Stochastic Models for Budget Optimization in Search-Based Advertising

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

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

Zoya Svitkina*
Dept. of Computer Science
Cornell University
Ithaca, NY
zoya@cs.cornell.edu

ABSTRACT

Internet search companies sell advertisement slots based on users' search queries via an auction. Advertisers have to solve a complex optimization problem of how to place bids on the keywords of their interest so that they can maximize their return (the number of user clicks on their ads) for a given budget. This is the *budget optimization* problem.

In this paper, we model budget optimization as it arises in Internet search companies and formulate *stochastic* versions of the problem. The premise is that Internet search companies can predict *probability distributions* associated with queries in the future. We identify three natural stochastic models. In the spirit of other stochastic optimization problems, two questions arise.

- *(Evaluation Problem)* Given a bid solution, can we evaluate the expected value of the objective function under different stochastic models?
- *(Optimization Problem)* Can we determine a bid solution that maximizes the objective function in expectation under different stochastic models?

Our main results are algorithmic and complexity results for both these problems for our three stochastic models. In particular, our algorithmic results show that simple *prefix* strategies that bid on all cheap keywords up to some level are either optimal or good approximations for many cases; we show other cases to be NP-hard.

1. INTRODUCTION

Internet search companies use auctions to sell advertising slots, in response to users' search queries. To participate in these auctions, the advertisers select a set of keywords that are relevant or descriptive of their business, and submit bids on each of them. Upon seeing a user's query, the search company runs an auction among the advertisers who have placed bids for keywords matching the query and arranges the winners in slots. The advertisers pay only if the user clicks on their ad. Advertiser's bid affects the position of their ad, which in turn affects the number of clicks received and the cost incurred. In addition to the bids, the advertiser specifies a daily budget. When the cost charged for the clicks reaches the budget, the advertiser's ads stop participating in the auctions. Thus the budget puts a cap on the amount of money spent in a day.

In what follows, we describe the vantage of an advertiser trying to optimize the return on their investment in such an auction and abstract the budget optimization problem; we also formulate models for predicting the future and derive stochastic variants of the budget optimization problem; finally, we state and prove our results.

1.1 Advertiser's Budget Optimization problem

We adapt the viewpoint of an advertiser and study the optimization problem she faces. The advertiser has to determine the daily budget, a good set of keywords, and bids for these keywords so as to maximize the effectiveness of her campaign. The daily budget and the choice of keywords are strategic and hard to model without specific knowledge. As a result, they are assumed to be given in our problem formulation. Effectiveness of a campaign is difficult to model too since clicks resulting from some keywords may be more desirable than others, and in some cases, just appearing on the results page for a user's query may have some utility. For most of the paper, we adapt the common measure of the effectiveness of the campaign, namely, the *number* of clicks.[1] Further, seen from an individual advertiser's point of view, the budgets and bids of other advertisers are fixed for the day.

For most of the paper, we consider the *single slot* case. Here, each keyword i has some threshold bid amount, bid_i, such that if the advertiser bids below bid_i, then she loses the auction and does not get any clicks. If she bids bid_i or above for keyword i, then she gets a number of clicks $\mathbf{clicks}_i$ for the queries that match keyword i, and has to pay cost-per-click $\mathbf{cpc}_i$ for each click. Under this setting, the only decision that has to be made about keyword i is whether to bid on it above its threshold or not. As a result, the value of bid_i becomes immaterial from the point of view of the optimization problem. Instead, we use decision variables b_i that represent *whether or not* to bid on keyword i. The decision variables b_i can be either *integral* or *fractional.* If they are integral, then $b_i \in \{0, 1\}$, indicating whether or not there

*This work was done while visiting Google, Inc., New York, NY.

WWW2007, May 8–12, 2007, Banff, Canada.

[1] Later we show how to extend our results to a more general model in which clicks for different keywords may have different values.

is a bid on keyword i. A fractional bid $b_i \in [0,1]$ represents bidding for b_i fraction of the queries that correspond to keyword i, or equivalently bids on each such query with probability b_i. Then the advertiser gets $b_i \cdot \mathbf{clicks}_i$ clicks for keyword i. Integer bid solutions are slightly simpler to implement than fractional bids and are more desirable when they exist.

Finally, consider the effect of the future on an advertiser. We abstract it using the function $\mathbf{clicks}_i$ which is the number of clicks the advertiser gets for queries that correspond to keyword i. Each such click entails a cost $\mathbf{cpc}_i$ for the advertiser which is determined by the rules of the auction.[2] There is a subtlety now because the advertiser is *budget-constrained*. There are choices of bids for which the cost will exceed the budget. One possibility would be to disallow all solutions that may exceed the budget. But this is unreasonable if we imagine an advertiser with a small budget who wants to bid on a popular or a highly varying keyword: to get any clicks, she should bid on that keyword, even if sometimes the budget runs out by the middle of the day. So we allow solutions that may exceed the budget, but we scale down the number of clicks obtained. Consider a solution $\boldsymbol{b}$ that bids on some keywords. If the budget were unlimited, then bidding on those keywords would bring $\mathbf{clicks}(\boldsymbol{b})$ clicks and all of these clicks together would cost $\mathbf{cost}(\boldsymbol{b})$. But when budget B is smaller than $\mathbf{cost}(\boldsymbol{b})$, this solution runs out of money before the end of the day, and misses the clicks that come after that point. If we assume that the queries and clicks for all keywords are distributed uniformly throughout the day and are well-mixed, then this solution reaches the budget after $B/\mathbf{cost}(\boldsymbol{b})$ fraction of the day passes, missing $(1 - B/\mathbf{cost}(\boldsymbol{b}))$ fraction of the possible clicks for each keyword. As a result, the number of clicks collected before the budget is exceeded is $\frac{\mathbf{clicks}(\boldsymbol{b})}{\mathbf{cost}(\boldsymbol{b})/B}$ in expectation.

Based on the discussion so far, we can now state the optimization problem an advertiser faces.

Definition 1 Budget Optimization Problem *(BO). An advertiser has a set T of keywords, with $|T| = n$, and a budget B. For each keyword $i \in T$, we are given $\mathbf{clicks}_i$, the number of clicks that correspond to i, and $\mathbf{cpc}_i$, the cost per click of these clicks. We define $\mathbf{cost}_i = \mathbf{cpc}_i \cdot \mathbf{clicks}_i$. The objective is to find a solution $\boldsymbol{b} = (b_1, ..., b_n)$ with a bid b_i for each $i \in T$ to maximize*

$$\mathbf{value}(\boldsymbol{b}) = \frac{\sum_{i\in T} b_i \mathbf{clicks}_i}{\max\left(1, \sum_{i\in T} b_i \mathbf{cost}_i/B\right)}. \quad (1)$$

The numerator of the objective function is the number of clicks available to $\boldsymbol{b}$, and the denominator scales it down in the case that the budget is exceeded. If we define $\mathbf{clicks}(\boldsymbol{b}) = \sum_{i\in T} b_i \mathbf{clicks}_i$, $\mathbf{cost}(\boldsymbol{b}) = \sum_{i\in T} b_i \mathbf{cost}_i$, and the average cost per click of solution $\boldsymbol{b}$ as $\mathbf{cpc}(\boldsymbol{b}) = \frac{\mathbf{cost}(\boldsymbol{b})}{\mathbf{clicks}(\boldsymbol{b})}$, then

$$\mathbf{value}(\boldsymbol{b}) = \begin{cases} \mathbf{clicks}(\boldsymbol{b}) & \text{if } \mathbf{cost}(\boldsymbol{b}) \le B \\ B/\mathbf{cpc}(\boldsymbol{b}) & \text{if } \mathbf{cost}(\boldsymbol{b}) > B \end{cases} \quad (2)$$

So maximizing $\mathbf{value}(\boldsymbol{b})$ is equivalent to maximizing the number of clicks in case that we are under budget, and minimizing the average cost per click if we are over budget.

[2]In Internet search, *second price* auctions are common. Here, informally, the cost for the advertiser who wins the slot is the highest bid of the others who lost the auction.

1.2 Stochastic versions

There are many variables that affect the number of clicks that an advertiser receives in a day. Besides the factors such as the advertiser's choice of her own budget and keywords which we take to be given, and the budgets of other advertisers and their choices which remain fixed, the main *variable* in our problem is the number of queries of relevance that users issue on that day, and the frequency with which the ads are clicked.[3] These quantities are not known precisely in advance. Our premise is that Internet search companies can use statistical methods to analyze past data and predict future with reasonable accuracy. They currently do provide limited amount of information about the range of values taken by these parameters.[4] Therefore, they can provide probability distributions for parameters of interest, thereby giving useful information regarding what *could* happen or what *is likely* to happen. This motivates us to study the problem in the *stochastic* setting where the goal is to maximize the *expected* value of the objective under such probability distributions.

In our stochastic versions, the numbers of clicks $\mathbf{clicks}_i$ corresponding to different keywords are random variables having some joint distribution. But general joint distributions are difficult to represent and to work with, so we formulate the following natural stochastic models. (In contrast, the problem in the previous section where $\mathbf{clicks}_i$ are known precisely for all i is called the *fixed* model from here on.)

- *Proportional Model.* Here the number of users, their queries and clicks vary from day to day, but the proportions of clicks for different keywords stay the same. For example, people click on ads for SHOES twice as much as they click on ads for DRESSES, although the actual number of clicks varies depending on the overall level of activity of online shoppers. This is modeled by having one global random variable that represents the total number of clicks in the day, as well as a fixed known multiplier for each keyword that represents that keyword's share of the clicks.

- *Independent Keywords Model.* Each keyword comes with its own probability distribution for the number of clicks, and the samples are drawn from these distributions independently. This is perhaps more realistic for keywords pertaining to different topics. For example, the number of clicks for a keyword related to MUSIC is independent of the number of clicks for a keyword related to SPORTS.

- *Scenario Model.* This model attempts to capture the full generality, but without the large number of bits needed to represent an arbitrary joint probability distribution. In this model, there is a number of *scenarios*, each of which specifies the exact number of clicks for each keyword. One scenario is sampled from a given probability distribution over scenarios, determining the numbers of clicks for the problem. For an

[3]The nature and number of queries vary significantly. An example in Google Trends shows the spikes in searches for shoes, flowers and chocolate: `http://www.google.com/trends?q=shoes,flowers,chocolate`.

[4]See for example the information provided to any AdWords advertiser. See also `https://adwords.google.com/support`.

arbitrarily large number of scenarios, this is a fully general model which can simulate any joint probability distribution. However, we typically think of this model with a limited (polynomial) number of scenarios. For example, the scenarios can represent different types of days with their unique click patterns, such as "weekend", "holiday" or "snowy day". They can also represent a set of samples from a more complex distribution, with the hope that a solution that works well for the majority of the samples would work well for the whole distribution.

There are two issues that arise in these stochastic models.

- STOCHASTIC EVALUATION PROBLEM (SE). Given a solution $\boldsymbol{b}$, can we evaluate $E[\mathbf{value}(\boldsymbol{b})]$ for any of the three models above? Even this is nontrivial as is typical in stochastic optimization problems. It is also of interest in solving the budget optimization problem below.

- STOCHASTIC BUDGET OPTIMIZATION PROBLEM (SBO). This is the Budget Optimization problem with one of the stochastic models above determining $\mathbf{clicks}_i$ for each i, with the objective to maximize

$$E[\mathbf{value}(\boldsymbol{b})] = E\left[\frac{\sum_{i\in T} b_i \mathbf{clicks}_i}{\max\left(1, \sum_{i\in T} b_i \mathbf{cost}_i/B\right)}\right]. \quad (3)$$

  The expectation is taken over the joint distribution of $\mathbf{clicks}_i$ for all $i \in T$.

1.3 Our results

We present algorithmic and complexity-theoretic results for the SE and SBO problems. We will state our results in their generality and they will depend on the description size of the probability distributions. Still, it should be clear that in practice one expects the probability distributions to be specified with a few points, and those are likely to be interesting, applicable cases.

For SE problems, our results are as follows. The problem is straight-forward to solve for the fixed and scenario models since the expression for the expected value of the objective can be explicitly written in polynomial time. For the proportional model, we give an exact algorithm to evaluate a solution in the case that the distribution for the total number of clicks does not have too many points in it, and we give a polynomial-time approximation scheme (PTAS) for the case that all points cannot be listed explicitly in polynomial time. For the independent model, the number of possibilities for different click quantities may be exponential in the number of keywords, in which case a polynomial-time algorithm is not able to enumerate all of them explicitly, and the problem of evaluating a solution is likely to be $\#P$-hard. We give a PTAS for evaluating a solution in the independent model. These results are used to derive algorithms for the SBO problem, though they may be of independent interest.

Our main results are for the SBO problem. In fact, all our algorithms produce a special kind of solutions called *prefix solutions*. Say the keywords are numbered in order of increasing $\mathbf{cpc}_i$. A prefix solution bids on some prefix of the list of keywords, i.e., on the cheap ones. Formally, an *integer prefix solution* with bids b_i has the property that there exists some i^* such that $b_i = 1$ for all $i \le i^*$, and $b_i = 0$ for $i > i^*$. For a *fractional prefix solution*, there exists an i^* such that $b_i = 1$ for $i < i^*$, $b_i = 0$ for $i > i^*$, and $b_{i^*} \in [0, 1]$. We show:

- For the proportional model, we can find an optimal fractional solution in polynomial time if the distribution of clicks can be described using polynomial number of points; else, we obtain a PTAS. We get this result by showing that the optimal fractional solution in this case is a prefix solution and using our evaluation algorithm.

- For the independent model which is more complex than the proportional case, we present an $2+\varepsilon$ approximation algorithm in polynomial time. We get this by showing the connection to integer prefix solution and using our PTAS for the evaluation algorithm. We also show that the best fractional prefix is not the optimal fractional solution for the independent model.

- For the scenario model, we show a negative result that finding the optimum, fractional or integer, is NP-hard. In this case, the best prefix solution is arbitrarily far from the optimum.

1.4 Related work

Together, our results represent a new theoretical study of stochastic versions of budget optimization problems in search-related advertising. The budget optimization problem was studied recently [4] in the fixed model, when $\mathbf{clicks}_i$'s are known. On one hand, our study is more general, with the emphasis on the uncertainty in modeling $\mathbf{clicks}_i$'s and the stochastic models we have formulated. We do not know of prior work in this area that formulates and uses our stochastic models. On the other hand, our study is less general as it does not consider the interaction between keywords that occurs when a user's search query matches two or more keywords, which was studied in [4]. There is a lot of work on stochastic versions of problems in optimization, such as facility location, Steiner trees, bin-packing and LP (see for example the survey [8]). In search-related advertising, a prior work [7] studied the *keyword selection* problem where the goal is to select a subset of keywords from a large pool for the advertiser to choose to bid. It is a special case of our budget optimization problem with the large pool as the input. The model used in [7] is similar to the proportional model here, but the underlying optimization problem in [7] is convex; in contrast, an intriguing and non-trivial aspect of our work is that in all three models, our optimization problem is non-convex: for three prefixes $i < j < k$, the objective function at j may be worse than the worst of i and k because of the budget scaling. In [7], the proportions of clicks for different keywords are unknown and an adaptive algorithm is developed that learns the proportions by bidding on different prefix solutions, and eventually converges to near-optimal profits [7], assuming that various parameters are concentrated around their means. Our work is different, as we consider algorithms that solve the problem in advance, and not by adaptive learning, and work for prespecified probability distributions, however arbitrary. There has been a lot of work on search-related auctions in the presence of budgets, but it has primarily focused on the game-theoretic aspects [6, 1], strategy-proof mechanisms [2, 3], and revenue maximization [5].

1.5 Map

We briefly discuss the fixed case first, and then discuss the three stochastic models in the following sections; in each case, we solve both evaluation and BO problems. In Sections 6 and 7, we present some extensions of our work, and state a few open problems.

2. FIXED MODEL

We first discuss the BO problem in the fixed model. A certain fractional prefix solution, which is easy to find, is optimal for this case. The algorithm is analogous to that for the fractional knapsack problem. We find the maximum index i^* such that $\sum_{i \leq i^*} \mathbf{cost}_i \leq B$. If i^* is the last index in T, we set $b_i = 1$ for all keywords i. Otherwise find a fraction $\alpha \in [0, 1)$ such that $\sum_{i \leq i^*} \mathbf{cost}_i + \alpha \cdot \mathbf{cost}_{i^*+1} = B$, and set $b_i = 1$ for $i \leq i^*$, $b_{i^*+1} = \alpha$, and $b_i = 0$ for $i > i^* + 1$. Call the resulting solution $\boldsymbol{b}$.

The fact that the optimal solution is a prefix solution can be shown by a simple interchange argument (we show this in a more general setting in Section 3.2). It remains to show that $\boldsymbol{b}$ is the best prefix solution. Notice that $\boldsymbol{b}$ is the maximal prefix solution whose cost does not exceed the budget. A different solution $\boldsymbol{b}' \neq \boldsymbol{b}$ with $\mathbf{cost}(\boldsymbol{b}') < B$ can clearly be improved by slightly increasing the bid on some keyword, as that increases the number of clicks while remaining within the budget. A prefix solution $\boldsymbol{b}'$ with $\mathbf{cost}(\boldsymbol{b}') > B$ can be improved by slightly decreasing the bid on its most expensive keyword: this decreases the average cost per click of the solution, thus increasing the value.

Theorem 2 *In the fixed model, the optimal fractional solution for the BO problem is the maximal prefix whose cost does not exceed the budget, which can be found in linear time.*

The integer version of this problem is NP-hard by reduction from KNAPSACK. The full proof is omitted here, but the idea is to construct an instance of the BO problem that has one special keyword 0 with large $\mathbf{clicks}_0$ and $\mathbf{cost}_0 = 0$, and the other keywords corresponding to the items in the KNAPSACK instance, with $\mathbf{clicks}_i$ being the value of the item, and $\mathbf{cost}_i$ being the size. The budget is set to the knapsack size. Then the optimal integer solution to this BO instance never exceeds the budget, and places bids on the subset of keywords corresponding to the optimal knapsack solution (in addition to keyword 0).

3. PROPORTIONAL MODEL

In the proportional model of SBO, we are given q_i, the *click frequency* for each keyword $i \in T$, with $\sum_{i \in T} q_i = 1$. The total number of clicks is denoted by a random variable C, and has a known probability distribution p. The number of clicks for a keyword i is then determined as $\mathbf{clicks}_i = q_i \cdot C$. For a specific value c of C, let $\mathbf{clicks}_i^c = q_i c$ and $\mathbf{cost}_i^c = \mathbf{cpc}_i \mathbf{clicks}_i^c$. The objective is to maximize the expected number of clicks, given by expression (3).

We show how to solve the evaluation problem efficiently in the proportional model, and then use it to find the optimal fractional solution to SBO, which, as we prove, is a prefix solution.

3.1 Evaluating a solution

Assuming that the distribution for C is given in such a way that it is easy to evaluate $\Pr[C > c^*]$ and $\sum_{c \leq c^*} c\, p(c)$ for any c^*, we show how to find $E[\mathbf{value}(\boldsymbol{b})]$ for any given solution $\boldsymbol{b}$ without explicitly going through all possible values of C and evaluating the objective function for each one.

The solution $\boldsymbol{b}$ may be under or over budget depending on the value of C. Define a threshold $c^* = B / \sum_{i \in T} b_i q_i \mathbf{cpc}_i$, so that for $c \leq c^*$, $\mathbf{cost}^c(\boldsymbol{b}) \leq B$, and for $c > c^*$, $\mathbf{cost}^c(\boldsymbol{b}) > B$. Notice that in the proportional model, $\mathbf{cpc}(\boldsymbol{b})$ is independent of C, as both $\mathbf{clicks}(\boldsymbol{b})$ and $\mathbf{cost}(\boldsymbol{b})$ are proportional to C. Then using expression (2) for $\mathbf{value}(\boldsymbol{b})$, the objective becomes easy to evaluate:

$$E[\mathbf{value}(\boldsymbol{b})] = \sum_{i \in T} b_i q_i \sum_{c \leq c^*} c\, p(c) + \frac{B}{\mathbf{cpc}(\boldsymbol{b})} \Pr[C > c^*]. \quad (4)$$

3.2 Prefix is the optimal solution

Theorem 3 *The optimal fractional solution for the SBO problem in the proportional model is a fractional prefix solution.*

PROOF. We use an interchange argument to show that any solution can be transformed into a prefix solution without decreasing its value. Consider a solution $\boldsymbol{b}$. If $\boldsymbol{b}$ is not a prefix solution, then there exist keywords i and j with $i < j$, $b_i < 1$, and $b_j > 0$. Choose the smallest such i and the largest such j. If $q_i \mathbf{cpc}_i = 0$, set $b_i = 1$ and continue. Otherwise pick the maximum $\delta_i, \delta_j > 0$ that satisfy

$$\delta_i \leq 1 - b_i, \qquad \delta_j \leq b_j, \qquad \delta_i = \frac{q_j \mathbf{cpc}_j}{q_i \mathbf{cpc}_i}\, \delta_j.$$

If we assign $b_i' = b_i + \delta_i$, $b_j' = b_j - \delta_j$, and $b_k' = b_k$ for $k \notin \{i, j\}$, then we get a solution $\boldsymbol{b}'$ such that for any c, $\mathbf{cost}^c(\boldsymbol{b}') = \mathbf{cost}^c(\boldsymbol{b})$ and $\mathbf{clicks}^c(\boldsymbol{b}') \geq \mathbf{clicks}^c(\boldsymbol{b})$:

$$\begin{aligned}
\mathbf{cost}^c(\boldsymbol{b}') - \mathbf{cost}^c(\boldsymbol{b}) &= c \cdot (q_i \mathbf{cpc}_i \delta_i - q_j \mathbf{cpc}_j \delta_j) = 0 \\
\mathbf{clicks}^c(\boldsymbol{b}') - \mathbf{clicks}^c(\boldsymbol{b}) &= c \cdot (q_i \delta_i - q_j \delta_j) \\
&= c \cdot q_j \left(\frac{\mathbf{cpc}_j}{\mathbf{cpc}_i} - 1\right) \delta_j \;\geq\; 0.
\end{aligned}$$

Since for any c, the value of the solution does not decrease, the expected value over C does not decrease either, $E[\mathbf{value}(\boldsymbol{b}')] \geq E[\mathbf{value}(\boldsymbol{b})]$. As a result of the transformation, either $b_i' = 1$ or $b_j' = 0$, so the process terminates after a finite number of steps, resulting in a prefix solution with expected value at least that of the original one. □

3.3 Finding the optimal prefix

It is nontrivial to find the best fractional prefix solution for the proportional case. First we mention two simple-minded approaches to this problem and give examples to show that they do not work. Then we show how to find the best prefix by listing all the *interesting* ones and evaluating them efficiently (notice that *all* the fractional prefixes cannot be listed).

One simple way to find a prefix in the proportional model is to convert it to a fixed case problem by setting the number of clicks for a keyword to its expectation, i.e. set $\mathbf{clicks}_i^* = E(\mathbf{clicks}_i) = q_i E(C)$, and then find the optimal prefix in this new instance as in Section 2. But the following example demonstrates that this is not optimal. There are two keywords, $\mathbf{cpc}_1 = 1$, $\mathbf{cpc}_2 = 5$, $q_1 = \frac{5}{6}$, $q_2 = \frac{1}{6}$. $C = 0$ with probability 90%, and $C = 60$ with probability 10%. So most of the time no clicks come, and sometimes we get 50 cheap clicks and 10 expensive ones. The best solution for budget $B = 10$ is to bid only on keyword 1, which gets 10 cheap clicks in the 10% case, or 1 click in expectation. However, using expectations to find a prefix, we get $\mathbf{clicks}^*_1 = 5$ and $\mathbf{clicks}^*_2 = 1$, and the best solution to that instance is to bid on both keywords. But in the original stochastic instance, this solution gets only 0.6 clicks in expectation.

Another possible approach is a greedy procedure that starts with the empty prefix, and keeps lengthening it while the solution improves. We show that this does not work either, because the expected value of the solution as a function of the length of the prefix can have multiple local maxima. Let $\alpha \in (0, 1)$ be a fixed fraction, and let $0 < \epsilon < \alpha$ be a quantity that approaches zero. There are three keywords, with $\mathbf{cpc}_1 = 0$, $\mathbf{cpc}_2 = \mathbf{cpc}_3 = 1$, $q_1 = \epsilon$, $q_2 = \alpha - \epsilon$, $q_3 = 1 - \alpha$; $B = 1$. $C = 1$ with probability $1 - \epsilon$, and $C = 1/\epsilon^2$ otherwise. If we compare the three integer prefixes, it turns out that the value of bidding on keyword 1 approaches 1 as $\epsilon \to 0$, the value of bidding on $\{1, 2, 3\}$ also approaches 1, but the value of the intermediate solution, bidding on keywords 1 and 2, approaches α, and therefore is smaller than the other two.

Our solution is as follows. If we were only interested in integer prefixes, we could evaluate all of them and find the best one in polynomial time. However, the fractional prefixes cannot be explicitly enumerated. But we show that it is sufficient to evaluate only a finite number of different prefixes in order to find the one with maximum value. The number of such points is $O(n + t)$, where t is the number of possible values that C can take. In the case that t is polynomial in n, the optimal prefix can be found in polynomial time. If not, then for a constant $\epsilon > 0$, the probability that C falls between successive powers of $(1+\epsilon)$ can be combined into buckets, resulting in a polynomial number of values for C. Using this approximate distribution yields a PTAS for finding the optimal prefix.

We *mark* some points in the space of possible prefixes. First, we mark all the integer prefixes. Then, for each value c of C that has non-zero probability, we mark the threshold prefix $\boldsymbol{b}$ that exactly spends the budget for $C = c$, i.e. such that $\mathbf{cost}^c(\boldsymbol{b}) = B$. This partitions the space of prefixes into intervals. Notice that for any two prefix solutions $\boldsymbol{b}$ and $\boldsymbol{b}'$ inside of the same interval I, the set of values of C that cause these solutions to exceed the budget is the same, i.e. $\{c \mid \mathbf{cost}^c(\boldsymbol{b}) > B\} = \{c \mid \mathbf{cost}^c(\boldsymbol{b}') > B\}$. Call this set $C_I^>$.

Now we show how to find the optimal prefix solution inside an interval defined by the marked points. Consider such an interval I, and suppose that all prefix solutions inside I bid $b_j = 1$ for $j < i$, $b_j = 0$ for $j > i$, and $b_i \in (b_1, b_2)$ for some $0 \le b_1 < b_2 \le 1$. Then the objective function for solutions in this interval becomes (analogously to equation (4))

$$\sum_{c \notin C_I^>} c\ p(c) \left(\sum_{j<i} q_j + b_i q_i \right) + \\ + B \Pr[C \in C_I^>] \cdot \frac{\sum_{j<i} q_j + b_i q_i}{\sum_{j<i} q_j \mathbf{cpc}_j + b_i q_i \mathbf{cpc}_i},$$

which we have to maximize over the possible values of the variable b_i. This can be done by taking the derivative of this expression with respect to b_i and setting it to zero, which has at most one solution on the interval (b_1, b_2). If this solution exists, we add it to a set of *interesting* points. To obtain the overall optimal solution, the algorithm evaluates all the prefixes defined by the *marked* and the *interesting* points.

Theorem 4 *The optimal fractional solution to SBO problem in the proportional model can be found exactly in time $O(n + t)$, where t is the number of possible values of C, or approximated by a PTAS.*

4. INDEPENDENT MODEL

In the independent model of SBO, the number of clicks for keyword $i \in T$, $\mathbf{clicks}_i$, has a probability distribution p_i (which can be different for different keywords). The key distinguishing feature of this model is that for $i \neq j$, the variables $\mathbf{clicks}_i$ and $\mathbf{clicks}_j$ are independent.

The independent model is more complex than the ones discussed so far. Theorem 5 shows that prefix solution may not be optimal even among fractional solutions in the independent model. However, in Section 4.1 we prove that some prefix solution is a 2-approximate integer solution. But finding this best prefix requires the ability to evaluate a given solution, which in this model is likely to be $\#P$-hard. So in Section 4.2, we give a PTAS for evaluating any proposed solution. Combined, these two results imply a $(2 + \epsilon)$-approximation algorithm for the BO problem in the independent model.

Theorem 5 *In the independent model of the SBO problem, the optimal fractional solution may not be a prefix solution.*

PROOF. We give an example with three keywords in which the optimal solution bids on the first and third keywords, and gets more clicks than any (even fractional) prefix solution. The idea is that the second and third keywords cost about the same, but the third one is better because it always comes in the same quantity, whereas the second one has high variance. Let $\mathbf{cpc}_1 = 0$, $\mathbf{cpc}_2 = 1$, $\mathbf{cpc}_3 = 1$; $\mathbf{clicks}_1 = 1$ with probability 1, $\mathbf{clicks}_2 = (0$ *or* $1)$ with probability $\frac{1}{2}$ each, and $\mathbf{clicks}_3 = 1$ with probability 1; $B = 1$. The optimal solution is $b_1 = b_3 = 1$ and $b_2 = 0$, which always gets 2 clicks. The best prefix solution is $b_1 = b_2 = 1$ and $b_3 = 0$, which gets 1 or 2 clicks with probability $\frac{1}{2}$ each, or only 1.5 clicks in expectation. The example can be modified so that the third keyword is strictly more expensive than the second one. $\square$

4.1 Prefix is a 2-approximation

In this section we show that for any instance of the SBO problem in the independent model, there exists an integer prefix solution whose expected value is at least half of the optimum. In particular, we show that any integer solution $\boldsymbol{b}$ can be transformed into a prefix solution $\boldsymbol{b}_V$ without losing more than half of its value. Let $S = \{i \mid b_i = 1\}$ be the set of keywords that $\boldsymbol{b}$ bids on.

Let σ be the event that clicks for each keyword $i \in T$ come in quantity $\mathbf{clicks}^\sigma(i)$. Then its probability is

$$p(\sigma) = \prod_{i \in T} p_i(\mathbf{clicks}^\sigma(i)).$$

Define $\mathbf{clicks}^\sigma(\boldsymbol{b}) = \sum_{i \in S} \mathbf{clicks}^\sigma(i)$, $\mathbf{cost}^\sigma(i) = \mathbf{cpc}_i \cdot \mathbf{clicks}^\sigma(i)$, and $\mathbf{cost}^\sigma(\boldsymbol{b}) = \sum_{i \in S} \mathbf{cost}^\sigma(i)$. The effective number of clicks (after taking the budget into account) that solution $\boldsymbol{b}$ gets from keyword i in the event σ is

$$\overline{\textbf{clicks}}^{\sigma}_{S}(i) = \frac{\textbf{clicks}^{\sigma}(i)}{\max(1, \textbf{cost}^{\sigma}(\boldsymbol{b})/B)},$$

and the total effective number of clicks is $\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}) = \sum_{i\in S} \overline{\textbf{clicks}}^{\sigma}_{S}(i)$. Then $E[\textbf{value}(\boldsymbol{b})] = \sum_{\sigma} p(\sigma)\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b})$.

Let $i^*(\boldsymbol{b})$ be the minimum index i^* such that

$$\sum_{\sigma} p(\sigma) \sum_{i\in S, i\le i^*} \overline{\textbf{clicks}}^{\sigma}_{S}(i) \ge \frac{1}{2}\ E[\textbf{value}(\boldsymbol{b})].$$

Theorem 6 *For any integer solution $\boldsymbol{b}$ to the SBO problem with independent keywords, there exists an integer prefix solution $\boldsymbol{b}_V$ such that $E[\textbf{value}(\boldsymbol{b}_V)] \ge \frac{1}{2}\ E[\textbf{value}(\boldsymbol{b})]$. In particular, the solution $\boldsymbol{b}_V$ bidding on the set $V = \{i \mid i \le i^*(\boldsymbol{b})\}$ has this property.*

The idea of the proof will be to think of the above prefix solution as being obtained in two steps from the original solution $\boldsymbol{b}$. First, we truncate $\boldsymbol{b}$ by discarding all keywords after i^*. Then we fill in the gaps in the resulting solution in order to make it into a prefix. To analyze the result, we first show that all keywords up to i^* are relatively cheap, and that the truncated solution (called $\boldsymbol{b}_U$) retains at least half the value of the original one (Claim 7). Then we show that filling in the gaps preserves this guarantee. Intuitively, two good things may happen: either clicks for the new keywords don't come, in which case we get all the clicks we had before; or they come in large quantity, spending the budget, which is good because they are cheap. Lemma 8 analyzes what happens if new clicks spend α^{σ} fraction of the budget.

Let $i^* = i^*(\boldsymbol{b})$. To analyze our proposed prefix solution $\boldsymbol{b}_V$, we break the set V into two disjoint sets U and N. $U = V \cap S = \{i \le i^* \mid i \in S\}$ is the set of cheapest keywords that get half the clicks of $\boldsymbol{b}$. The new set $N = V \setminus S = \{i \le i^* \mid i \notin S\}$ fills in the gaps in U. Let $\boldsymbol{b}_U$ and $\boldsymbol{b}_N$ be the solutions that bid on keywords in U and N respectively.

Define the average cost per click of solution $\boldsymbol{b}$ as

$$\textbf{cpc}^* = \frac{\sum_{\sigma} p(\sigma)\ \sum_{i\in S} \textbf{cpc}_i\ \overline{\textbf{clicks}}^{\sigma}_{S}(i)}{\sum_{\sigma} p(\sigma)\ \overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b})},$$

where the numerator is the average amount of money spent by $\boldsymbol{b}$, and the denominator is the average number of clicks obtained. A useful fact to notice is that since the numerator of this expression never exceeds the budget, and the denominator is equal to $E[\textbf{value}(\boldsymbol{b})]$, we have that

$$E[\textbf{value}(\boldsymbol{b})] \le \frac{B}{\textbf{cpc}^*}. \tag{5}$$

We make some observations about $\boldsymbol{b}_U$ and i^*.

Claim 7 $E[\textbf{value}(\boldsymbol{b}_U)] \ge \frac{1}{2}E[\textbf{value}(\boldsymbol{b})]$ *and* $\textbf{cpc}_{i^*} \le 2\textbf{cpc}^*$.

PROOF. Since $U \subseteq S$, for all σ, $cost^{\sigma}(\boldsymbol{b}_U) \le cost^{\sigma}(\boldsymbol{b})$, which implies that $\overline{clicks}^{\sigma}_{U}(i) \ge \overline{clicks}^{\sigma}_{S}(i)$ for any $i \in U$. So $E[\textbf{value}(\boldsymbol{b}_U)] =$

$$\sum_{\sigma} p(\sigma) \sum_{i\in U} \overline{clicks}^{\sigma}_{U}(i) \ge \sum_{\sigma} p(\sigma) \sum_{i\in U} \overline{clicks}^{\sigma}_{S}(i)$$

$\ge \frac{1}{2}\ E[\textbf{value}(\boldsymbol{b})]$ by definitions of U and i^*.

The second part follows by Markov's inequality:

$$\textbf{cpc}^* \ge \frac{\sum_{\sigma} p(\sigma)\ \sum_{i\in S, i\ge i^*} \textbf{cpc}_{i^*}\ \overline{\textbf{clicks}}^{\sigma}_{S}(i)}{\sum_{\sigma} p(\sigma)\ \overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b})} \ge \textbf{cpc}_{i^*} \cdot \frac{1}{2},$$

where the second inequality is by minimality of i^*. $\square$

Lemma 8 *For any σ, let $\alpha^{\sigma} = \frac{\min(B,\ \textbf{cost}^{\sigma}(\boldsymbol{b}_N))}{B}$. Then*

$$\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_V)\ \ge\ \alpha^{\sigma}\frac{B}{2\textbf{cpc}^*} + (1-\alpha^{\sigma})\ \overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_U).$$

PROOF. If $\textbf{cost}^{\sigma}(\boldsymbol{b}_N) \ge B$, then $\textbf{cost}^{\sigma}(\boldsymbol{b}_V) \ge B$, so

$$\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_V) = \frac{\sum_{i\in V} \textbf{clicks}^{\sigma}(i)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)/B} =$$

$$= B \cdot \frac{\sum_{i\in V} \textbf{clicks}^{\sigma}(i)}{\sum_{i\in V} \textbf{cpc}_i \textbf{clicks}^{\sigma}(i)} \ge \frac{B}{\textbf{cpc}_{i^*}} \ge \frac{B}{2\textbf{cpc}^*},$$

which proves the lemma for the case of $\alpha^{\sigma} = 1$. Intuitively, in this case the whole budget is spent, and since all keywords in V cost at most $2\textbf{cpc}^*$ (by Claim 7), V gets at least $\frac{B}{2\textbf{cpc}^*}$ clicks. For the rest of the proof assume that $\textbf{cost}^{\sigma}(\boldsymbol{b}_N) < B$. Then $\alpha^{\sigma} = \frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_N)}{B} < 1$.

Another simple case is $\textbf{cost}^{\sigma}(\boldsymbol{b}_V) \le B$. Then the budget is not reached and V collects all the clicks from U and N:

$$\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_V) = \textbf{clicks}^{\sigma}(\boldsymbol{b}_N) + \textbf{clicks}^{\sigma}(\boldsymbol{b}_U) \ge$$
$$\frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_N)}{2\textbf{cpc}^*} + \overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_U) \ge \frac{\alpha^{\sigma} B}{2\textbf{cpc}^*} + (1-\alpha^{\sigma})\ \overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_U).$$

Now consider the case when $\textbf{cost}^{\sigma}(\boldsymbol{b}_N) + \textbf{cost}^{\sigma}(\boldsymbol{b}_U) > B$. Here at most α^{σ} fraction of the budget is used for the new keywords from N, which cost at most $2\textbf{cpc}^*$ per click, and the remaining budget is able to buy $(1-\alpha^{\sigma})$ fraction of the clicks that $\boldsymbol{b}_U$ was getting.

Define $\textbf{cpc}^{\sigma}(\boldsymbol{b}_U) = \frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_U)}{\textbf{clicks}^{\sigma}(\boldsymbol{b}_U)}$, and similarly for N. Then

$$\frac{B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_U)} = \frac{\textbf{clicks}^{\sigma}(\boldsymbol{b}_U)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_U)/B} \ge \overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_U). \tag{6}$$

Now $\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_V)$

$$\begin{aligned}
&= \frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_U)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)} \cdot \frac{B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_U)} + \frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_N)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)} \cdot \frac{B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_N)} \\
&= \frac{(1-\alpha)B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_U)} + \left[\frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_U)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)} - (1-\alpha)\right] \frac{B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_U)} \\
&\quad + \frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_N)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)} \cdot \frac{B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_N)} \\
&\ge \frac{(1-\alpha)B}{\textbf{cpc}^{\sigma}(\boldsymbol{b}_U)} + \left[\frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_U)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)} - (1-\alpha)\right] \frac{B}{2\textbf{cpc}^*} \\
&\quad + \frac{\textbf{cost}^{\sigma}(\boldsymbol{b}_N)}{\textbf{cost}^{\sigma}(\boldsymbol{b}_V)} \cdot \frac{B}{2\textbf{cpc}^*} \\
&\ge (1-\alpha)\overline{\textbf{clicks}}^{\sigma}(\boldsymbol{b}_U) + \alpha\frac{B}{2\textbf{cpc}^*}
\end{aligned}$$

where the first inequality follows because $\textbf{cpc}^{\sigma}(\boldsymbol{b}_U) \le 2\textbf{cpc}^*$, $\textbf{cpc}^{\sigma}(\boldsymbol{b}_N) \le 2\textbf{cpc}^*$, and the quantity in square brackets is non-negative. The second inequality follows from (6). $\square$

PROOF OF THEOREM 6. We now use the above results to prove the theorem. Let σ_U be the event that clicks for each keyword $i \in U$ come in quantity $\textbf{clicks}^{\sigma_U}(i)$. Then its probability is

$$p(\sigma_U) = \prod_{i\in U} p_i(\textbf{clicks}^{\sigma_U}(i)).$$

Here the independence of keywords becomes crucial. In particular, what we need is that the number of clicks that come for keywords in U is independent of the number of clicks for keywords in N. So the probability of σ_V is the product of $p(\sigma_U)$ and $p(\sigma_N)$, where σ_V is the event that both σ_U and σ_N happen. Notice that α^σ of Lemma 8 depends only on keywords in N, and is independent of what happens with keywords in U. So here we call it α^{σ_N}. We have

$$\begin{aligned}
E[\mathbf{value}(\boldsymbol{b}_V)] &= \sum_{\sigma_V} p(\sigma_V)\overline{\mathbf{clicks}}^{\sigma_V}(\boldsymbol{b}_V) \\
&\geq \sum_{\sigma_N}\sum_{\sigma_U} p(\sigma_N)p(\sigma_U)\left[\frac{\alpha^{\sigma_N}B}{2\mathbf{cpc}^*} + (1-\alpha^{\sigma_N})\,\overline{\mathbf{clicks}}^{\sigma_V}(\boldsymbol{b}_U)\right] \\
&= \sum_{\sigma_N} p(\sigma_N)\left[\frac{\alpha^{\sigma_N}B}{2\mathbf{cpc}^*} + (1-\alpha^{\sigma_N})\sum_{\sigma_U} p(\sigma_U)\overline{\mathbf{clicks}}^{\sigma_V}(\boldsymbol{b}_U)\right] \\
&\geq \sum_{\sigma_N} p(\sigma_N)\,\frac{1}{2}E[\mathbf{value}(\boldsymbol{b})] \;=\; \frac{1}{2}E[\mathbf{value}(\boldsymbol{b})],
\end{aligned}$$

bounding both $\frac{B}{2\mathbf{cpc}^*}$ and $E[\mathbf{value}(\boldsymbol{b}_U)]$ by $\frac{1}{2}E[\mathbf{value}(\boldsymbol{b})]$ using inequality (5) and Claim 7. $\square$

4.2 Evaluating a solution in independent model

In this section we present a PTAS for the SE problem in the independent model. We are given an instance of the SBO problem, and an (integer or fractional) solution $\boldsymbol{b}$. For a keyword $i \in T$, let $C_i = \{c \mid p_i(c) > 0\}$ be the set of values that $\mathbf{clicks}_i$ can take. For now we assume that $\sum_i |C_i|$ is polynomial in the size of the input, and later show how to remove this assumption. Let $\mathbf{cost}(\boldsymbol{b}_{-i}) = \sum_{j\neq i} b_j \mathbf{cpc}_j \mathbf{clicks}_j$ be the cost of clicks for all keywords except i. By some algebraic manipulation, one can show the following.

Claim 9 *$E[\mathbf{value}(\boldsymbol{b})]$ is equal to*

$$\sum_{i\in T}\sum_{c\in C_i} p_i(c)\; b_i c \sum_{d\geq 0} \frac{1}{f_i(c,d)} \Pr[\mathbf{cost}(\boldsymbol{b}_{-i}) = d],$$

where $f_i(c,d) = \max(1, \frac{d + c\cdot \mathbf{cpc}_i}{B})$.

In this expression, $b_i c$ is the number of clicks from keyword i, and the expression in the third sum is the amount by which this number should be scaled because of the budget. The variable d represents the cost of all keywords other than i.

As a result, the problem of finding $E[\mathbf{value}(\boldsymbol{b})]$ reduces to evaluating, for any given i and c, the expression

$$s(i,c) = \sum_{d\geq 0} \frac{1}{f_i(c,d)} \Pr[\mathbf{cost}(\boldsymbol{b}_{-i}) = d]. \tag{7}$$

Lemma 10 *For any given $\epsilon > 0$, there is a polynomial-time algorithm that finds a value s' such that $s(i,c) \leq s' \leq (1+\epsilon)\; s(i,c)$.*

Proof. We build a dynamic programming table that represents an estimate of $\Pr[\mathbf{cost}(\boldsymbol{b}_{-i}) = d]$ as a function of d. Fix an ordering of elements in $T - \{i\}$ and construct a table P indexed by j and d, where $P(j,d)$ is the probability that the total cost of the first j elements is d. To make sure the table is of polynomial size, scale the costs so that the minimum non-zero value of $\mathbf{cost}_j$ for any j is 1, and restrict the possible values of d to 0 and $(1+\frac{\epsilon}{n})^k$ for non-negative integers k. If we let $M = \sum_{i\in T}\max\{c\cdot \mathbf{cpc}_i \mid c \in C_i\}$ be the maximum possible cost of all the clicks, then the number of values of d in the table is at most $\log_{1+\epsilon/n} M = O(\frac{n}{\epsilon}\log M)$, which is polynomial in the size of the input.

The table is initialized with $P(0,0) = 1$ and other entries equal to zero. Then for each keyword $j \in T - \{i\}$, each possible number of clicks $c \in C_j$, and each entry $P(j-1,d)$ in the previous row, we update $P(j, \lfloor d + c\cdot cpc_j\rfloor) = P(j, \lfloor d + c\cdot cpc_j\rfloor) + p_j(c)\cdot P(j-1,d)$. Here the operator $\lfloor\ \rfloor$ represents rounding down to the next available value of d. After filling the table, the algorithm outputs the value of expression (7) as determined by probabilities in the last row of the table.

To bound the error incurred by rounding down the costs, we consider an event σ that specifies a number of clicks $c_j \in C_j$ for each $j \in T - \{i\}$, and has probability $p(\sigma) = \prod_j p_j(c_j)$. Expression (7) can be rewritten as

$$s(i,c) = \sum_{\sigma} p(\sigma)\cdot \frac{1}{f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i}))} \tag{8}$$

As a result of a series of updates, the probability of σ contributes to some entry of the last row of P, say to the one with $d^\sigma = (1+\frac{\epsilon}{n})^k$. This d^σ is an estimate of the value of $\mathbf{cost}^\sigma(\boldsymbol{b}_{-i}) = \sum_{j\neq i} b_j \mathbf{cpc}_j c_j$. Since we only rounded down, we have $d^\sigma \leq \mathbf{cost}^\sigma(\boldsymbol{b}_{-i})$. Now note that since the intervals between successive powers of $(1+\frac{\epsilon}{n})$ are increasing, the biggest amount that we could have lost during any one rounding is $(1+\frac{\epsilon}{n})^{k+1} - (1+\frac{\epsilon}{n})^k = \frac{\epsilon}{n}\,(1+\frac{\epsilon}{n})^k = \frac{\epsilon}{n}\cdot d^\sigma$. Since we performed the rounding during at most n updates relevant to σ, we have that the true value $\mathbf{cost}^\sigma(\boldsymbol{b}_{-i}) \leq d^\sigma + n\cdot\frac{\epsilon}{n}\cdot d^\sigma = (1+\epsilon)d^\sigma$. So we have that the estimated cost d^σ for the event σ is $\frac{\mathbf{cost}^\sigma(\boldsymbol{b}_{-i})}{1+\epsilon} \leq d^\sigma \leq \mathbf{cost}^\sigma(\boldsymbol{b}_{-i})$.

Now the only thing left to do in order to show that the algorithm evaluates expression (7) accurately is to take into account $f_i(c,d)$. By monotonicity of f_i, we have

$$f_i\left(c, \frac{\mathbf{cost}^\sigma(\boldsymbol{b}_{-i})}{1+\epsilon}\right) \leq f_i(c, d^\sigma) \leq f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i})).$$

But notice that

$$f_i\left(c, \frac{\mathbf{cost}^\sigma(\boldsymbol{b}_{-i})}{1+\epsilon}\right) \geq \frac{f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i}))}{1+\epsilon}.$$

So we have that

$$\frac{f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i}))}{1+\epsilon} \leq f_i(c, d^\sigma) \leq f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i})),$$

and therefore

$$\frac{1}{f_i(c,d^\sigma)} \in \left[\frac{1}{f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i}))}, \frac{(1+\epsilon)}{f_i(c, \mathbf{cost}^\sigma(\boldsymbol{b}_{-i}))}\right].$$

So evaluating expression (8) using entries from the dynamic programming table instead of the true costs and probabilities incurs a multiplicative error of at most $(1+\epsilon)$. $\square$

If the input distributions p_i are represented implicitly, such that $\sum_i |C_i|$ is not polynomial in the input size, then we first convert them into distributions with polynomial number of points by combining the probability mass between successive powers of $(1+\epsilon')$ into buckets (rounding down). Then we run the above algorithm for discrete distributions so as to obtain a $(1+\epsilon')$-approximation for the rounded instance. This will be a $(1+\epsilon')^2$-approximation for the original instance, so if ϵ' is chosen such that $(1+\epsilon')^2 \leq (1+\epsilon)$, we obtain the desired $(1+\epsilon)$-approximation.

This result combined with Theorem 6 gives a simple algorithm for finding a $(2+\epsilon)$-approximate solution: evaluate each integer prefix using the PTAS and output the one with maximum value.

Theorem 11 *There is a $(2+\epsilon)$-approximation algorithm for the SBO problem in the independent model, which runs in time polynomial in n, $\frac{1}{\epsilon}$, and $\log M$.*

5. SCENARIO MODEL

In the scenario model, we are given T, B and costs $\mathbf{cpc}_i$ as usual. The numbers of clicks are determined by a set of scenarios Σ and a probability distribution p over it, so that a scenario $\sigma \in \Sigma$ materializes with probability $p(\sigma)$, in which case each keyword i gets $\mathbf{clicks}_i^\sigma$ clicks. The scenarios are disjoint and $\sum_{\sigma\in\Sigma} p(\sigma) = 1$. The reason this model does not capture the full generality of arbitrary distributions is that we assume that the number of scenarios, $|\Sigma|$, is relatively small, in the sense that algorithms are allowed to run in time polynomial in $|\Sigma|$. On the other hand, if, for example, we express the independent model in terms of scenarios, their number would be exponential in the number of keywords.

The evaluation of a given solution in the scenario model does not present a problem, as it can be done explicitly in time polynomial in $|\Sigma|$, by evaluating each scenario and taking the expectation. Nevertheless, this is the most difficult model for the SBO that we consider. We show two negative results.

Theorem 12 *The SBO problem is NP-hard in the scenario model.*

The proof shows, by reduction from CLIQUE, that it's NP-hard to find either an integer or a fractional solution to this problem, and appears in the Appendix.

Theorem 13 *The gap between the optimal fractional prefix solution and the optimal (integer or fractional) solution to the SBO problem in the scenario model can be arbitrarily large.*

PROOF. We give an example in which the ratio between the value of the optimal solution and the value of any prefix solution can be arbitrarily large. The example contains n scenarios and $2n$ keywords, numbered 1 through $2n$. The cost per click of keywords increases exponentially, with $\mathbf{cpc}_i = c^i$, for some constant $c > 1$. There is a budget $B > 0$. Say that the n scenarios are numbered $\sigma = 1$ to n. In scenario σ, only keywords $2\sigma - 1$ and 2σ receive clicks, and they receive $B/c^{2\sigma-1}$ clicks each. The probabilities of scenarios increase exponentially, and they are equal to $\alpha c^{2\sigma-1}$ for scenario σ (α is chosen to make the probabilities sum to 1). The idea here is that in each scenario, there are two types of clicks, cheap and expensive (clicks for the even-numbered keywords are c times more expensive than for their preceding odd-numbered keywords), and there are enough cheap clicks to spend the whole budget. So for a particular scenario σ, the best thing to do is to bid only on the cheap keyword $2\sigma - 1$, which gets $B/c^{2\sigma-1}$ clicks. Bidding on both keywords exceeds the budget and decreases the number of clicks to $\frac{2}{c+1}(B/c^{2\sigma-1})$. Since the sets of keywords that receive clicks in different scenarios are disjoint, the optimal solution overall (which happens to be integer) is to bid on all the odd-numbered keywords, but not the even-numbered ones. This gets the maximum number of clicks for each scenario individually, and therefore gets the maximum number of clicks in expectation. The expected number of clicks for the optimal solution is therefore

$$\sum_{\sigma} \alpha c^{2\sigma-1} \cdot \frac{B}{c^{2\sigma-1}} = n\alpha B.$$

Now consider some prefix solution for this example, either integer or fractional, and the keyword i^* such that $b_i = 1$ for $i < i^*$ and $b_i = 0$ for $i > i^*$. Let $\sigma^* = \lceil i^*/2 \rceil$ be the scenario containing clicks for keyword i^*. Intuitively, the prefix solution ruins the scenarios numbered less than σ^* because it bids for both keywords in them, and it ruins the scenarios numbered greater than σ^* because it does not bid at all for the keywords in them. As a result, a prefix solution can do well in at most one scenario. It gets the small number of clicks, $\frac{2}{c+1}(B/c^{2\sigma-1})$, for scenarios $\sigma < \sigma^*$, and it gets 0 clicks for scenarios $\sigma > \sigma^*$. It may get up to $B/c^{2\sigma^*-1}$ clicks for σ^*. So the value of a prefix solution is at most

$$\begin{aligned} &\sum_{\sigma<\sigma^*} \alpha c^{2\sigma-1} \cdot \frac{2}{(c+1)} \frac{B}{c^{2\sigma-1}} + \alpha c^{2\sigma^*-1} \cdot \frac{B}{c^{2\sigma^*-1}} \\ &\leq\; n\alpha B \left(\frac{2}{c+1} + \frac{1}{n} \right), \end{aligned}$$

which can be arbitrarily far from $OPT = n\alpha B$ as c and n increase. □

6. EXTENSIONS

We briefly describe a few extensions of our work.

6.1 Click values

Here we show that our results easily generalize to the case when the clicks from different keywords have different values to the advertiser. For example, a weight associated with a keyword might represent the probability that a user clicking on the ad for that keyword will make a purchase.

For each keyword i, we are given a weight w_i which is the value of a click associated with this keyword, and we would like to maximize the weighted number of clicks obtained:

$$E[\mathbf{value}(\boldsymbol{b})] = E\left[\frac{\sum_{i\in T} b_i w_i \mathbf{clicks}_i}{\max\left(1, \sum_{i\in T} b_i \mathbf{cpc}_i \mathbf{clicks}_i / B\right)} \right].$$

Obviously, the keywords with $w_i = 0$ can be just discarded. Now we make a substitution of variables, defining $\mathbf{clicks}'_i = w_i \mathbf{clicks}_i$ and $\mathbf{cpc}'_i = \mathbf{cpc}_i / w_i$. Substituting them into the objective function,

$$E[\mathbf{value}(\boldsymbol{b})] = E\left[\frac{\sum_{i\in T} b_i \mathbf{clicks}'_i}{\max\left(1, \sum_{i\in T} b_i \mathbf{cpc}'_i \mathbf{clicks}'_i / B\right)} \right],$$

we see that the problem reduces to the original unweighted SBO instance, with different keyword parameters. The proportional, independent, and scenario models of click arrival maintain their properties under this transformation, only some of the distributions for the numbers of clicks have to be scaled.

6.2 Extension to multiple-slot auctions

We now extend our results to the case when there are multiple slots, and in particular, we assume the Generalized

Second Price (GSP) auction currently used by search-related advertising engines. When advertising slots are allocated by a second-price auction with multiple slots, the bid amount for a keyword determines the position of the corresponding ads, which affects the number of clicks obtained for this keyword and the cost per click of these clicks. When a user clicks on the ad in slot i, the advertiser at slot i is charged the bid amount of the advertiser at slot $i+1$.[5]

Let us first focus on a particular keyword i and an auction in which it participates. If the auction has k available slots, then there are k threshold bid amounts, $bid_1 \geq bid_2 \geq ... \geq bid_k$, such that bidding any amount in the interval $[bid_j, bid_{j-1})$ places the ad in slot j, which has a probability of a click (clickthrough rate) $\mathbf{ctr}_i^j$ and a cost per click $\mathbf{cpc}_i^j$. Since we are considering the GSP auction, the cost per click is not affected by the exact bid amount, as long as it is in the specified interval. Both $\mathbf{ctr}_i^j$ and $\mathbf{cpc}_i^j$ are monotone non-decreasing step functions of the bid amount. To better represent the options for bidding on keyword i, we visualize the k possible pairs of $(\mathbf{cpc}_i^j \mathbf{ctr}_i^j,\ \mathbf{ctr}_i^j)$ values on a "plot" called a *landscape*. Notice that when both the axes are scaled by the number of queries, then it becomes a plot of $\mathbf{clicks}_i$ vs. $\mathbf{cost}_i$, with points for different options of how to bid (a more detailed description of landscapes appears in [4]). Landscapes for multiple auctions for the same keyword can be combined to obtain an *aggregate* landscape.

Some of our results extend to the model with such aggregate landscapes. Roughly speaking, a keyword with a landscape can be viewed as a list of several simple individual keywords with an additional constraint that a solution has to bid on some prefix of this list. For the fixed and proportional models, the optimal solutions without landscapes are prefix solutions anyway (by Theorems 2 and 3), so if we solve the problem as in the one-slot case, the solution will automatically satisfy the prefix constraint for keywords with landscapes, which means that it will also be optimal for the multiple-slot problem. In the independent model, however, the approximation ratio of 2 for the prefix solutions (Theorem 6), that we prove for the one-slot case, does not extend to the case of landscapes. This is because some of the "keywords" are no longer independent, but are actually the different bidding options for the same keyword. In fact, a prefix solution can be arbitrarily bad compared to the optimal solution, by an example that is very similar to one in Theorem 13. The only difference is that, instead of keywords $2i-1$ and $2i$ being coupled by occurring in the same scenario, they are coupled by representing the landscape of the same keyword. The negative results (Theorems 12 and 13) about the scenario model of course still hold for the more general case of multiple-slot auctions.

7. CONCLUDING REMARKS

We have initiated the study of stochastic version of budget optimization with the three models. We obtained upper bounds via prefix bids and showed hardness results for other cases. A lot remains to be done, both technically and conceptually. Technically, we need to extend the results to the case when there are interactions between keywords, that is, two or more of them apply to a user query and some resolution is needed. Also, we need to study online algorithms, including online budget optimization. Further, we would like to obtain some positive approximation results on the scenario model which seems quite intriguing from an application point of view. The conceptual challenge is one of modeling. Are there other suitable stochastic models for search-related advertising, that are both expressive, physically realistic and computationally feasible?

[5] There are some details, e.g. the cost is typically a small amount more than the bid of the advertiser at slot $i+1$.

8. REFERENCES

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

[2] C. Borgs, J. Chayes, N. Immorlica, M. Mahdian, and A. Saberi. Multi-Unit Auctions with Budget-Constrained Bidders. *ACM Conference on Electronic Commerce* (EC) 2005.

[3] C. Borgs, J. Chayes, O. Etesami, N. Immorlica, K. Jain and M. Mahdian. Bid optimization in online advertisement auctions. *Manuscript*, `http://www.bus.ualberta.ca/kasdemir/ssa2/borgsetal.pdf`, 2006.

[4] J. Feldman, S. Muthukrishnan, M. Pal and C. Stein. Budget optimization in search-based advertising auctions. *ACM Conference on Electronic Commerce* (EC) 2007.

[5] A. Mehta, A. Saberi, U. Vazirani and V. Vazirani. AdWords and Generalized On-line Matching. *IEEE FOCS*, 2005: 264-273

[6] M. Ostrovsky, B. Edelman and M. Schwarz. Internet Advertising and the Generalized Second Price Auction: Selling Billions of Dollars Worth of Keywords. September 2005, Revised August 2006. *American Economic Review*, forthcoming.

[7] P. Rusmevichientong and D. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. *ACM Conference on Electronic Commerce* (EC) 2006: 260-269.

[8] C. Swamy and D. Shmoys. Approximation Algorithms for 2-Stage Stochastic Optimization Problems. Survey article in *ACM SIGACT News*, 37(1):33-46, 2006.

9. APPENDIX

9.1 NP-hardness of scenario model: Proof of Theorem 12

We show that finding the optimal solution, either integer or fractional, to the SBO problem in the scenario model is NP-hard.

The reduction is from CLIQUE. We are given an instance of the CLIQUE problem with a graph G containing n nodes and m edges, and a desired clique size k. We use G and k to construct an instance I of SBO problem and a number V such that there is a solution to I with expected value of at least V if and only if G contains a clique of size k.

To specify I, let us construct a new bipartite graph $H = (L \cup R, E')$ whose right side R contains n nodes corresponding to nodes of G, and whose left side L contains m nodes corresponding to edges of G. A node in L corresponding to an edge (u, v) is connected to nodes in R that correspond to its endpoints u and v. We first describe the idea of the construction, and later show how to set the parameters to make it work. There will be three parameters, small positive values ϵ and δ, and a large value t.

All nodes of H are keywords, expensive ones on the left, with $\mathbf{cpc}_i = 1$ for $i \in L$, and cheap ones on the right, with $\mathbf{cpc}_i = \epsilon$ for $i \in R$. The budget is $K = \binom{k}{2}$. The goal will be to force a solution to select K nodes from L that are incident to at most k nodes in R, which corresponds to finding a set of k nodes with K edges in G, i.e. a k-clique. The scenarios in the SBO problem are as follows. There is a high-probability scenario σ_0 in which one click comes for each keyword in L. This scenario is sufficiently likely (occurs with probability $1-\delta$) that any integer solution to SBO has to bid for at least K of these keywords. Notice that since K is the budget, it does not make sense to bid on any more than K keywords. In addition to σ_0, there are n scenarios $\sigma_1 \ldots \sigma_n$, each occurring with probability δ/n. Scenario σ_i contains K/ϵ clicks for keyword $i \in R$ and a large number t of clicks for each of i's neighbors from L.

We now explain the intuition for why there is a good integrally-bidding solution for our SBO instance if and only if the graph G contains a k-clique. By the way we constructed the low-probability scenarios, if a solution does not bid on any neighbors of $i \in R$, then in scenario σ_i it would spend its whole budget on K/ϵ cheap clicks at cost ϵ each, thus obtaining many clicks. However, if it bids on any neighbors of i, then most of the budget will be spent on the expensive clicks from L, resulting in few clicks overall. So bidding on a keyword l from L effectively ruins the scenarios containing l's neighbors in R. Recall that the high probability of scenario σ_0 forces any good integral solution to bid on exactly K keywords from L. As a result, if G contains a k-clique, then it is possible to select K keywords corresponding to edges of G that ruin only k scenarios corresponding to nodes of G. However, if there is no k-clique, then bidding on any K keywords on the left ruins at least $k+1$ scenarios, thus producing a solution with a lower value.

We now show how to set the parameters of the construction and prove that the reduction works even if fractional bidding on keywords is allowed. First, assume that G contains a k-clique. Then a solution $\boldsymbol{b}$ to I, with $b_i = 1$ for all $i \in R$ and $b_i = 1$ for the K keywords in L that correspond to edges of the clique, achieves the expected value of at least

$$V = (1-\delta) \cdot K + \frac{\delta(n-k)}{n} \cdot \frac{K}{\epsilon},$$

where the first term is the value from scenario σ_0, and the second term is the value from scenarios σ_i such that node i in G is not in the clique. Such scenarios are unaffected by the selected nodes in L and therefore get K/ϵ clicks each. There is additional value from scenarios σ_i for i in the clique, but we disregard it for this lower bound. Thus we get the following claim.

Claim 14 *If G contains a k-clique, then there exists a solution $\boldsymbol{b}$ to I such that $E[\mathbf{value}(\boldsymbol{b})] \geq V$.*

To ensure that if there is no k-clique in G, then value V cannot be achieved by any bids, we set the parameters as follows.

1. Select ϵ such that $0 < \epsilon < \frac{1}{k+1}$.
2. Select $\delta > 0$ small enough that $\frac{1-\delta}{2} - \frac{k\delta K}{n\epsilon} > 0$. This is possible because the limit of the expression on the left as $\delta \to 0$ is $\frac{1}{2}$.
3. Let $\alpha = \frac{1}{2m}$.
4. Choose t large enough that $(k+1)\frac{K/\epsilon+\alpha t}{K+\alpha t} < \frac{1}{\epsilon}$. This is possible because $\lim_{t\to\infty}(k+1)\frac{K/\epsilon+\alpha t}{K+\alpha t} = k+1 < \frac{1}{\epsilon}$ by the choice of ϵ.

Claim 15 *If $E[\mathbf{value}(\boldsymbol{b})] \geq V$ for some fractional solution $\boldsymbol{b}$ to the constructed instance I, then there must be at least K keywords $i \in L$ such that $b_i \geq \alpha$.*

PROOF. Notice that $\sum_{i\in L} b_i \geq (K-1+m\alpha)$ implies that $|\{i \in L \mid b_i \geq \alpha\}| \geq K$, because b_i's are always at most 1.

What remains to show is that if $\sum_{i\in L} b_i < (K-1+m\alpha)$, then $E[\mathbf{value}(\boldsymbol{b})] < V$. This follows from the way we defined the parameters. Notice that $\delta\frac{K}{\epsilon}$ is an upper bound on the value that any solution can obtain from scenarios $\sigma_1 \ldots \sigma_n$. Then

$$E[\mathbf{value}(\boldsymbol{b})] < (1-\delta)(K-1+m\alpha) + \frac{\delta K}{\epsilon}$$
$$= (1-\delta)(K-\frac{1}{2}) + \frac{\delta K}{\epsilon} = V - \frac{1-\delta}{2} + \frac{k}{n}\frac{\delta K}{\epsilon} < V,$$

where the first equality comes from the definition of α, and the last inequality from the choice of δ. $\square$

Claim 16 *Let $X = \{i \in L \mid b_i \geq \alpha\}$. If H contains at least $(k+1)$ nodes in R that have neighbors in X, then $E[\mathbf{value}(\boldsymbol{b})] < V$.*

PROOF. For a node $i \in R$, let $\alpha_i = \sum_{j\in N_i} b_j$, where $N_i \subseteq L$ is the set of neighbors of i. Assuming that there are at least $(k+1)$ nodes $i \in R$ such that $\alpha_i \geq \alpha$, we show that $E[\mathbf{value}(\boldsymbol{b})] < V$. Notice that the value of a solution is always maximized by bidding on all keywords in R, because that maximizes the number of cheap clicks. So without loss of generality, we assume that $b_i = 1$ for all $i \in R$.

In a given scenario σ_i, $\boldsymbol{b}$ gets $\mathbf{clicks}^{\sigma_i} = \frac{K}{\epsilon} + \alpha_i t$, where K/ϵ clicks come from keyword i, and $\alpha_i t$ come from its neighbors in L. The cost paid in this scenario is $\mathbf{cost}^{\sigma_i} = K + \alpha_i t$, where K is spent for the cheap keywords and $\alpha_i t$ is spent on keywords of cost 1. Using $(1-\delta)K$ as an upper bound on the value obtained from σ_0, and remembering that the budget is K, we have

$$\begin{aligned}
E[\mathbf{value}(\boldsymbol{b})] &\leq (1-\delta)K + \frac{\delta}{n}\sum_i \frac{\mathbf{clicks}^{\sigma_i}}{\mathbf{cost}^{\sigma_i}/K} \\
&= (1-\delta)K + \frac{K\delta}{n}\sum_i \frac{K/\epsilon+\alpha_i t}{K+\alpha_i t} \\
&\leq (1-\delta)K + \frac{K\delta}{n}\left[(k+1)\cdot\frac{K/\epsilon+\alpha t}{K+\alpha t} + (n-k-1)\cdot\frac{1}{\epsilon}\right] \\
&< V
\end{aligned}$$

where we use the fact that the fraction in the sum increases with decreasing α_i, bound α_i by α for $(k+1)$ of the terms and by 0 for the others, and use the choice of t for the final inequality. $\square$

Clearly, if there is no k-clique in G, then every K edges in G will be incident on at least $k+1$ nodes. So from the preceding two claims, we may conclude that if G does not contain a k-clique, then no solution to I has expected value of V or more. Together with Claim 14, this proves that SBO problem in the scenario model is NP-hard.