Budget Optimization in Search-Based Advertising Auctions

Jon Feldman
Google, Inc.
New York, NY
jonfeld@google.com

S. Muthukrishnan
Google, Inc.
New York, NY
muthu@google.com

Martin Pál
Google, Inc.
New York, NY
mpal@google.com

Cliff Stein*
Department of IEOR
Columbia University
cliff@ieor.columbia.edu

ABSTRACT

Internet search companies sell advertisement slots based on users' search queries via an auction. While there has been previous work on the auction process and its game-theoretic aspects, most of it focuses on the Internet company. In this work, we focus on the advertisers, who must solve a complex optimization problem to decide how to place bids on keywords to maximize their return (the number of user clicks on their ads) for a given budget. We model the entire process and study this budget optimization problem. While most variants are NP-hard, we show, perhaps surprisingly, that simply randomizing between two uniform strategies that bid equally on all the keywords works well. More precisely, this strategy gets at least a $1 - 1/e$ fraction of the maximum clicks possible. As our preliminary experiments show, such uniform strategies are likely to be practical. We also present inapproximability results, and optimal algorithms for variants of the budget optimization problem.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity; J.4 [**Computer Applications**]: Social and Behavioral Sciences —Economics

General Terms

Algorithms, Economics, Theory.

Keywords

Sponsored Search, Optimization, Auctions, Bidding.

1. INTRODUCTION

Online search is now ubiquitous and Internet search companies such as Google, Yahoo! and MSN let companies and individuals advertise based on search queries posed by users. Conventional media outlets, such as TV stations or newspapers, price their ad slots individually, and the advertisers buy the ones they can afford. In contrast, Internet search companies find it difficult to set a price explicitly for the advertisements they place in response to user queries. This difficulty arises because supply (and demand) varies widely and unpredictably across the user queries, and they must price slots for billions of such queries in real time. Thus, they rely on the market to determine suitable prices by using auctions amongst the advertisers. It is a challenging problem to set up the auction in order to effect a stable market in which all the parties (the advertisers, users as well as the Internet search company) are adequately satisfied. Recently there has been systematic study of the issues involved in the game theory of the auctions [5, 1, 2], revenue maximization [10], etc.

The perspective in this paper is not of the Internet search company that displays the advertisements, but rather of the advertisers. The challenge from an advertiser's point of view is to understand and interact with the auction mechanism. The advertiser determines a set of keywords of their interest and then must create ads, set the bids for each keyword, and provide a total (often daily) budget. When a user poses a search query, the Internet search company determines the advertisers whose keywords match the query and who still have budget left over, runs an auction amongst them, and presents the set of ads corresponding to the advertisers who "win" the auction. The advertiser whose ad appears pays the Internet search company if the user clicks on the ad.

The focus in this paper is on how the advertisers bid. For the particular choice of keywords of their interest[1], an advertiser wants to optimize the overall effect of the advertising campaign. While the effect of an ad campaign in any medium is a complicated phenomenon to quantify, one commonly accepted (and easily quantified) notion in search-based advertising on the Internet is to maximize the number of clicks. The Internet search companies are supportive to-

*Work done while visiting Google, Inc., New York, NY.

EC'07, June 11–15, 2007, San Diego, California, USA.

[1]The choice of keywords is related to the domain-knowledge of the advertiser, user behavior and strategic considerations. Internet search companies provide the advertisers with summaries of the query traffic which is useful for them to optimize their keyword choices interactively. We do not directly address the choice of keywords in this paper, which is addressed elsewhere [13].

wards advertisers and provide statistics about the history of click volumes and prediction about the future performance of various keywords. Still, this is a complex problem for the following reasons (among others):

- Individual keywords have significantly different characteristics from each other; e.g., while "fishing" is a broad keyword that matches many user queries and has many competing advertisers, "humane fishing bait" is a niche keyword that matches only a few queries, but might have less competition.

- There are complex *interactions* between keywords because a user query may match two or more keywords, since the advertiser is trying to cover all the possible keywords in some domain. In effect the advertiser ends up competing with herself.

As a result, the advertisers face a challenging optimization problem. The focus of this paper is to solve this optimization problem.

1.1 The Budget Optimization Problem

We present a short discussion and formulation of the optimization problem faced by advertisers; a more detailed description is in Section 2.

A given advertiser sees the state of the auctions for search-based advertising as follows. There is a set K of keywords of interest; in practice, even small advertisers typically have a large set K. There is a set Q of queries posed by the users. For each query $q \in Q$, there are functions giving the $\text{clicks}_q(b)$ and $\text{cost}_q(b)$ that result from bidding a particular amount b in the auction for that query, which we model more formally in the next section. There is a bipartite graph G on the two vertex sets representing K and Q. For any query $q \in Q$, the neighbors of q in K are the keywords that are said to "match" the query q.[2]

The *budget optimization problem* is as follows. Given graph G together with the functions $\text{clicks}_q(\cdot)$ and $\text{cost}_q(\cdot)$ on the queries, as well as a budget U, determine the bids b_k for each keyword $k \in K$ such that $\sum_q \text{clicks}_q(b_q)$ is maximized subject to $\sum_q \text{cost}_q(b_q) \leq U$, where the "effective bid" b_q on a query is some function of the keyword bids in the neighborhood of q.

While we can cast this problem as a traditional optimization problem, there are different challenges in practice depending on the advertiser's access to the query and graph information, and indeed the reliability of this information (e.g., it could be based on unstable historical data). Thus it is important to find solutions to this problem that not only get many clicks, but are also simple, robust and less reliant on the information. In this paper we define the notion of a "uniform" strategy which is essentially a strategy that bids uniformly on all keywords. Since this type of strategy obviates the need to know anything about the particulars of the graph, and effectively aggregates the click and cost functions on the queries, it is quite robust, and thus desirable in practice. What is surprising is that uniform strategy actually performs well, which we will prove.

[2]The particulars of the matching rule are determined by the Internet search company; here we treat the function as arbitrary.

1.2 Our Main Results and Technical Overview

We present positive and negative results for the budget optimization problem. In particular, we show:

• Nearly all formulations of the problem are NP-Hard. In cases slightly more general than the formulation above, where the clicks have weights, the problem is inapproximable better than a factor of $1 - \frac{1}{e}$, unless P=NP.

• We give a $(1-1/e)$-approximation algorithm for the budget optimization problem. The strategy found by the algorithm is a *two-bid uniform strategy*, which means that it randomizes between bidding some value b_1 on all keywords, and bidding some other value b_2 on all keywords until the budget is exhausted[3]. We show that this approximation ratio is tight for uniform strategies. We also give a $(1/2)$-approximation algorithm that offers a *single-bid uniform strategy*, only using one value b_1. (This is tight for single-bid uniform strategies.) These strategies can be computed in time nearly linear in $|Q| + |K|$, the input size.

Uniform strategies may appear to be naive in first consideration because the keywords vary significantly in their click and cost functions, and there may be complex interaction between them when multiple keywords are relevant to a query. After all, the optimum can configure arbitrary bids on each of the keywords. Even for the simple case when the graph is a *matching*, the optimal algorithm involves placing different bids on different keywords via a knapsack-like packing (Section 2). So, it might be surprising that a simple two-bid uniform strategy is 63% or more effective compared to the optimum. In fact, our proof is stronger, showing that this strategy is 63% effective against a strictly more powerful adversary who can bid independently on the *individual queries*, i.e., not be constrained by the interaction imposed by the graph G.

Our proof of the $1 - 1/e$ approximation ratio relies on an adversarial analysis. We define a factor-revealing LP (Section 4) where primal solutions correspond to possible instances, and dual solutions correspond to distributions over bidding strategies. By deriving the optimal solution to this LP, we obtain both the proof of the approximation ratio, and a tight worst-case instance.

We have conducted simulations using real auction data from Google. The results of these simulations, which are highlighted at the end of Section 4, suggest that uniform bidding strategies could be useful in practice. However, important questions remain about (among other things) alternate bidding goals, on-line or stochastic bidding models [11], and game-theoretic concerns [3], which we briefly discuss in Section 8.

2. MODELING A KEYWORD AUCTION

We describe an auction from an advertiser's point of view. An advertiser bids on a *keyword,* which we can think of as a word or set of words. Users of the search engine submit *queries*. If the query "matches" a keyword that has been bid on by an advertiser, then the advertiser is entered into an auction for the available ad slots on the results page. What constitutes a "match" varies depending on the search engine.

[3]This type of strategy can also be interpreted as bidding one value (on all keywords) for part of the day, and a different value for the rest of the day.

An advertiser makes a single bid for a keyword that remains in effect for a period of time, say one day. The keyword could match many different user queries throughout the day. Each user query might have a different set of advertisers competing for clicks. The advertiser could also bid different amounts on multiple keywords, each matching a (possibly overlapping) set of user queries.

The ultimate goal of an advertiser is to maximize traffic to their website, given a certain advertising budget. We now formalize a model of keyword bidding and define an optimization problem that captures this goal.

2.1 Landscapes

We begin by considering the case of a *single* keyword that matches a *single* user query. In this section we define the notion of a "query landscape" that describes the relationship between the advertiser's bid and what will happen on this query as a result of this bid[9]. This definition will be central to the discussion as we continue to more general cases.

2.1.1 Positions, bids and click-through rate

The search results page for a query contains p possible positions in which our ad can appear. We denote the highest (most favorable) position by 1 and lowest by p.

Associated with each position i is a value $\alpha[i]$ that denotes the *click-through rate* (ctr) of the ad in position i. The ctr is a measure of how likely it is that our ad will receive a click if placed in position i. The ctr can be measured empirically using past history. We assume throughout this work that that $\alpha[i] \leq \alpha[j]$ if $j < i$, that is, higher positions receive at least as many clicks as lower positions.

In order to place an ad on this page, we must enter the *auction* that is carried out among all advertisers that have submitted a *bid* on a keyword that matches the user's query. We will refer to such an auction as a *query auction*, to emphasize that there is an auction for each query rather than for each keyword. We assume that the auction is a *generalized second price* (GSP) auction [5, 7]: the advertisers are ranked in decreasing order of bid, and each advertiser is assigned a price equal to the amount bid by the advertiser below them in the ranking.[4] In sponsored search auctions, this advertiser pays only if the user actually clicks on the ad. Let $(\mathrm{b}[1], \ldots, \mathrm{b}[p])$ denote the bids of the top p advertisers in this query auction. For notational convenience, we assume that $\mathrm{b}[0] = \infty$ and $\mathrm{b}[p] = \alpha[p] = 0$. Since the auction is a generalized second price auction, higher bids win higher positions; i.e. $b[i] \geq b[i+1]$. Suppose that we bid b on some keyword that matches the user's query, then our position is defined by the largest $\mathrm{b}[i]$ that is at most b, that is,

$$\mathrm{pos(b)} = \arg\max_i(\mathrm{b}[i] : \mathrm{b}[i] \leq \mathrm{b}). \quad (1)$$

Since we only pay if the user clicks (and that happens with probability $\alpha[i]$), our expected *cost* for winning position i would be $\mathrm{cost}[i] = \alpha[i] \cdot \mathrm{b}[i]$, where $i = \mathrm{pos(b)}$. We use $\mathrm{cost}_q(b)$ and $\mathrm{clicks}_q(b)$ to denote the expected cost and clicks that result from having a bid b that qualifies for a query auction q, and thus

$$\mathrm{cost}_q(b) = \alpha[i] \cdot \mathrm{b}[i] \quad \text{where } i = \mathrm{pos(b)}, \quad (2)$$

$$\mathrm{clicks}_q(b) = \alpha[i] \quad \text{where } i = \mathrm{pos(b)}. \quad (3)$$

The following observations about cost and clicks follow immediately from the definitions and equations (1), (2) and (3). We use $\mathbb{R}_+$ to denote the nonnegative reals.

OBSERVATION 1. *For $b \in \mathbb{R}_+$,*

1. *$(\mathrm{cost}_q(b), \mathrm{clicks}_q(b))$ can only take on one of a finite set of values $V_q = \{(\mathrm{cost}[1], \alpha[1]), \ldots, (\mathrm{cost}[p], \alpha[p])\}$.*
2. *Both $\mathrm{cost}_q(b)$ and $\mathrm{clicks}_q(b)$ are non-decreasing functions of b. Also, cost-per-click (cpc) $\mathrm{cost}_q(b)/\mathrm{clicks}_q(b)$ is non-decreasing in b.*
3. *$\mathrm{cost}_q(b)/\mathrm{clicks}_q(b) \leq b$.*

For bids $(\mathrm{b}[1], \ldots, \mathrm{b}[p])$ that correspond to the bids of other advertisers, we have: $\mathrm{cost}_q(b[i])/\mathrm{clicks}_q(b[i]) = b[i]$, $i \in [p]$. When the context is clear, we drop the subscript q.

2.1.2 Query Landscapes

We can summarize the data contained in the functions $\mathrm{cost}(b)$ and $\mathrm{clicks}(b)$ as a collection of points in a plot of cost vs. clicks, which we refer to as a *landscape*. For example, for a query with four slots, a landscape might look like Table 1.

bid range	cost per click	cost	clicks
[$2.60,∞)	$2.60	$1.30	.5
[$2.00,$2.60)	$2.00	$0.90	.45
[$1.60,$2.00)	$1.60	$0.40	.25
[$0.50,$1.60)	$0.50	$0.10	.2
[$0,$0.50)	$0	$0	0

Table 1: A *landscape* for a query

It is convenient to represent this data graphically as in Figure 1 (ignore the dashed line for now). Here we graph clicks as a function of cost. Observe that in this graph, the cpc $(\mathrm{cost}(b)/\mathrm{clicks}(b))$ of each point is the reciprocal of the slope of the line from the origin to the point. Since $\mathrm{cost}(b)$, $\mathrm{clicks}(b)$ and $\mathrm{cost}(b)/\mathrm{clicks}(b)$ are non-decreasing, the slope of the line from the origin to successive points on the plot decreases. This condition is slightly weaker than concavity.

Suppose we would like to solve the budget optimization problem for a single query landscape.[5] As we increase our bid from zero, our cost increases and our expected number of clicks increases, and so we simply submit the highest bid such that we remain within our budget.

One problem we see right away is that since there are only a finite set of points in this landscape, we may not be able to target arbitrary budgets efficiently. Suppose in the example from Table 1 and Figure 1 that we had a budget

[4]Google, Yahoo! and MSN all use some variant of the GSP auction. In the Google auction, the advertisers' bids are multiplied by a *quality score* before they are ranked; our results carry over to this case as well, which we omit from this paper for clarity. Also, other auctions besides GSP have been considered; e.g., the Vickrey Clark Groves (VCG) auction [14, 4, 7]. Each auction mechanism will result in a different sort of optimization problem. In the conclusion we point out that for the VCG auction, the bidding optimization problem becomes quite easy.

[5]Of course it is a bit unrealistic to imagine that an advertiser would have to worry about a budget if only one user query was being considered; however one could imagine multiple instances of the same query and the problem scales.

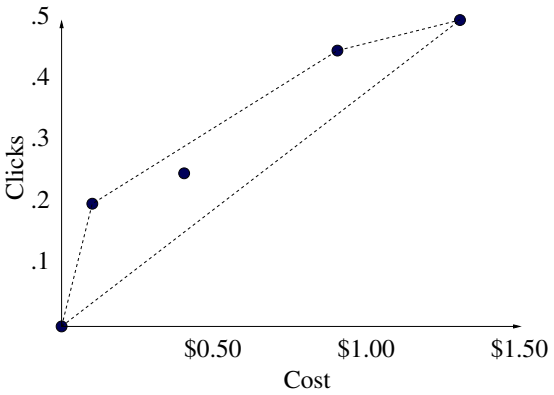

Figure 1: A bid landscape.

of $1.00. Bidding between $2.00 and $2.60 uses only $0.90, and so we are under-spending. Bidding more than $2.60 is not an option, since we would then incur a cost of $1.30 and overspend our budget.

2.1.3 Randomized strategies

To rectify this problem and better utilize our available budget, we allow *randomized bidding strategies*. Let $\mathcal{B}$ be a distribution on bids $\mathrm{b} \in \mathbb{R}_+$. Now we define $\mathrm{cost}(\mathcal{B}) = E_{\mathrm{b}\sim\mathcal{B}}[\mathrm{cost(b)}]$ and $\mathrm{clicks}(\mathcal{B}) = E_{\mathrm{b}\sim\mathcal{B}}[\mathrm{clicks(b)}]$. Graphically, the possible values of $(\mathrm{cost}(\mathcal{B}), \mathrm{clicks}(\mathcal{B}))$ lie in the convex hull of the landscape points. This is represented in Figure 1 by the dashed line.

To find a bid distribution $\mathcal{B}$ that maximizes clicks subject to a budget, we simply draw a vertical line on the plot where the cost is equal to the budget, and find the highest point on this line in the convex hull. This point will always be the convex combination of at most *two* original landscape points which themselves lie *on* the convex hull. Thus, given the point on the convex hull, it is easy to compute a distribution on two bids which led to this point. Summarizing,

LEMMA 1. *If an advertiser is bidding on one keyword, subject to a budget U, then the optimal strategy is to pick a convex combination of (at most) two bids which are at the endpoints of the line on the convex hull at the highest point for cost U.*

There is one subtlety in this formulation. Given any bidding strategy, randomized or otherwise, the resulting cost is itself a random variable representing the expected cost. Thus if our budget constraint is a hard budget, we have to deal with the difficulties that arise if our strategy would be over budget. Therefore, we think of our budget constraint as *soft*, that is, we only require that our expected cost be less than the budget. In practice, the budget is often an average daily budget, and thus we don't worry if we exceed it one day, as long as we are meeting the budget in expectation. Further, either the advertiser or the search engine (possibly both), monitor the cost incurred over the day; hence, the advertiser's bid can be changed to zero for part of the day, so that the budget is not overspent.[6] Thus in the remainder of this paper, we will formulate a budget constraint that only needs to be respected in expectation.

2.1.4 Multiple Queries: a Knapsack Problem

As a warm-up, we will consider next the case when we have a set of queries, each which its own landscape. We want to bid on each query independently subject to our budget: the resulting optimization problem is a small generalization of the *fractional knapsack* problem, and was solved in [9].

The first step of the algorithm is to take the convex hull of each landscape, as in Figure 1, and remove any landscape points not on the convex hull. Each piecewise linear section of the curve represents the incremental number of clicks and cost incurred by moving one's bid from one particular value to another. We regard these "pieces" as *items* in an instance of fractional knapsack with *value* equal to the incremental number of clicks and *size* equal to the incremental cost. More precisely, for each piece connecting two consecutive bids b' and b'' on the convex hull, we create a knapsack item with value $[\mathrm{clicks}(b'') - \mathrm{clicks}(b')]$ and size $[\mathrm{cost}(b'') - \mathrm{cost}(b')]$. We then emulate the greedy algorithm for knapsack, sorting by value/size (cost-per-click), and choosing greedily until the budget is exhausted.

In this reduction to knapsack we have ignored the fact that some of the pieces come from the same landscape and cannot be treated independently. However, since each curve is concave, the pieces that come from a particular query curve are in increasing order of cost-per-click; thus from each landscape we have chosen for our "knapsack" a set of pieces that form a prefix of the curve.

2.2 Keyword Interaction

In reality, search advertisers can bid on a large set of keywords, each of them qualifying for a different (possibly overlapping) set of queries, but most search engines do not allow an advertiser to appear twice in the same search results page.[7] Thus, if an advertiser has a bid on two different keywords that match the same query, this conflict must be resolved somehow. For example, if an advertiser has a bid out on the keywords "shoes" and "high-heel," then if a user issues the query "high-heel shoes," it will match on two different keywords. The search engine specifies, in advance, a rule for resolution based on the query the keyword and the bid. A natural rule is to take the keyword with the highest bid, which we adopt here, but our results apply to other resolution rules.

We model the keyword interaction problem using an undirected bipartite graph $G = (K \cup Q, E)$ where K is a set of keywords and Q is a set of queries. Each $q \in Q$ has an associated landscape, as defined by $\mathrm{cost}_q(\mathrm{b})$ and $\mathrm{clicks}_q(\mathrm{b})$. An edge $(k, q) \in E$ means that keyword k matches query q.

The advertiser can control their individual *keyword bid vector* $\mathbf{a} \in \mathbb{R}_+^{|K|}$ specifying a bid $\mathbf{a}_k$ for each keyword $k \in K$. (For now, we do not consider randomized bids, but we will introduce that shortly.) Given a particular bid vector $\mathbf{a}$ on the keywords, we use the resolution rule of taking the maximum to define the "effective bid" on query q as

$$\mathrm{b}_q(\mathbf{a}) = \max_{k:(k,q)\in E} \mathbf{a}_k.$$

By submitting a bid vector $\mathbf{a}$, the advertiser receives some

[6]See `https://adwords.google.com/support/bin/answer.py?answer=22183`, for example.

[7]See `https://adwords.google.com/support/bin/answer.py?answer=14179`, for example.

number of clicks and pays some cost on each keyword. We use the term *spend* to denote the total cost; similarly, we use the term *traffic* to denote the total number of clicks:

$$\text{spend}(\mathbf{a}) = \sum_{q \in Q} \text{cost}_q(\text{b}_q(\mathbf{a})); \quad \text{traffic}(\mathbf{a}) = \sum_{q \in Q} \text{clicks}_q(\text{b}_q(\mathbf{a}))$$

We also allow randomized strategies, where an advertiser gives a distribution $\mathcal{A}$ over bid vectors $\mathbf{a} \in \mathbb{R}_+^{|K|}$. The resulting spend and traffic are given by

$$\text{spend}(\mathcal{A}) = E_{\mathbf{a} \sim \mathcal{A}}[\text{spend}(\mathbf{a})]; \quad \text{traffic}(\mathcal{A}) = E_{\mathbf{a} \sim \mathcal{A}}[\text{traffic}(\mathbf{a})]$$

We can now state the problem in its full generality:

BUDGET OPTIMIZATION
Input: a budget U, a keyword-query graph $G = (K \cup Q, E)$, and landscapes $(\text{cost}_q(\cdot), \text{clicks}_q(\cdot))$ for each $q \in Q$.
Find: a distribution $\mathcal{A}$ over bid vectors $\mathbf{a} \in \mathbb{R}_+^{|K|}$ such that $\text{spend}(\mathcal{A}) \leq U$ and $\text{traffic}(\mathcal{A})$ is maximized.

We conclude this section with a small example to illustrate some feature of the budget optimization problem. Suppose you have two keywords $K = \{u, v\}$ and two queries $Q = \{x, y\}$ and edges $E = \{(u, x), (u, y), (v, y)\}$. Suppose query x has one position with ctr $\alpha^x[1] = 1.0$, and there is one bid $b_1^x = \$1$. Query y has two positions with ctrs $\alpha^y[1] = \alpha^y[2] = 1.0$, and bids $b_1^y = \$\epsilon$ and $b_2^y = \$1$ To get any clicks from x, an advertiser must bid at least \$1 on u. However, because of the structure of the graph, if the advertiser sets b_u to \$1, then his effective bid is \$1 on both x *and* y. Thus he must trade-off between getting the clicks from x and getting the bargain of a click for $\$\epsilon$ that would be possible otherwise.

3. UNIFORM BIDDING STRATEGIES

As we will show in Section 5, solving the BUDGET OPTIMIZATION problem in its full generality is difficult. In addition, it may be difficult to reason about strategies that involve arbitrary distributions over arbitrary bid vectors. Advertisers generally prefer strategies that are easy to understand, evaluate and use within their larger goals. With this motivation, we look at restricted classes of strategies that we can easily compute, explain and analyze.

We define a *uniform bidding strategy* to be a distribution $\mathcal{A}$ over bid vectors $\mathbf{a} \in \mathbb{R}_+^{|K|}$ where each bid vector in the distribution is of the form $(b, b, \ldots, b)$ for some real-valued bid b. In other words, each vector in the distribution bids the same value on every keyword.

Uniform strategies have several advantages. First, they do not depend on the edges of the interaction graph, since all effective bids on queries are the same. Thus, they are effective in the face of limited or noisy information about the keyword interaction graph. Second, uniform strategies are also independent of the priority rule being used. Third, any algorithm that gives an approximation guarantee will then be valid for *any* interaction graph over those keywords and queries.

We now show that we can compute the best *uniform strategy* efficiently.

Suppose we have a set of queries Q, where the landscape V_q for each query q is defined by the set of points $V_q = \{(\text{cost}_q[1], \alpha_q[1]), \ldots, (\text{cost}_q[p], \alpha_q[p])\}$. We define the set of *interesting bids* $I_q = \{\text{cost}_q[1]/\alpha_q[1], \ldots, \text{cost}_q[p]/\alpha_q[p]\}$, let $\mathcal{I} = \cup_{q \in Q} I_q$, and let $N = |\mathcal{I}|$. We can index the points in $\mathcal{I}$ as $b_1, \ldots, b_N$ in increasing order. The ith point in our *aggregate landscape* $\mathcal{V}$ is found by summing, over the queries, the cost and clicks associated with bid b_i, that is, $\mathcal{V} = \cup_{i=1}^{N} (\sum_{q \in Q} \text{cost}_q(b_i), \sum_{q \in Q} \text{clicks}_q(b_i))$.

For any possible bid b, if we use the aggregate landscape just as we would a regular landscape, we exactly represent the cost and clicks associated with making that bid simultaneously on all queries associated with the aggregate landscape. Therefore, all the definitions and results of Section 2 about landscapes can be extended to aggregate landscapes, and we can apply Lemma 1 to compute the best uniform strategy (using the convex hull of the points in this aggregate landscape). The running time is dominated by the time to compute the convex hull, which is $O(N \log N)$[12].

The resulting strategy is the convex combination of two points on the aggregate landscape. Define a *two-bid strategy* to be a uniform strategy which puts non-zero weight on at most two bid vectors. We have shown

LEMMA 2. *Given an instance of* BUDGET OPTIMIZATION *in which there are a total of N points in all the landscapes, we can find the best uniform strategy in $O(N \log N)$ time. Furthermore, this strategy will always be a two-bid strategy.*

Putting these ideas together, we get an $O(N \log N)$-time algorithm for BUDGET OPTIMIZATION, where N is the total number of landscape points (we later show that this is a $(1 - \frac{1}{e})$-approximation algorithm):

1. Aggregate all the points from the individual query landscapes into a single aggregate landscape.
2. Find the convex hull of the points in the aggregate landscape.
3. Compute the point on the convex hull for the given budget, which is the convex combination of two points α and β.
4. Output the strategy which is the appropriate convex combination of the uniform bid vectors corresponding to α and β.

We will also consider a special case of two-bid strategies. A *single-bid* strategy is a uniform strategy which puts non-zero weight on at most one *non-zero* vector, i.e. advertiser randomizes between bidding a certain amount b^* on all keywords, and not bidding at all. A single-bid strategy is even easier to implement in practice than a two-bid strategy. For example, the search engines often allow advertisers to set a maximum daily budget. In this case, the advertiser would simply bid b^* until her budget runs out, and the ad serving system would remove her from all subsequent auctions until the end of the day. One could also use an "ad scheduling" tool offered by some search companies[8] to implement this strategy. The best single-bid strategy can also be computed easily from the aggregate landscape. The optimal strategy for a budget U will either be the point x s.t. $\text{cost}(x)$ is as large as possible without exceeding U, or a convex combination of zero and the point y, where $\text{cost}(y)$ is as small as possible while larger than U.

[8]See `https://adwords.google.com/support/bin/answer.py?answer=33227`, for example.

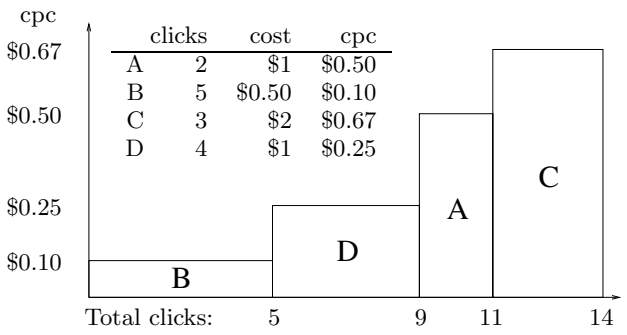

Figure 2: Four queries and their click-price curve.

4. APPROXIMATION ALGORITHMS

In the previous section we proposed using uniform strategies and gave an efficient algorithm to compute the best such strategy. In section we prove that there is always a good uniform strategy:

THEOREM 3. *There always exists a uniform bidding strategy that is $(1-\frac{1}{e})$-optimal. Furthermore, for any $\epsilon > 0$, there exists an instance for which all uniform strategies are at most $(1-\frac{1}{e}+\epsilon)$-optimal.*

We introduce the notion of a *click-price curve*, which is central to our analysis. This definition makes it simple to show that there is always a *single-bid* strategy that is a $\frac{1}{2}$-approximation (and this is tight); we then build on this to prove Theorem 3.

4.1 Click-price curves

Consider a set of queries Q, and for each query $q \in Q$, let $(\text{clicks}_q(\cdot), \text{cost}_q(\cdot))$ be the corresponding bid landscape. Consider an adversarial bidder Ω with the power to bid independently on each *query*. Note that this bidder is more powerful than an optimal bidder, which has to bid on the keywords. Suppose this strategy bids b_q^* for each query q. Thus, Ω achieves traffic $C_\Omega = \sum_i \text{clicks}(b_i^*)$, and incurs total spend $U_\Omega = \sum_i \text{cost}(b_i^*)$.

Without loss of generality we can assume that Ω bids so that for each query q, the cost per click is equal to b_q^*, i.e. $\text{cost}_q(b_q^*)/\text{clicks}_q(b_q^*) = b_q^*$. We may assume this because for some query q, if $\text{cost}_q(b_q^*)/\text{clicks}_q(b_q^*) < b_q^*$, we can always lower b_q^* and without changing the cost and clicks.

To aid our discussion, we introduce the notion of a *click-price curve* (an example of which is shown in Figure 2), which describes the cpc distribution obtained by Ω. Formally the curve is a non-decreasing function $h : [0, C_\Omega] \mapsto \mathbb{R}_+$ defined as $h(r) = \min\{c \mid \sum_{q:b_q^* \le c} \text{clicks}_q(b_q^*) \ge r\}$. Another way to construct this curve is to sort the queries in increasing order by $b_q^* = \text{cost}_q(b_q^*)/\text{clicks}_q(b_q^*)$, then make a step function where the qth step has height b_q^* and width $\text{clicks}_q(b_q^*)$ (see Figure 2). Note that the area of each step is $\text{cost}_q(b_q^*)$. The following claim follows immediately:

CLAIM 1. $U_\Omega = \int_0^{C_\Omega} h(r)dr$.

Suppose we wanted to buy some fraction r'/C_Ω of the traffic that Ω is getting. The click-price curve says that if we bid $h(r')$ on every keyword (and therefore every query), we get at least r' traffic, since this bid would ensure that for all q such that $b_q^* \le h(r')$ we win as many clicks as Ω. Note that by bidding $h(r')$ on every keyword, we may actually get even more than r' traffic, since for queries q where b_q^* is much less than $h(r')$ we may win more clicks than Ω. However, all of these extra clicks still cost at most $h(r')$ per click.

Thus we see that for any $r' \in [0, C_\Omega]$, if we bid $h(r')$ on every keyword, we receive at least r' traffic at a total spend of at most $h(r')$ per click. Note that by randomizing between bidding zero and bidding $h(r')$, we can receive *exactly* r' traffic at a total spend of at most $r' \cdot h(r')$. We summarize this discussion in the following lemma:

LEMMA 4. *For any $r \in [0, C_\Omega]$, there exists a single-bid strategy that randomizes between bidding $h(r)$ and bidding zero, and this strategy receives exactly r traffic with total spend at most $r \cdot h(r)$.*

Lemma 4 describes a landscape as a continuous function. For our lower bounds, we will need to show that given any continuous function, there exists a discrete landscape that approximates it arbitrarily well.

LEMMA 5. *For any $C, U > 0$ and non-decreasing function $f : [0, C] \to \mathbb{R}_+$ such that $\int_0^C f(r)dr = U$, and any small $\epsilon' > 0$, there exists an instance of* BUDGET OPTIMIZATION *with budget $U + \epsilon'$, where the optimal solution achieves C clicks at cost $U + \epsilon'$, and all uniform bidding strategies are convex combinations of single-bid strategies that achieve exactly r clicks at cost exactly $rf(r)$ by bidding $f(r)$ on all keywords.*

PROOF. Construct an instance as follows. Let $\epsilon > 0$ be a small number that we will later define in terms of ϵ'. Define $r_0 = 0, r_1, r_2, \ldots, r_m = C$ such that $r_{i-1} < r_i \le r_{i-1} + \epsilon$, $f(r_{i-1}) \le f(r_i) \le f(r_{i-1}) + \epsilon$, and $m \le (C + f(C))/\epsilon$. (This is possible by choosing r_i's spaced by $\min(\epsilon, f(r_i) - f(r_{i-1}))$) Now make a query q_i for all $i \in [m]$ with bidders bidding $h(r_i), h(r_{i+1}), \ldots, h(r_m)$, and ctrs $\alpha[1] = \alpha[2] = \cdots = \alpha[m - i + 1] = r_i - r_{i-1}$. The graph is a matching with one keyword per query, and so we can imagine the optimal solution as bidding on queries. The optimal solution will always bid exactly $h(r_i)$ on query q_i, and if it did so on all queries, it would spend $U' := \sum_{i=1}^m (r_i - r_{i-1})h(r_i)$. Define ϵ small enough so that $U' = U + \epsilon'$, which is always possible since

$$\begin{aligned} U' &\le \int_0^C f(r)dr + \sum_{i=1}^m (r_i - r_{i-1})(h(r_i) - h(r_{i-1})) \\ &\le U + \epsilon^2 m \le U + \epsilon(C + f(C)). \end{aligned}$$

Note that the only possible bids (i.e., all others have the same results as one of these) are $f(r_0), \ldots, f(r_m)$, and bidding uniformly with $f(r_i)$ results in $\sum_{j=1}^i r_i - r_{i-1} = r_i$ clicks at cost $h(r_i)r_i$. □

4.2 A $\frac{1}{2}$-approximation algorithm

Using Lemma 4 we can now show that there is a uniform single-bid strategy that is $\frac{1}{2}$-optimal. In addition to being an interesting result in its own right, it also serves as a warm-up for our main result.

THEOREM 6. *There always exists a uniform single-bid strategy that is $\frac{1}{2}$-optimal. Furthermore, for any $\epsilon > 0$ there exists an instance for which all single-bid strategies are at most $(\frac{1}{2} + \epsilon)$-optimal.*

PROOF. Applying Lemma 4 with $r = C_\Omega/2$, we see that there is a strategy that achieves traffic $C_\Omega/2$ with spend $C_\Omega/2 \cdot h(C_\Omega/2)$. Now, using the fact that h is a non-decreasing function combined with Claim 1, we have

$$(C_\Omega/2)h(C_\Omega/2) \leq \int_{C_\Omega/2}^{C_\Omega} h(r)dr \leq \int_0^{C_\Omega} h(r)dr = U_\Omega, \quad (4)$$

which shows that we spend at most U_Ω. We conclude that there is a $\frac{1}{2}$-optimal single-bid strategy randomizing between bidding $C_\Omega/2$ and zero.

For the second part of the theorem, we construct a tight example using two queries $Q = \{x, y\}$, two keywords $K = \{u, v\}$, and edges $E = \{(u, x), (v, y)\}$.

Fix some α where $0 < \alpha \leq 1$, and fix some very small $\epsilon > 0$. Query x has two positions, with bids of $b_1^x = 1/\alpha$ and $b_2^x = \epsilon$, and with identical click-through rates $\alpha^x[1] = \alpha^x[2] = \alpha$. Query y has one position, with a bid of $b_1^y = 1/\alpha$ and a click-through rate of $\alpha^y[1] = \alpha$. The budget is $U = 1 + \epsilon\alpha$. The optimal solution is to bid ϵ on u (and therefore x) and bid $1/\alpha$ on v (and therefore y), both with probability 1. This achieves a total of 2α clicks and spends the budget exactly. The only useful bids are 0, ϵ and $1/\alpha$, since for both queries all other bids are identical in terms of cost and clicks to one of those three. Any single-bid solution that uses ϵ as its non-zero bid gets at most α clicks. Bidding $1/\alpha$ on both keywords results in 2α clicks and total cost 2. Thus, since the budget is $U = 1 + \epsilon\alpha < 2$, a single-bid solution using $1/\alpha$ can put weight at most $(1+\epsilon\alpha)/2$ on the $1/\alpha$ bid. This results in at most $\alpha(1 + \epsilon\alpha)$ clicks. This can be made arbitrarily close to α by lowering ϵ. □

4.3 A $(1 - \frac{1}{e})$-approximation algorithm

The key to the proof of Theorem 3 is to show that there is a distribution over single-bid strategies from Lemma 4 that obtains at least $(1 - \frac{1}{e})C_\Omega$ clicks. In order to figure out the best distribution, we wrote a linear program that models the behavior of a player who is trying to maximize clicks and an adversary who is trying to create an input that is hard for the player. Then using linear programming duality, we were able to derive both an optimal strategy and a tight instance. After solving the LP numerically, we were also able to see that there is a uniform strategy for the player that always obtains $(1 - \frac{1}{e})C_\Omega$ clicks; and then from the solution were easily able to "guess" the optimal distribution. This methodology is similar to that used in work on factor-revealing LPs [8, 10].

4.3.1 An LP for the worst-case click-price curve.

Consider the adversary's problem of finding a click-price curve for which no uniform bidding strategy can achieve αC_Ω clicks. Recall that by Lemma 1 we can assume that a uniform strategy randomizes between two bids u and v. We also assume that the uniform strategy uses a convex combination of strategies from Lemma 4, which we can assume by Lemma 5. Thus, to achieve αC_Ω clicks, a uniform strategy must randomize between bids $h(u)$ and $h(v)$ where $u \leq \alpha C_\Omega$ and $v \geq \alpha C_\Omega$. Call the set of such strategies S. Given a $(u, v) \in S$, the necessary probabilities in order to achieve αC_Ω clicks are easily determined, and we denote them by $p_1(u, v)$ and $p_2(u, v)$ respectively. Note further that the advertiser is trying to figure out which of these strategies to use, and ultimately wants to compute a distribution over uniform strategies. In the LP, she is actually going to compute a distribution over pairs of strategies in S, which we will then interpret as a distribution over strategies.

Using this set of uniform strategies as constraints, we can characterize a set of worst-case click-price curves by the constraints

$$\int_0^{C_\Omega} h(r)dr \leq U$$

$$\forall (u, v) \in S \quad p_1(u, v)uh(u) + p_2(u, v)vh(v) \geq U$$

A curve h that satisfies these constraints has the property that all uniform strategies that obtain αC_Ω clicks spend more than U. Discretizing this set of inequalities, and pushing the first constraint into the objective function, we get the following LP over variables h_r representing the curve:

$$\begin{aligned} \min \quad & \sum_{r \in \{0, \epsilon, 2\epsilon, \ldots, C_\Omega\}} \epsilon \cdot h_r \quad \text{s.t.} \\ \forall (u, v) \in S, \quad & p_1(u, v)uh_u + p_2(u, v)vh_v \geq U \end{aligned}$$

In this LP, S is defined in the discrete domain as $S = \{(u, v) \in \{0, \epsilon, 2\epsilon, \ldots, C_\Omega\}^2 : 0 \leq u \leq \alpha C_\Omega \leq v \leq C_\Omega\}$.

Solving this LP for a particular α, if we get an objective less than U, we know (up to some discretization) that an instance of BUDGET OPTIMIZATION exists that cannot be approximated better than α. (The instance is constructed as in the proof of Lemma 5.) A binary search yields the smallest such α where the objective is exactly U.

To obtain a strategy for the advertiser, we look at the dual, constraining the objective to be equal to U in order to get the polytope of optimum solutions:

$$\begin{aligned} & \sum_{(u,v) \in S} w_{u,v} = 1 \\ \forall (u, v) \in S, \quad & \sum_{v' : (u,v') \in S} p_1(u, v') \cdot u \cdot w_{u,v'} \leq \epsilon \quad \text{and} \\ & \sum_{u' : (u',v) \in S} p_2(u', v) \cdot v \cdot w_{u',v} \leq \epsilon. \end{aligned}$$

It is straightforward to show that the second set of constraints is equivalent to the following:

$$\begin{aligned} \forall h \in \mathbb{R}^{C_\Omega/\epsilon} : \sum_r \epsilon h_r &= U, \\ \sum_{(u,v) \in S} w_{u,v}(p_1(u, v) \cdot u \cdot h_u + p_2(u, v) \cdot v \cdot h_v) &\leq U. \end{aligned}$$

Here the variables can be interpreted as weights on strategies in S. A point in this polytope represents a convex combination over strategies in S, with the property that for *any* click-price curve h, the cost of the mixed strategy is at most U. Since all strategies in S get at least αC_Ω clicks, we have a strategy that achieves an α-approximation. Interestingly, the equivalence between this polytope and the LP dual above shows that there is a mixture over values $r \in [0, C]$ that achieves an α-approximation for *any curve* h.

After a search for the appropriate α (which turned out to be $1 - \frac{1}{e}$), we solved these two LPs and came up with the plots in Figure 3, which reveal not only the right approximation ratio, but also a picture of the worst-case distribution and the approximation-achieving strategy.[9] From the pic-

[9]The parameters U and C_Ω can be set arbitrarily using scaling arguments.

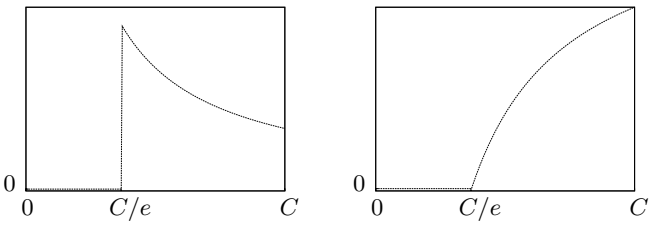

Figure 3: The worst-case click-price curve and $(1-1/e)$-approximate uniform bidding strategy, as found by linear programming.

tures, we were able to quickly "guess" the optimal strategy and worst case example.

4.3.2 Proof of Theorem 3

By Lemma 4, we know that for each $r \leq U_\Omega$, there is a strategy that can obtain traffic r at cost $r \cdot h(r)$. By mixing strategies for multiple values of r, we construct a uniform strategy that is guaranteed to achieve at least $1-e^{-1} = 0.63$ fraction of Ω's traffic and remain within budget. Note that the "final" resulting bid distribution will have some weight on the zero bid, since the single-bid strategies from Lemma 4 put some weight on bidding zero.

Consider the following probability density function over such strategies (also depicted in Figure 3):

$$g(r) = \begin{cases} 0 & \text{for } r < C_\Omega/e, \\ 1/r & \text{for } r \geq C_\Omega/e. \end{cases}$$

Note that $\int_0^{C_\Omega} g(r)\,dr = \int_{C_\Omega/e}^{C_\Omega} \frac{1}{r} dr = 1$, i.e. g is a probability density function. The traffic achieved by our strategy is equal to

$$\text{traffic} \;=\; \int_0^{C_\Omega} g(r)\cdot r\,dr \;=\; \int_{C_\Omega/e}^{C_\Omega} \frac{1}{r}\cdot r\,dr \;=\; \left(1-\frac{1}{e}\right) C_\Omega.$$

The expected total spend of this strategy is at most

$$\begin{aligned} \text{spend} &= \int_0^{C_\Omega} g(r)\cdot r h(r)\,dr \\ &= \int_{C_\Omega/e}^{C_\Omega} h(r)\,dr \leq \int_0^{C_\Omega} h(r)\,dr = U_\Omega. \end{aligned}$$

Thus we have shown that there exists a uniform bidding strategy that is $(1-\frac{1}{e})$-optimal.

We now show that no uniform strategy can do better. We will prove that for all $\epsilon > 0$ there exists an instance for which all uniform strategies are at most $(1-\frac{1}{e}+\epsilon)$-optimal.

First we define the following click-price curve over the domain $[0,1]$:

$$h(r) = \begin{cases} 0 & \text{for } r < e^{-1} \\ \dfrac{1}{e-2}\left(e-\dfrac{1}{r}\right) & \text{for } r \geq e^{-1} \end{cases}$$

Note that h is non-decreasing and non-negative. Since the curve is over the domain $[0,1]$ it corresponds to an instance where $C_\Omega = 1$. Note also that $\int_0^1 h(r)\,dr = \frac{1}{e-2}\int_{1/e}^1 e - \frac{1}{r}\,dr = 1$. Thus, this curve corresponds to an instance where $U_\Omega = 1$. Using Lemma 5, we construct an actual instance where the best uniform strategies are convex combinations of strategies that bid $h(u)$ and achieve u clicks and $u \cdot h(u)$ cost.

Suppose for the sake of contradiction that there exists a uniform bidding strategy that achieves $\alpha > 1-e^{-1}$ traffic on this instance. By Lemma 1 there is always a two-bid optimal uniform bidding strategy and so we may assume that the strategy achieving α clicks randomizes over two bids. To achieve α clicks, the two bids must be on values $h(u)$ and $h(v)$ with probabilities p_u and p_v such that $p_u + p_v = 1$, $0 \leq u \leq \alpha \leq v$ and $p_u u + p_v v = \alpha$.

To calculate the spend of this strategy consider two cases: if $u = 0$ then we are bidding $h(v)$ with probability $p_v = \alpha/v$. The spend in this case is:

$$\text{spend} = p_v \cdot v \cdot h(v) = \alpha h(v) = \frac{\alpha e - \alpha/v}{e-2}.$$

Using $v \geq \alpha$ and then $\alpha > 1 - \frac{1}{e}$ we get

$$\text{spend} \geq \frac{\alpha e - 1}{e-2} > \frac{(1-1/e)e-1}{e-2} = 1,$$

contradicting the assumption.

We turn to the case $u > 0$. Here we have $p_u = \frac{v-\alpha}{v-u}$ and $p_v = \frac{\alpha-u}{v-u}$. Note that for $r \in (0,1]$ we have $h(r) \geq \frac{1}{e-2}(e-\frac{1}{r})$. Thus

$$\begin{aligned} \text{spend} &\geq p_u \cdot u h(u) + p_v \cdot v h(v) \\ &= \frac{(v-\alpha)(ue-1) + (\alpha-u)(ve-1)}{(v-u)(e-2)} \\ &= \frac{\alpha e - 1}{e-2} > 1. \end{aligned}$$

The final inequality follows from $\alpha > 1 - \frac{1}{e}$. Thus in both cases the spend of our strategy is over the budget of 1. □

4.4 Experimental Results

We ran simulations using the data available at Google which we briefly summarize here. We took a large advertising campaign, and, using the set of keywords in the campaign, computed three different curves (see Figure 4) for three different bidding strategies. The x-axis is the budget (units removed), and the y-axis is the number of clicks obtained (again without units) by the optimal bid(s) under each respective strategy. "Query bidding" represents our (unachievable) upper bound Ω, bidding on each query independently. The "uniform bidding" curves represent the results of applying our algorithm: "deterministic" uses a single bid level, while "randomized" uses a distribution. For reference, we include the lower bound of a $(e-1)/e$ fraction of the top curve.

The data clearly demonstrate that the best single uniform bid obtains almost all the possible clicks in practice. Of course in a more realistic environment without full knowledge, it is not always possible to find the best such bid, so further investigation is required to make this approach useful. However, just knowing that there is such a bid available should make the on-line versions of the problem simpler.

5. HARDNESS RESULTS

By a reduction from vertex cover we can show the following (proof omitted):

THEOREM 7. BUDGET OPTIMIZATION *is strongly NP-hard.*

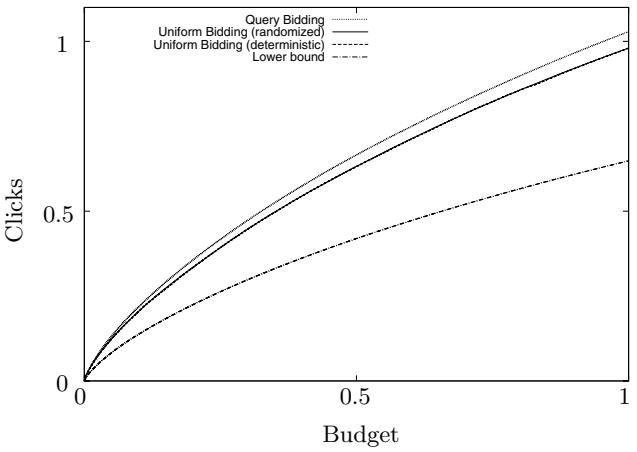

Figure 4: An example with real data.

Now suppose we introduce weights on the queries that indicate the relative value of a click from the various search users. Formally, we have weights w_q for all $q \in Q$ and our goal is maximize the total *weighted* traffic given a budget. Call this the WEIGHTED KEYWORD BIDDING problem.

With this additional generalization we can show hardness of approximation via a simple reduction from the MAXIMUM COVERAGE problem, which is known to be $(1-1/e)$-hard [6] (proof omitted).

THEOREM 8. *The* WEIGHTED KEYWORD BIDDING *problem is hard to approximate to within* $(1-1/e)$.

6. EXACT ALGORITHMS FOR LAMINAR GRAPHS

If a graph has special structure, we can sometimes solve the budget optimization problem exactly. Note that the knapsack algorithm in Section 2 solves the problem for the case when the graph is a simple matching. Here we generalize this to the case when the graph has a laminar structure, which will allow us to impose a (partial) ordering on the possible bid values, and thereby give a pseudopolynomial algorithm via dynamic programming.

We first show that to solve the BUDGET OPTIMIZATION problem (for general graphs) optimally in pseudopolynomial time, it suffices to provide an algorithm that solves the deterministic case. The proof (omitted) uses ideas similar to Observation 1 and Lemma 1.

LEMMA 9. *Let I be an input to the* BUDGET OPTIMIZATION *problem and suppose that we find the optimal deterministic solution for every possible budget $U' \le U$. Then we can find the optimal solution in time $O(U \log U)$.*

A collection S of n sets $S_1, \ldots, S_2$ is *laminar* if, for any two sets S_i and S_j, if $S_i \cap S_j \neq \emptyset$ then either $S_i \subseteq S_j$ or $S_j \subseteq S_i$.

Given a keyword interaction graph G, we associate a set of neighboring queries $Q_k = \{q : (k,q) \in E\}$ with each keyword k. If this collection of sets if laminar, we say that the graph has the *laminar property*. Note that a laminar interaction graph would naturally fall out as a consequence of designing a "hierarchical" keyword set (e.g., "shoes," "high-heel shoes," "athletic shoes").

We call a solution *deterministic* if it consists of one bid vector, rather than a general distribution over bid vectors. The following lemma will be useful for giving a structure to the optimal solution, and will allow dynamic programming.

LEMMA 10. *For keywords $i, j \in K$, if $Q_i \subseteq Q_j$ then there exists an optimal deterministic solution to the* BUDGET OPTIMIZATION *problem with $\mathbf{a}_i \ge \mathbf{a}_j$.*

We can view the laminar order as a tree with keyword j as a parent of keyword i if Q_j is the minimal set containing Q_i. In this case we say that j is a child of i. Given a keyword j with c children $i_1, \ldots, i_c$, we now need to enumerate over all ways to allocate the budget among the children and also over all possible minimum bids for the children. A complication is that a node may have many children and thus a term of U^c would not even be pseudopolynomial. We can solve this problem by showing that given any laminar ordering, there is an equivalent one in which each keyword has at most 2 children.

LEMMA 11. *Let G be a graph with the laminar property. There exists another graph G' with the same optimal solution to the* BUDGET OPTIMIZATION *problem, where each node has at most two children in the laminar ordering. Furthermore, G' has at most twice as many nodes as G.*

Given a graph with at most two children per node, we define $F[i,b,U]$ to be the maximum number of clicks achievable by bidding at least b on each of keywords j s.t. $Q_j \subseteq Q_i$ (and exactly b on keyword i) while spending at most U. For this definition, we use $Z(b,U)$ to denote set of allowable bids and budgets over children:

$$\begin{aligned} Z(b,U) &= \{b,' b'', U', U'' : b' \ge b, U' \le U, \\ &\qquad b'' \ge b, U'' \le U, U' + U'' \le U\} \end{aligned}$$

Given a keyword i and a bid $\mathbf{a}_i$, compute an incremental spend and traffic associated with bidding $\mathbf{a}_i$ on keyword i, that is

$$\begin{aligned} \hat{t}(i, \mathbf{a}_i) &= \sum_{q \in Q_i \setminus Q_{i-1}} \text{clicks}_q(\mathbf{a}_i), \quad \text{and} \\ \hat{s}(i, \mathbf{a}_i) &= \sum_{q \in Q_i \setminus Q_{i-1}} \text{cost}_q(\mathbf{a}_i). \end{aligned}$$

Now we define $F[i,b,U]$ as

$$\max_{\substack{b,' b'', U', U'' \\ \in Z(b,U)}} \left\{ F[j', b', U'] + F[j'', b'', U''] + \hat{t}(i,b) \right\} \qquad (5)$$

if $(\hat{s}(i,b) \le U - U' - U''$ and $i > 0)$, and $F[i,b,U] = 0$ otherwise.

LEMMA 12. *If the graph G has the laminar property, then, after applying Lemma 11, the dynamic programming recurrence in (5) finds an optimal deterministic solution to the* BUDGET OPTIMIZATION *problem exactly in $O(B^3 U^3 n)$ time.*

In addition, we can apply Lemma 9 to compute the optimal (randomized) solution. Observe that in the dynamic program, we have already solved the instance for every budget $U' \le U$, so we can find the randomized solution with no additional asymptotic overhead.

LEMMA 13. *If the graph G has the laminar property, then, by applying Lemma 11, the dynamic programming recurrence in (5), and Lemma 9, we can find an optimal solution to the* BUDGET OPTIMIZATION *problem in* $O(B^3U^3n)$ *time.*

The bounds in this section make pessimistic assumptions about having to try *every* budget and *every* level. For many problems, you only need to choose from a discrete set of bid levels (e.g., multiples of one cent). Doing so yields the obvious improvement in the bounds.

7. BID OPTIMIZATION UNDER VCG

The GSP auction is not the only possible auction one could use for sponsored search. Indeed the VCG auction and variants [14, 4, 7, 1] offer alternatives with compelling game-theoretic properties. In this section we argue that the budget optimization problem is easy under the VCG auction.

For a full definition of VCG and its application to sponsored search we refer the reader to [1, 2, 5]. For the sake of the budget optimization problem we can define VCG by just redefining $\text{cost}_q(b)$ (replacing Equation (2)):

$$\text{cost}_q(\text{b}) = \sum_{j=i}^{p-1} (\alpha[j] - \alpha[j+1]) \cdot \text{b}[j] \quad \text{where } i = \text{pos(b)}.$$

Observation 1 still holds, and we can construct a landscape as before, where each landscape point corresponds to a particular bid $b[i]$.

We claim that in the VCG auction, the landscapes are convex. To see this, consider two consecutive positions i,$i+1$. The slope of line segment between the points corresponding to those two positions is

$$\frac{\text{cost}(b[i]) - \text{cost}(b[i+1])}{\text{clicks}(b[i]) - \text{clicks}(b[i+1])} = \frac{(\alpha[i] - \alpha[i+1]) \cdot b[i]}{\alpha[i] - \alpha[i+1]} = b[i].$$

Since $b[i] \geq b[i+1]$, the slopes of the "pieces" of the landscape decrease, and we get that the curve is convex.

Now consider running the algorithm described in Section 2.1.4 for finding the optimal bids for a set of queries. In this algorithm we took all the pieces from the landscape curves, sorted them by incremental cpc, then took a prefix of those pieces, giving us bids for each of the queries. But, the equation above shows that each piece has its incremental cpc equal to the bid that achieves it; thus in the case of VCG the pieces are also sorted by bid. Hence we can obtain any prefix of the pieces via a uniform bid on all the keywords. We conclude that the best uniform bid is an optimal solution to the budget optimization problem.

8. CONCLUDING REMARKS

Our algorithmic result presents an intriguing heuristic in practice: bid a single value b on all keywords; at the end of the day, if the budget is under-spent, adjust b to be higher; if budget is overspent, adjust b to be lower; else, maintain b. If the scenario does not change from day to day, this simple strategy will have the same theoretical properties as our one-bid strategy, and in practice, is likely to be much better. Of course the scenario does change, however, and so coming up with a "stochastic" bidding strategy remains an important open direction, explored somewhat by [11, 13].

Another interesting generalization is to consider weights on the clicks, which is a way to model *conversions*. (A conversion corresponds to an action on the part of the user who clicked through to the advertiser site; e.g., a sale or an account sign-up.) Finally, we have looked at this system as a black box returning clicks as a function of bid, whereas in reality it is a complex repeated game involving multiple advertisers. In [3], it was shown that when a set of advertisers use a strategy similar to the one we suggest here, under a slightly modified first-price auction, the prices approach a well-understood market equilibrium.

Acknowledgments

We thank Rohit Rao, Zoya Svitkina and Adam Wildavsky for helpful discussions.

9. REFERENCES

[1] G. Aggarwal, A. Goel and R. Motwani. Truthful auctions for pricing search keywords. *ACM Conference on Electronic Commerce*, 1-7, 2006.

[2] G. Aggarwal, J. Feldman and S. Muthukrishnan Bidding to the Top: VCG and Equilibria of Position-Based Auctions *Proc. WAOA*, 2006.

[3] C. Borgs, J. Chayes, O. Etesami, N. Immorlica, K. Jain, and M. Mahdian. Dynamics of bid optimization in online advertisement auctions. *Proc. WWW* 2007.

[4] E. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971.

[5] B. Edelman, M. Ostrovsky and M. Schwarz. Internet Advertising and the Generalized Second Price Auction: Selling Billions of Dollars Worth of Keywords. *Second workshop on sponsored search auctions*, 2006.

[6] U. Feige. A threshold of $\ln n$ for approximating set cover. *28th ACM Symposium on Theory of Computing*, 1996, pp. 314–318.

[7] T. Groves. Incentives in teams. *Econometrica*, 41(4): 617-631, 1973.

[8] K. Jain, M. Mahdian, E. Markakis, A. Sabieri and V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *J. ACM*, 50(6): 795-824, 2003.

[9] W. Labio, M. Rose, S. Ramaswamy. *Internal Document, Google, Inc.* May, 2004.

[10] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani, Adwords and Generalized Online Matching. *FOCS* 2005.

[11] S. Muthukrishnan, M. Pál and Z. Svitkina. Stochastic models for budget optimization in search-based advertising. To appear in *3rd Workshop on Sponsored Search Auctions*, WWW 2007.

[12] F. Preparata and M. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.

[13] P. Rusmevichientong and D. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services *Proc. 7th ACM conference on Electronic commerce*, 260 - 269, 2006.

[14] W. Vickrey. Counterspeculation, auctions and competitive-sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.