where the $w_a \geq 0$ are arbitrary advertiser specific weights).

We can similarly define fair allocations and optimal fair allocations for the profit objective and the conversions objective.

In Example 1, the allocation which maximizes the sum (giving 150 clicks) is not a fair allocation, while allocating both a and b to q_1 is (even though this reduces the total clicks to 110). We note that our definition of fair allocation is analogous to that of a *Nash equilibrium* in games.

5.3 Optimal Fair Allocation When No Adjacent Budget Constrained Advertisers

When there are multiple budget constrained advertisers, a natural local algorithm is to cycle over the different advertisers until convergence, running Algorithm Offline-OT-Clicks-Single-Advertiser for each advertiser a. In the remainder of this section we analyze this algorithm, which is defined in Figure 5.

In a GSP auction, there are two ways in which the introduction or removal of an impression i of one budget constrained advertiser can effect an impression i' of another:

- i can change the cpc of i'
- i can change the position of i', and hence the expected spend from i'.

Due to this interaction, it is unclear whether Algorithm Offline-OT-Clicks always yields an optimal fair allocation. However we will show that it does converge to a optimal fair allocation when there are no adjacent budget constrained bidders, i.e., in the auction ranking of all eligible bidders, there are no consecutive budget constrained bidders. The key property is that in such a scenario, the cpc of a budget constrained impression is independent of other budget constrained impressions (even though the position may change due to the insertion or removal of other budget constrained impressions). As an aside, this key property also holds for first price auctions, and the following results also hold for first price auctions.

1. Begin with an allocation with only budget unconstrained advertisers.
2. For each budget constrained advertiser $a \in A$ (in turn):
   –Run Offline-OT-Clicks-Single-Advertiser for a.
   –Update I.
3. If the allocation has not converged, go to step 2.

Figure 5: Algorithm Offline-OT-Clicks.

Given this property, for each advertiser a, the 1 / cpc ordering of the impressions in E_a is fixed throughout the algorithm, independent of the allocations of the other advertisers. From the definition, it is easy to see that every fair allocation has the property that for each $a \in A$, I_a is a prefix of this fixed ordering of E_a, since a non-prefix would violate Property (b) in Definition 2. By definition of the algorithm, this is true also for the allocations produced during every step of the algorithm. We call allocations with this property *prefix-allocations*.

Lemma 2. *Algorithm Offline-OT-Clicks converges to a fair allocation if there are no adjacent constrained bidders in any query.*

Proof. Fix an advertiser a, and consider the allocation to a in each round. We claim that the prefix chosen for a only increases in length in subsequent rounds. Since for each advertiser the prefix cannot increase indefinitely, this means that the algorithm converges to some allocation, say I^*. Property (a) in Definition 2 holds for I^* because of the termination condition of Offline-OT-Clicks. Property (b) holds by definition of Offline-OT-Clicks-Single-Advertiser, which is run in every round of Offline-OT-Clicks.

It remains to prove the claim that the prefix for a can only increase in each round. Suppose this is true up to the time we process advertiser a in round k. In between the times we process a in rounds k and $k+1$, the algorithm may have introduced impressions of other advertisers in the auctions for which we show a's impressions in round k. The only effect this can have on a's impression is to possibly lower its position, and therefore of its expected spend. Thus, in round $k+1$, the algorithm may need to pick a longer prefix to finish a's budget. □

For two prefix-allocations I, J, we say $I \prec J$ if for every advertiser $a \in A$, the prefix length in I is at most the prefix length in J, and therefore $I_a \subseteq J_a$.

Lemma 3. *Let I^* be the fair allocation that Algorithm Offline-OT-Clicks converges to (when there are no adjacent constrained bidders). Then*

$$I^* \prec I, \text{ for all fair allocations } I$$

Proof. If this is not true for some fair allocation I, then consider the *first time* during the run of the algorithm that

some advertiser a's prefix becomes longer than its prefix in I. Comparing to I, the algorithm's current allocation has all advertisers $a' \neq a$ with smaller or equal prefixes. Thus the position normalizers of a's impressions are larger or equal during this step of the algorithm than in the allocation I. This implies that the prefix of a in the current allocation should be shorter or equal than that in I, since the expected spend in each auction is at least that in I, a contradiction. □

Note that Lemma 3 implies that there is a unique $\prec$-minimal fair allocation, and that the algorithm converges to it.

THEOREM 4. *Algorithm Offline-*OT-CLICKS *converges to an optimal fair allocation if there are no adjacent constrained bidders in any query.*

PROOF. From Lemma 2 we know that the algorithm converges to a fair allocation I^*. From Lemma 3 we get that for every advertiser a, and every fair allocation I, a's prefix in I^* is no longer than its prefix in I. Thus a spends the same amount of money (in expectation) in I^* as in I, but spends it on a subset $(I^*)_a \subseteq I_a$ such that impressions in $I_a \backslash (I^*)_a$ have lower (or equal) ratios of value of click to cost of click, as impressions in $(I^*)_a$. This implies that a gets at least as much total expected value in I^* as in I, in turn implying that I^* maximizes any weighted average (over advertisers) of the total expected value obtained, among all fair allocations. □

As mentioned earlier, it is easy to prove similar statements about other linear objectives such as optimizing profit and optimizing conversions.

6. EMPIRICAL EVALUATION

We report two types of experiments below, offline simulations to compare different optimized allocation solutions, and live experiments on Google traffic. We start with a description of prior approaches: LP and BidScaling.

6.1 LP and BidScaling

6.1.1 Linear Programming

Assuming complete knowledge of the data, a theoretical benchmark for any budget allocation algorithm can be obtained via linear programming [5]. For any query q, let s denote the set of bidders chosen to participate in the auction by an allocation mechanism. Let $\mathcal{C}(q)$ be the collection of all such sets, generated by *any* conceivable algorithm. Clearly, we can completely specify an allocation policy by specifying for each query q, the set $s \in \mathcal{C}(q)$ of bidders permitted by the algorithm to participate in the auction for q.

We can then attempt to discover the best allocation policy (for a given linear objective, say click maximization) using linear programming. Let N_q be the number of times query q appears in the data and x_{qs} be the number of times the set s of bidders is selected for query q. For a bidder $i \in s$, let cpc_{qsi} and ctr_{qsi} be the cost-per-click and click-through rate for i in the auction for q. Let α_{si} be the position normalizer for impression i. Let B_i be the budget of bidder i. Then, we can discover the best allocation policy that maximizes total clicks provided to budget constrained advertisers as the solution of the following linear program:

$$\max \sum_{q,s,i} \alpha_{si} \text{ctr}_{qsi} x_{qs}$$

$$\text{Allocation constraint:} \quad \sum_{s} x_{qs} \leq N_q \quad \forall q$$

$$\text{Budget constraint:} \quad \sum_{q,s} \text{cpc}_{qsi} \alpha_{si} \text{ctr}_{qsi} x_{qs} \leq B_i \quad \forall i$$

One can optimize for other linear metrics, such as revenue by suitably changing the objective function. It is not possible to directly optimize for non-linear metrics such as CTR.

The advantage of LP (compared to our approach) is that the LP fully incorporates interactions between different budget constrained advertisers. However, even the most efficient LP solvers cannot solve linear programs on the volume of data a search engine sees in a day. Thus we have to limit LP to the head portion of query traffic, and use an approach such as Vanilla Probabilistic Throttling on the long tail.

6.1.2 Bid Scaling

Both the OT algorithms and the LP formulation work under the constraint that the bidder's inputs (bids) can not be changed by the search engine, but they can only be throttled. Bid Scaling algorithms go outside this design space by lowering the bids of the constrained bidders appropriately. As we discussed in Section 1, bid scaling is not an option for our problem. Nevertheless, it is interesting to compare our approach against bid scaling.

Our bid scaling algorithm finds one bid multiplier per bidder and applies it to all the advertiser's bids, similar to the Budget Optimizer product in Google Adwords [1]. The multiplier is calculated so as to spend exactly the bidder's budget.

6.2 Simulation Methodology

We conducted simulations using a 20% sample of all US queries made over a week to the Google search engine. We sorted queries by the total impressions for that query (summed over all query instances), and picked the queries with the most impressions as the "head" queries for the LP. The number of queries in the head was chosen so that the LP could run in memory. (Note that as we move from the head to the tail, each query has relatively few instances. So even doubling the memory will not substantially increase the fraction of revenue or clicks covered by the "head" queries.) For each set (head and tail), we computed an appropriate budget for that set by scaling down the total budgets. For each query we also have the candidate ads together with the relevant metrics, namely, the bid and the predicted CTR. Our simulation is offline, i.e., the set of queries and candidates is fixed. Since all the algorithms we simulate are time independent (as opposed to some of the bid scaling algorithms studied earlier, e.g., [22, 9, 13, 16, 15]), we do not need to worry about the time-arrival order of the queries.

We simulated the following throttling algorithms for our first set of comparisons:

1. VPT: Vanilla Probabilistic Throttling.
2. OT-CLICKS: Optimized Throttling, objective is clicks (or inverse of cpc).

3. OT-CTR: Optimized Throttling, objective is CTR.
4. BidScaling, as described in Section 6.1.2
5. LP-Clicks: Click maximizing Linear Program on head queries.

6.3 Results

We show the changes in various objectives relative to the baseline of Vanilla Probabilistic Throttling (VPT). It is important to note that while we expect the overall conclusions to carry over to an online setting where the query distribution changes over time, the exact numbers will change. In general, the gains from optimized budget allocation or bid scaling will be significantly lower in live experiments due to changes in query traffic. For this reason, as well as data confidentiality, we omit the scale from our graphs below.

6.3.1 Comparison with LP

Figure 6 shows the change in clicks per dollar for budget constrained advertisers for each of the algorithms. The first set of numbers, "head", show the results when we artificially restrict all the algorithms to operate over the same set of head queries as LP-Clicks, with VPT on the tail. Since LP-Clicks is not just optimal, but can also generate solutions that are not fair (unlike the other algorithms), it is not surprising that LP-Clicks outperforms the alternatives.

However, when we allow the algorithms to optimize over the entire dataset – the "all" numbers – the algorithms that can use the full data dramatically outperform LP-Clicks. In fact, even OT-CTR, which is optimizing CTR and not CPC, yields a higher drop in CPC (or equivalently, more clicks per dollar) than LP-Clicks. The reason for the poor performance of LP-Clicks is that the LP can be run only on the head, and even though the head queries account for a substantial portion of revenue, they are relatively homogeneous – the potential gains from optimization are more in the tail than the head. We found that this held for the other metrics as well, i.e., the substantial majority of the gains from optimization came from the tail queries.[3]

6.3.2 Comparison with BidScaling

The other interesting comparison in Figure 6 is between OT-Clicks and BidScaling. BidScaling performs slightly better than OT-Clicks when restricted to head queries, as many advertisers may appear for a relatively small number of queries in the head. Thus OT-Clicks, which doesn't have the flexibility to scale bids, has a bit less room to maneuver. Over all queries, OT-Clicks has much more scope to differentiate between queries, and hence does slightly better than BidScaling.

However, OT-Clicks may be getting the gains by dropping high bid, high cpc clicks which might still yield more profit for the advertiser than low bid, low cpc clicks. Figure 7 shows how the algorithms do on estimated profit-per-dollar: the sum of the bids minus the total cost, divided by the total cost, over all budget constrained campaigns. (From this point, all figures represent performance using all queries, not just the head.) Here OT-Clicks does much worse than BidScaling, though it is still positive. In fact, even OT-CTR beats OT-Clicks. OT-Profit, which attempts to optimize profit, yields similar results to BidScaling.

[3]An implementation of LP that used more resources could narrow the gap by increasing the fraction of queries covered by the "head". However, as the number of distinct queries increases rapidly for each fraction of additional coverage, every doubling of resources will only yield incremental gains.

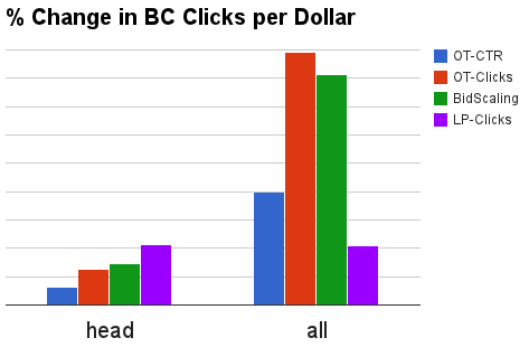

Figure 6: Impact on clicks-per-dollar, over budget constrained campaigns. The baseline is VPT.

As discussed in the introduction, user experience (quality of ads) is as important as advertiser ROI for long-term success. At first glance, one might expect that optimizing clicks per dollar would yield similar results to optimizing CTR: doesn't a higher click-through rate mean more clicks? However, what matters for optimizing clicks per dollar (with a fixed budget) is the cost per click, not the click-through rate.

Figure 8 shows the change in CTR for each of the algorithms. OT-CTR dramatically outperforms all other algorithms, not surprising since it is the only algorithm explicitly trying to optimize quality. Interestingly, while both OT-Profit and BidScaling gave similar gains in profit-per-dollar, their effect on quality is quite different: OT-Profit improves CTR, while BidScaling reduces CTR.

6.3.3 Multiple Objectives

We now present results with metrics that blend the CTR and profit objectives. In Section 4 we had conjectured that blended metrics might yield better results than individual metrics, since different advertisers may have better scope for optimization along different dimensions. Figures 9 and 10 show the impact of two blended metrics: $\texttt{ctr}(\texttt{bid}_i - \texttt{cpc}_i)/\texttt{cpc}$, and $\texttt{ctr}^2(\texttt{bid}_i - \texttt{cpc}_i)/\texttt{cpc}$. Notice that OT-CTR-Profit, which uses the former as the metric, almost matches OT-Profit on profit-per-dollar, while yielding significantly higher gains in CTR than OT-Profit. OT-CTR2-Profit further increases CTR gains, for a bit more drop in profit-per-dollar. In addition to validating our conjecture that blended metrics may yield better results, such blended metrics let the search engine pick any arbitrary point in a curve that trades gains in user quality for gains in advertiser value.

6.3.4 Summary

For optimizing clicks-per-dollar, OT-Clicks dramatically outperformed LP-Clicks by using all the data. For optimizing profit-per-dollar, OT-Profit matched BidScal-

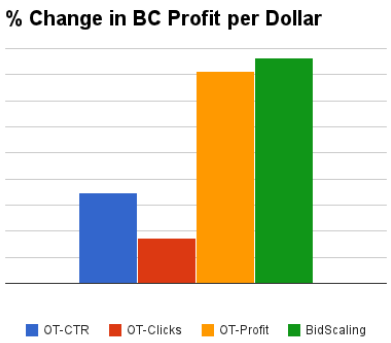

Figure 7: Impact on profit-per-dollar, over budget constrained campaigns. The baseline is VPT.

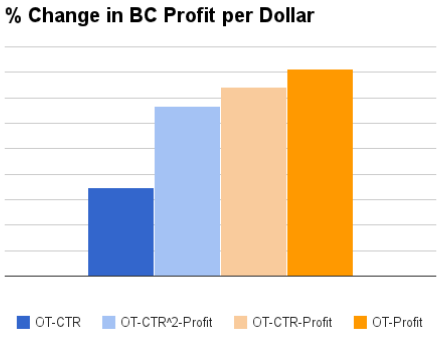

Figure 9: Multiple objectives: impact on Profit-per-dollar. The baseline is VPT.

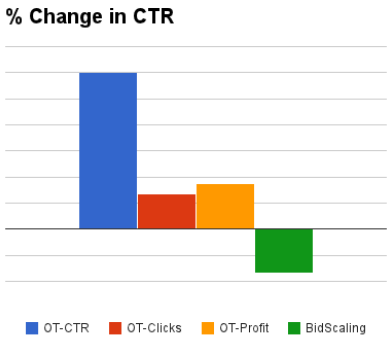

Figure 8: Impact on CTR (including all campaigns). The baseline is VPT.

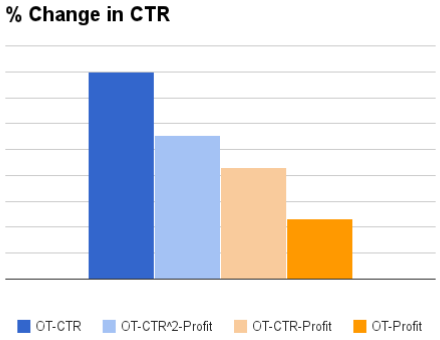

Figure 10: Multiple objectives: impact on CTR. The baseline is VPT.

ING on profit-per-dollar, while yielding better CTR (quality for users). If the primary goal was quality, OT-CTR blew away the other algorithms, while still improving advertiser profit-per-dollar and clicks-per-dollar. Finally, blending multiple metrics can yield better results than a single metric, since different advertisers have more scope for optimization along different metrics.

One might wonder whether implementation of such techniques would incentivize budget unconstrained advertisers to lower their budgets and become budget constrained. The answer is negative: BIDSCALING did slightly better than OT-PROFIT from an advertiser's perspective (ignoring quality for users), and BIDSCALING only used the information available to the search engine, not the additional information available to the advertiser. With additional information about conversion rates for each keyword, or the true value of each click (rather than using the bid as the proxy for value), the advertiser easily get better ROI by optimizing their campaign (versus becoming budget constrained).

6.4 Live Traffic Experiments

We implemented our algorithms in Google's production ad serving system, and ran experiments on live traffic, with both OT-CTR and BIDSCALING. The results were consistent with our simulations, though the magnitude of the gains was significantly less than in the simulations, since we have complete knowledge of future queries in the simulations, unlike the partial predictability of live traffic.

For OT-CTR, the experiments showed statistically significant improvements in quality for users, clicks, and conversions, while revenue was neutral – a Pareto improvement to all objectives. Gains in conversions per dollar were significantly higher than the gains in clicks per dollar, since CTR is correlated with conversion rate. Thus by shifting spend to ads with higher CTR, we also increased the number of conversions.

7. CONCLUSION

We studied the problem of allocating budget constrained spend in order to maximize objectives such as quality for users, or ROI for advertisers. We introduced the concept of fair allocations (analogous to Nash equilibriums), and constrained the space of algorithms to those that yielded fair allocations. We were also constrained (in our setting) to not modify bids. We proposed a family of *Optimized Throttling* algorithms that work within these constraints, and can be used to optimize different objectives. In fact, they can be tuned to pick an arbitrary point in the tradeoff curve between multiple objectives.

Prior approaches such linear programming and bid scaling

are not applicable in our setting: linear programming yields unfair allocations, and bid scaling changes bids. It was nevertheless interesting to study how much of a penalty (if any) our algorithms pay for working within these constraints (fair allocations, fixed bids). We found that, surprisingly, our algorithms dramatically outperform linear programming – by being fast enough to use all the data rather than being limited to head queries. Our algorithms are also competitive with bid scaling on advertiser metrics, while yielding better ad quality for users.

The Optimized Throttling algorithms are designed for implementation in a high throughput production system. The computation overhead at serving time is negligible: just a few comparisons. The algorithms also have a minimal memory footprint, as little as 8 bytes (plus hash table overhead) per advertiser. Finally, they are robust with respect to errors in estimating future traffic, since they only need the total volume of traffic and the distribution of the chosen metric, not the number of occurrences of each query. We validated our system design by implementing our algorithms in the Google ads serving system, and running experiments on live traffic. The experiments showed significant improvements in both advertiser ROI (conversions per dollar) and user experience.

Acknowledgments: We thank Anshul Kothari for his contributions to the algorithms and system design.

8. REFERENCES

[1] Google automatic bidding product. `http://adwords.google.com/support/aw/bin/answer.py?hl=en&answer=113234`.

[2] Google conversion optimizer product. `http://www.google.com/adwords/conversionoptimizer/`.

[3] Protocol buffers. Website, 2008. `http://code.google.com/p/protobuf`.

[4] Z. Abrams. Revenue maximization when bidders have budgets. In *SODA*, 2006.

[5] Z. Abrams, S. Keerthi, O. Mendelevitch, and J. Tomlin. Ad delivery with budgeted advertisers: a comprehensive lp approach. *J. Electronic Commerce Research*, 9(1), 2008.

[6] G. Aggarwal, A. Goel, and R. Motwani. Truthful auctions for pricing search keywords. In *EC*, 2006.

[7] C. Borgs, J. Chayes, N. Immorlica, K. Jain, O. Etesami, and M. Mahdian. Dynamics of bid optimization in online advertisement auctions. In *Proc. of the 16th international conference on World Wide Web*, pages 531–540. ACM, 2007.

[8] C. Borgs, J. Chayes, N. Immorlica, M. Mahdian, and A. Saberi. Multi-unit auctions with budget-constrained bidders. In *EC*, pages 44–51, 2005.

[9] N. Buchbinder, K. Jain, and J. Naor. Online Primal-Dual Algorithms for Maximizing Ad-Auctions Revenue. In *ESA*, 2007.

[10] M. Cary, A. Das, B. Edelman, I. Giotis, K. Heimerl, A. Karlin, C. Mathieu, and M. Schwarz. Greedy bidding strategies for keyword auctions. In *Proc. of the 8th ACM conference on Electronic commerce*, pages 262–271. ACM New York, NY, USA, 2007.

[11] D. X. Charles, M. Chickering, N. R. Devanur, K. Jain, and M. Sanghi. Fast algorithms for finding matchings in lopsided bipartite graphs with applications to display ads. In *ACM Conference on Electronic Commerce*, pages 121–128, 2010.

[12] Y. Chen, P. Berkhin, B. Anderson, and N. Devanur. Real-time bidding algorithms for performance-based display ad allocation. In *KDD*, pages 1307–1315. ACM, 2011.

[13] N. R. Devanur and T. P. Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *ACM Conference on Electronic Commerce*, pages 71–78, 2009.

[14] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet Advertising and the Generalized Second-Price Auction. *American Economic Review*, 97(1):242–259, 2007.

[15] J. Feldman, M. Henzinger, N. Korula, V. S. Mirrokni, and C. Stein. Online stochastic packing applied to display ad allocation. In *ESA (1)*, pages 182–194, 2010.

[16] J. Feldman, N. Korula, V. S. Mirrokni, S. Muthukrishnan, and M. Pál. Online ad assignment with free disposal. In *WINE*, pages 374–385, 2009.

[17] J. Feldman, S. Muthukrishnan, M. Pal, and C. Stein. Budget optimization in search-based advertising auctions. In *EC*, 2007.

[18] S. Ghemawat, H. Gobioff, and S.-T. Leung. The Google File System. In *19th ACM Symposium on Operating Systems Principles*, 2003.

[19] G. Goel and A. Mehta. Online budgeted matching in random input models with applications to Adwords. In *SODA*, 2008.

[20] K. Hosanagar and V. Cherepanov. Optimal bidding in stochastic budget constrained slot auctions. In *EC*, 2008.

[21] M. Mahdian, H. Nazerzadeh, and A. Saberi. Allocating online advertisement space with unreliable estimates. In *EC*, 2007.

[22] A. Mehta, A. Saberi, U. V. Vazirani, and V. V. Vazirani. Adwords and generalized online matching. *J. ACM*, 54(5), 2007.

[23] R. Pike, S. Dorward, R. Griesemer, and S. Quinlan. Interpreting the data: Parallel analysis with sawzall. *Scientific Programming Journal*, 13:277–298, 2005.

[24] P. Rusmevichientong and D. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In *EC*, 2006.

[25] H. Varian. Position auctions. *International Journal of Industrial Organization*, 25(6):1163–1178, 2007.

[26] E. Vee, S. Vassilvitskii, and J. Shanmugasundaram. Optimal online assignment with forecasts. In *ACM Conference on Electronic Commerce*, pages 109–118, 2010.