Clustering-Based Bidding Languages for Sponsored Search

Mohammad Mahdian* Grant Wang*

Abstract

Sponsored search auctions provide a marketplace where advertisers can bid for millions of advertising opportunities to promote their products. The main difficulty facing the advertisers in this market is the complexity of picking and evaluating keywords and phrases to bid on. This is due to the sheer number of possible keywords that the advertisers can bid on, and leads to inefficiencies in the market such as lack of coverage for "rare" keywords. Approaches such as *broad matching* have been proposed to alleviate this problem. However, as we will observe in this paper, broad matching has undesirable economic properties (such as the non-existence of equilibria) that can make it hard for an advertiser to determine how much to bid for a broad-matched keyword.

The main contribution of this paper is to introduce a bidding language for sponsored search auctions based on broad-matching keywords to *non-overlapping* clusters that greatly simplifies the bidding problem for the advertisers. We investigate the algorithmic problem of computing the *optimal* clustering given a set of estimated values and give an approximation algorithm for this problem. Furthermore, we present experimental results using real advertisers' data that show that it is possible to extract close to the optimal social welfare with a number of clusters considerably smaller than the number of keywords. This demonstrates the applicability of the clustering scheme and our algorithm in practice.

1 Introduction

Search engine companies such as Google, Yahoo!, and Microsoft run today's largest auction houses, selling millions of advertising opportunities to hundreds of thousands of advertisers every day. Apart from simple budget or capacity constraints, the valuations of bidders in these auctions (known as sponsored search auctions) tend to be additive. Therefore, ignoring such constraints, from a classical auction-theory perspective of maximizing social welfare, the optimal design is to conduct an independent auction for each item, and let the advertisers submit a separate bid for each auction they are interested in. This is, in fact, very close to the mechanisms that are employed in practice.

*Yahoo! Inc., Santa Clara, CA, USA. emails: {mahdian,gjw}yahoo-inc.com

A major problem with this design is that it ignores the complexity of bidding for the advertisers. The main source of complexity in sponsored search auctions is the sheer size of the market: the number of different types of items that are up for sale in this market is huge (or practically infinite), as each keyword or phrase that a search engine user might potentially search for corresponds to an item that can be sold in the market. This is in contrast to traditional auction houses, where the number of different types of items that are offered for sale is usually not too large, and the main factor that gives rise to complexity is that bidders can have valuations for specific combinations of items.

Ignoring the complexity has lead to inefficiencies such as lack of coverage for rare yet potentially valuable phrases in the sponsored search markets. Approaches such as *broad matching* keywords that an advertiser is interested in to search queries have been applied to alleviate this problem. This means that when an advertiser bids on a keyword a using the broad match option, the search engine applies this bid to a broader set S_a of keywords that are deemed semantically related to a. Broad matching helps the advertisers to expand their bid to other potentially valuable keywords, and the search engine to increase their keyword coverage and thicken the market, thereby increasing the revenue.

However, as we will observe in Section 3, when the sets S_a of broad-matched keywords are allowed to overlap, the mechanism has undesirable economic properties that make it hard for the advertisers to determine their bid, even if we assume only one slot per keyword and a second-price pricing scheme. Intuitively, this is due to the fact that an advertiser bidding on the set S_a using broad-match on keyword a might have different valuation for different keywords in S_a, and hence if a second advertiser overbids them on some (but not all) keywords in S_a, this can change their valuation for the bundle of keywords they receive. Using this intuition, we show examples that demonstrate that in such a broad-match mechanism (with second price auctions for individual keywords), once advertisers fix the set of keywords they are bidding on, the bidding game might not have any Nash equilibrium.[1]

To resolve this issue, we restrict the bidding language to only allow broad-match sets S_a that are non-overlaping. In other words, the broad match is performed through a proper clustering of the set of all keywords into disjoint subsets, and advertisers are only allowed to bid for a cluster (and not for individual keywords). This clearly resolves the problem with the non-existence of equilibria, since an advertiser can compute his expected value for keywords in a cluster independent of other advertisers' bids (since other advertisers either take the whole cluster or none of it), and then bid this value if the underlying payment scheme is truthful (e.g., second price in the case of a single slot or more generally VCG payments), or more generally, according to the equilibria of the mechanism in the case of a single keyword (e.g.,

[1] Note that the choice of the set of keywords to bid on is not included in this game. If we include this as part of the game, the game always has an equilibrium where no advertiser uses broad match. However, our purpose is to show that assuming the advertisers have chosen to use the broad-match feature, determining the bid values might reduce to a game with no equilibrium.

as described in [7, 16] in the case of the generalized second price auction). Furthermore, to describe a cluster to the advertiser, the search engine only needs to give a random sample of a small number of keywords in the cluster (with sampling probabilities proportional to the historical query volumes) to the advertiser. The advertisers can get an accurate estimate of their value for the (potentially infinite) cluster by evaluating the random sample. Therefore, this scheme greatly simplifies keyword discovery and bidding for the advertiser.

The only drawback of restricting the advertisers to bid for clusters of keywords instead of individual keywords is that this restriction can cause a loss in social welfare. Furthermore, it is not clear how the clustering should be computed. In this paper we study these questions, both from a theoretical and an empirical perspective. First, we give approximation algorithms for the clustering problem (in two models, one deterministic and one stochastic) that given a set of estimated values and a limit k on the number of clusters, compute a clustering with social welfare within a constant factor of the optimal k-clusterings. Next, we give empirical results using real advertisers' bidding data that demonstrate that in practice, with a value of k that is significantly smaller than the total number of keywords, we can achieve a social welfare quite close to the optimal social welfare.[2]

Related work. While there has been extensive research into various aspects of the sponsored search market during the past few years (see, for example, [7, 16, 10, 1, 11, 3, 17, 9, 14, 12]), the topic of broad matching has remained largely unexplored. The only papers we are aware of that study broad match from an economic and optimization perspective are [8, 15]. Even-Dar et al. [8] study broad match from an advertiser's point of view, and present algorithms that optimize the advertiser's bids in a broad-match system taking prices as given. Singh and Roychowdhury [15] study the revenue and efficiency impacts of broad match. There are also a number of papers that study the problem of computing relevant queries for broad match or for query expansion from an information retrieval perspective; see, for example, [5]. Our observation on the lack of existence of equilibria in general broad-match bidding languages is related to the concept of *cherry picking* that has been studied in the economic literature [2], in particular in the context of the insurance industry.

Organization. The rest of this paper is organized as follows: in Section 2 we give a formal definition of the model. In Section 3 we give several examples that demonstrate that the bidding game in general broad-matched auctions might not have a full-information Nash equilibrium. In Section 4 we state the clustering problem in a deterministic and a stochastic model, and give approximation algorithms for solving this problem. Section 5 contains a presentation of our experimental results. We conclude in Section 6.

[2] Note that here the loss in social welfare due to the restriction to k-clusterings is compounded with the loss due to the fact that our algorithm is only an approximation algorithm. Therefore, our empirical results show that neither of these losses is large.

2 The Model

We study a setting where m advertisers in a set $\mathcal{A}$ bid for a number of impressions.[3] Each impression is indexed by a keyword that comes from a distribution $\mathcal{D}$ over the (potentially infinite) collection $\mathcal{K}$ of all potential keywords. We assume that advertisers have additive valuations, i.e., their value for a set of impressions is the sum of their value for individual impressions in the set.[4] Given this assumption and linearity of expectations, we can focus on the auctioning of a single impression (indexed by a keyword picked from $\mathcal{D}$). The value that an advertiser $i \in \mathcal{A}$ has for an impression indexed by a keyword $j \in \mathcal{K}$ is a fixed number denoted by $v(i, j)$. We assume that given the keyword $j \in \mathcal{K}$, the advertiser i is able to evaluate $v(i, j)$. However, since the number $|\mathcal{K}|$ of potential keywords is huge, or even infinite, we cannot expect the bidders to list all these values. Instead, the bidding language allows for a *broad match* option: for each keyword $a \in \mathcal{K}$, the search engine has a pre-computed set $S_a \subset \mathcal{K}$ of broad-matched keywords, and if an advertiser places a bid on keyword a using the broad-match option, this bid will be applied to all keywords in S_a. This way, the advertiser can bid only on a small set of keywords and the search engine expands his bid to a larger (potentially infinite) set. We assume that the search engine can communicate the sets S_a to the bidder. In practice, this can be done by giving the advertisers a random sample of keywords in S_a.

After the bidders place their bids, the search engine conducts a separate auction for each impression that arrives. That is, as a new query arrives, the search engine runs an auction between the set of bidders that have bid (either directly or through broad match) on this keyword. Throughout this paper, we assume for simplicity that there is only one ad slot on the page, and therefore only a single bidder with the highest bid will win the auction (we will discuss in Section 6 how this assumption can be relaxed). This bidder will be charged an amount equal to the second highest bid on this keyword. The utility of a bidder is the expectation of the value he receives minus the price paid, where the expectation is over the random draw of the keyword from $\mathcal{D}$.

The focus of this paper is mainly on a restriction of the broad-match bidding language,

[3] In practice, the commodities sold in sponsored search auctions are often clicks and not impressions. However, one can transform a pay-per-click system to a pay-per-impression one by multiplying each bidder's valuation on a keyword (or set of keywords) by her corresponding *click-through rate*. Therefore, we choose to simplify the model by assuming a pay-per-impression system. Of course, estimating the click-through rate of bidders is a non-trivial problem, and it is worth noting that our proposed clustering-based bidding language can in fact help with this problem by pooling together a number of rare keywords. We leave a formal study of this subject to the future work.

[4] This assumption is not entirely without loss of generality, since some advertisers in sponsored search auctions are budget-constrained. However, given the small fraction of such advertisers, the fact that in practice budget constraints are often flexible, and that incorporating budgets makes even the simplest auctions hard to analyze theoretically [4, 6], we feel that it is justified to abstract away such constraints and focus on the complexity problem.

which we call a *clustering-based bidding language*. In this restriction, the advertisers are only allowed to place broad-matched bids (i.e., no exact match), and sets S_a are selected such that for two keywords a and b, the sets S_a and S_b are either disjoint or identical. In other words, the search engine computes a clustering of the set $\mathcal{K}$ of keywords, and each bidder is only allowed to place a bid for all or none of the keywords in a cluster.[5] It is easy to see that the resulting mechanism is dominant-strategy incentive compatible in the sense that for each advertiser, it is a dominant strategy to place a bid for each cluster equal to the expected value of a keyword in that cluster (where the keyword is drawn according to the distribution $\mathcal{D}$ restricted to the cluster).

Clearly, restricting the advertisers to a clustering-based bidding language comes at a loss to the social welfare. To quantify this loss and measure its tradeoff against the complexity of the bidding language, we compare the social welfare of the truthful equilibrium with an idealized situation where each keyword is given to the advertiser with the highest valuation for that keyword.

3 Non-existence of Broad-Match Equilibria

In this section we present examples that show that non-clustering-based broad match languages can have undesirable economic properties; most importantly that the induced bidding game might not have an equilibrium. Before stating the examples, we need to clarify what we mean by the bidding game. We assume that each bidder i has already fixed the set T_i of keywords she is bidding on, and her move in the game consists of picking a single bid for this set. In other words, we do not include the picking of the keywords to bid on as part of the game, and focus on the case that i has fixed a single keyword to bid on, which maps to the set T_i through broad-match. Excluding the keyword selection from the game is essential, as without it the game always has an equilibrium where no bidder uses the broad match option. Our purpose is to show that the bidding problem might not have an equilibrium if the advertisers have already committed to use the broad-match option.

We study full-information Nash equilibria of the bidding game. We start by a very simple example that (even though contains an equilibrium) explains the essence of the problem.

Example 1. *(Non-truthfulness) Assume there are two bidders* 1 *and* 2*, and two keywords a and b occuring with the same frequency. Bidder 1 bids on $T_1 = \{a, b\}$ and bidder 2 bids on $T_2 = \{a\}$. Bidder 1 has a value of 10 for a and 2 for b, while bidder 2 has a value of 7 for a. If each bidder bids truthfully according to her expected value for the bundle, bidder 1's bid*

[5] Although not allowing advertisers to place exact match bids might sound like a harsh constraint, it can help the advertisers at the end by simplifying the bidding process and assuring them that other advertisers can not cherry pick. In fact, this is quite similar to the constraints that states often impose on the insurance industry to disallow cherry picking.

would be $(10+2)/2 = 6$, and hence she loses in the auction for a, receiving only the keyword b which has a value of 2.

The above example still has an equilibrium (in fact even an envy-free one) where bidder 1 overbids and wins both keywords. We now show that there are examples with no equilibrium. We start with a simpler example with no envy-free equilibrium (as defined by Edelman et al. [7]), and then add gadgets to this example to show that sometimes no equilibrium exists.

Example 2. *(Non-existence of envy-free equilibria) Assume there are 3 bidders 1, 2, and 3, and three keywords a, b, and c with equal frequencies. The bidders bid on the sets $T_1 = \{a, b\}$, $T_2 = \{b, c\}$, and $T_3 = \{c, a\}$, and have values $v(1, a) = v(2, b) = v(3, c) = 20$ and $v(1, b) = v(2, c) = v(3, a) = 10$. See Figure 1. Assume, without loss of generality, that the bids satisfy $b_1 \geq b_2 \geq b_3$ and 1 wins against 2 and 3 and 2 wins against 3.*[6] *Then bidder 1 wins a and b at prices b_3 and b_2, and bidder 2 wins c at price b_3. For bidder 2 not to envy the set $\{b, c\}$, we must have $(20 + 10) - (b_2 + b_3) \leq 10 - b_3$ and hence $b_2 \geq 20$. Also, for 3 not to envy $\{c\}$, we must have $b_3 \geq 20$. This means that bidder 1 ends up with negative utility.*

Note that the above example still has non-envy-free equilibria, for example one where 1 bids infinity and wins $\{a, b\}$, 2 bids 0 and wins nothing, and 3 bids 11 and wins $\{c\}$.

Example 3. *(Non-existence of equilibria) This example consists of Example 2 plus a few additional gadgets. For each advertiser $i = 1, 2, 3$, we add five new keywords d_i, e_i, f_i, g_i, and h_i, and four new advertisers x_i, y_i, z_i, and t_i with values as described in Figure 1(b) and 1(c). An edge in this example indicates that the keyword is included in the advertiser's bidding set, and the numbers on the edges indicate the values. The frequency of the new keywords (d_i's through h_i's) is equal to each other, but significantly smaller than the frequency of a, b, and c. As in Example 2, assume that $b_1 \geq b_2 \geq b_3$, and 1 wins against 2 and 3 and 2 wins against 3 (the $b_1 \geq b_3 \geq b_2$ is not entirely symmetric, but can be argued similarly). First, we make the following observations about the gadgets in Figures 1(b) and (c): the larger of the bids of x_i and y_i is at least 19 and the smaller of the bids of z_i and t_i is at most 11. This is because, if for example the larger of the bids of x_i and y_i is less than 19, the bidder with the smaller bid can strictly benefit by bidding 19. On the other hand, we argue that the bid b_1 of advertiser 1 is at least 20. This is because if this is not the case, 2 can benefit by overbidding 1 and winning the item b. Next, we argue that the second highest bid b_2 cannot be less than 19. This is because if $b_2 \leq 19$, the bidder 1 can decrease his bid to 19, still winning items a and b, but gaining by losing the item d_1 which is priced at least 19. This means that 1 is winning the item b at a price of at least 19. Therefore, to have a positive utility, the price of item c must not be more than 11. This means that the bid of 3 should not be more than 11. However, this means that this bidder can gain by winning one of the items f_i or g_i.*

[6] For simplicity we state our argument in the case of a consistent deterministic tie-breaking rule. It is not hard to see that the same reasoning works for a randomized tie breaking rule.

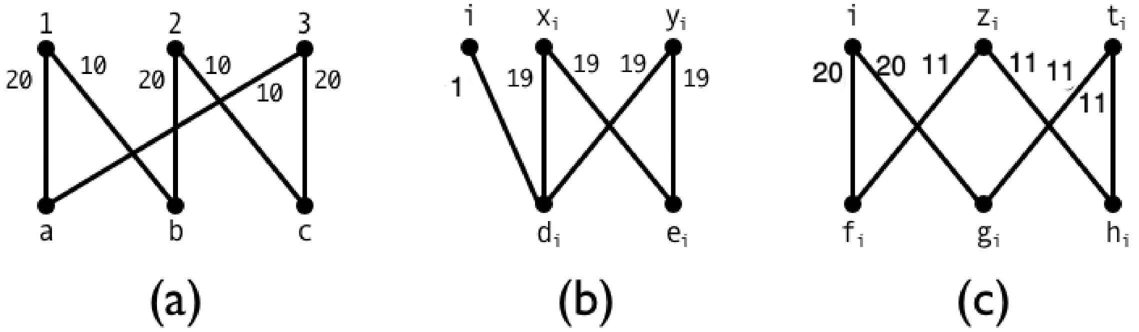

Figure 1: Non-existence of equilibria

4 The Clustering Problem

In this section, we define the search engine's problem of computing the clustering of the keywords as an optimization problem. To do this, we need to specify what kind of information the clustering algorirthm gets about the valuations of the bidders. We start with a deterministic formulation of the problem which assumes that the algorithm is given an estimated set of valuations and needs to build a clustering which maximizes social welfare with respect to this set. In practice, such estimates might be obtained by a combination of looking at historical bids on the keywords and using natural language processing or information retrieval techniques (as in [5]) to deduce other valuations. We will use historical bid data in the experiments presented in Section 5.

The clustering problem:
Given a set $\mathcal{A}$ of advertisers, a set $\mathcal{K}$ of keywords, the distribution $\mathcal{D}$ over $\mathcal{K}$, the values $v(i,j)$ for each advertiser $i \in \mathcal{A}$ and each keyword $j \in \mathcal{K}$, and a number k, compute a clustering of $\mathcal{K}$ into k clusters $S_1, \ldots, S_k$ to maximize the social welfare:

$$\sum_{r=1}^{k} \max_{i \in \mathcal{A}} v(i, S_r).$$

where $v(i, S_r) = \sum_{a \in S_r} v(i,a)$.

Since the set $\mathcal{K}$ can be very large or infinite, we would like an algorithm whose running time does not depend on $|\mathcal{K}|$. The algorithm is given access to an oracle that outputs samples according to $\mathcal{D}$ and another oracle that given i and j, outputs $v(i,j)$. The output of the algorithm also needs to be in the form of an oracle that given a keyword, decides which cluster the keyword belongs to.

It is not hard to show that the above problem is NP-hard, and therefore we cannot hope for an exact efficient solution.

Stochastic formulation. The deterministic formulation of the problem makes the assumption that the algorithm has access to a set of estimated values. This assumption, while makes the clustering problem easy to state and more useful in practice, is somewhat unsatisfactory from a theoretical perspective. Therefore, we also study another formulation of the problem, where the algroithm only knows distributional information about the values, and needs to compute a clustering that optimizes the expected social welfare. We assume the distribution of the values is explicitly given to the algorithm, i.e., there is a fixed number ℓ of different *types* of advertisers, with each type t is associated with a probability p_t and a set of valuations $v(t, j)$ for an advertiser of this type (given using an oracle as described above). There are m advertisers, each having a type picked according to the probabilities p_t. The type of the advertiser determines his valuation. We will discuss this formulation of the problem in Section 4.2, and give an approximation algorithm in cases where probabilities p_t are not too small.

4.1 A $(1 - 1/e)$-approximation algorithm for the deterministic clustering problem

In this section, we give an approximation algorithm for the deterministic formulation of the clustering problem by reducing it to submodular function maximization. The main component of the solution is the definition of the function.

We define a function f from the domain $\mathcal{A}$ to the set of non-negative reals. The value of the function on a set $R \subset \mathcal{A}$ is defined as follows:

$$f(R) = \mathrm{Exp}_{j \leftarrow \mathcal{D}}[\max_{a \in R} v(a, j)].$$

Corresponding to the set $R \subset \mathcal{A}$, we define a $|R|$-clustering C_R of $\mathcal{K}$ as follows: each cluster S_a in C_R corresponds to an advertiser a in R, and each keyword $j \in \mathcal{K}$ is assigned to the cluster corresponding to an advertiser $a \in \mathrm{argmax}_{a \in R} v(a, j)$. Note that with this definition and the definition of f, the value of $f(R)$ is precisely the expected value of a keyword in a cluster to the advertiser in R that this cluster is assigned to, i.e.,

$$f(R) = \sum_{a \in R} \Pr_{j \leftarrow \mathcal{D}}[j \in S_a] \cdot \mathrm{Exp}_{j \leftarrow \mathcal{D}|S_a}[v(a, j)].$$

This implies:

Lemma 1. *For every set $R \subseteq \mathcal{A}$, the social welfare of the clustering C_R is at least $f(R)$.*

Proof. Consider a set S_a in the clustering C_R and assume this set corresponds to the advertiser $a \in R$. The expected value of the advertiser a for this cluster is precisely $\mathrm{Exp}_{j \leftarrow \mathcal{D}|S_a}[v(a,j)]$. Therefore, the expected value of the advertiser with the maximum value for this cluster must be at least this much. Hence, if each cluster in C_R is assigned to the advertiser with the maximum expected value, the resulting welfare will be at least $f(R)$. □

Note that the inequality in the above lemma can sometimes be strict, i.e., the correspondence between the set R of advertisers and the clustering C_R is not one-to-one. However, we can show that at the social optimum, this correspondence *is* one-to-one.

Lemma 2. *Let OPT denote a partitioning of $\mathcal{K}$ into at most k clusters that maximizes the social welfare among all such clusterings. Let R^* denote the set of advertisers that win (i.e., have the maximum value for) at least one cluster in OPT. Then the social welfare of the clustering OPT is precisely $f(R^*)$.*

Proof. By Lemma 1, the welfare of the clustering C_{R^*} is at least $f(R^*)$. On the other hand, by the definition of f, if for each keyword $j \in \mathcal{K}$, a_j is the advertiser to whom j is assigned in OPT, we have

$$f(R^*) = \sum_{j \leftarrow \mathcal{D}} [\max_{a \in R^*} v(a,j)] \geq \sum_{j \leftarrow \mathcal{D}} [v(a_j, j)] = OPT.$$

Therefore, $OPT \leq f(R^*) \leq \mathrm{welfare}(c_{R^*})$. However, since OPT is the optimal clustering, we must have equality. □

By the above two lemmas, it is clear that a c-approximation algorithm for finding the maximum value of $f(R)$ subject to $|R| \leq k$ gives a c-approximation algorithm for the clustering problem: the clustering would simply be C_{R^*}, where R^* is the set that approximately maximizes f. Furthermore, the resulting clustering can be given compactly as a membership oracle: for each given keyword $j \in \mathcal{K}$, the oracle simply evaluates $v(a,j)$ for $a \in R^*$ and assigns j to the cluster with the maximum value.

The only thing that remains is to give an algorithm that maximizes f. Fortunately, this is easy as the function f is submodular.

Lemma 3. *The function f is submodular and non-decreasing.*

The proof of the above lemma is easy and is omitted here. Given the submodularity of f, the maximization problem can be solved using the classical greedy algorithm of Nemhauser et al. [13] with an approximation factor of $1 - 1/e$. This algorithm consists of iteratively picking an element (in this case an advertiser $a \in \mathcal{A}$) with the maximum marginal value. In the case of our problem where the input is given using oracles, instead of computing the marginal value of each $a \in \mathcal{A}$ exactly in each iteration, we can use sampling, with a sample size of $O(\log k)$ to estimate this marginal value to with a small error. It is not hard to bound the total error using the union bound and show that the greedy algorithm with an approximate evaluation of the marginal values can achieve an approximation factor arbitrarily close to $1 - 1/e$ with high probability. The details of the argument are straightforward and are omitted from the conference version of this paper.

Theorem 1. *For every constant $\epsilon > 0$, there is an algorithm that approximates the deterministic formulation of the clustering problem to within a factor of $1 - \frac{1}{e} - \epsilon$. The running time of this algorithm is polynomial in $|\mathcal{A}|$ and k and does not depend on $|\mathcal{K}|$.*

4.2 An approximation algorithm for the stochastic clustering problem

In this section, we give an algorithm for the stochastic formulation of the clustering problem, in cases where the probability of different types of advertisers is not too small. Our main result is the following.

Theorem 2. *Assume an instance of the stochastic clustering problem satisfies $\min_t p_t \geq \frac{1}{cm}$ for a constant c. Then there is a polynomial time algorithm that approximately solves this problem on such instance with an approximation ratio of $\frac{1}{2}(1 - \frac{1}{e})$.*

The rest of this section contains a sketch of the proof of the above theorem. The main idea behind the algorithm is as follows: first, we use the same techniques as the ones used in the deterministic case to reduce the problem to a submodular function maximization problem. However, in this case, the submodular function is considerably more complex: it is a function defined on set of all permutations of advertiser types. Therefore, the greedy approximation algorithm of Nemhauser et al. [13] cannot be implemented on this function in polynomial time.[7] To resolve this problem, we show that the assumption about the instance allows us to truncate the permutations and approximate the objective function by a related objective function that is defined on a polynomial-sized space. The latter objective function also turns out to be submodular, and therefore, we can use the greedy algorithm to optimize this function. Furthermore, since the function is defined on a polynomial-sized space, each iteration of the greedy algorithm can be implemented in polynomial time by exhaustively searching the space for the element with the maximum marginal value.

We present the algorithm in several steps.

Step 1 (Restatement of the problem). Let Π denote the set of all permutations of the set of types $\{1, \ldots, \ell\}$. For a permutation $(\pi_1, \ldots, \pi_\ell)$, we define

$$q_i^\pi = \Pr[\text{there is no bidder of type } \pi_1, \ldots, \text{ or } \pi_{i-1}.] = \left(1 - \sum_{r=1}^{i-1} p_{\pi_r}\right)^m.$$

[7] In fact, we can prove that the problem of computing the permutation with the maximum marginal value that the greedy algorithm needs for each iteration is NP-hard.

We omit the superscript π when there is no ambiguity. For a keyword j and a permutation π, we define

$$v(\pi, j) = \sum_{i=1}^{\ell} (q_i - q_{i+1}) v(\pi_i, j).$$

For a collection $\mathcal{S}$ of permutations in Π, we define the value of this collection as follows:

$$\text{value}(\mathcal{S}) = \text{Exp}_{j \leftarrow \mathcal{D}}[\max_{\pi \in \mathcal{S}} \{v(\pi, j)\}].$$

The problem is to find a collection $\mathcal{S} \subset \Pi$ of size at most k that has the maximum value. We call this problem $P1$.

For a collection $\mathcal{S} \subset \Pi$ and permutation $\pi \in \mathcal{S}$, we define

$$\text{Cluster}(\pi, \mathcal{S}) = \{j \in \mathcal{K} : \forall \pi' \in \mathcal{S}, v(\pi, j) \geq v(\pi', j)\}.$$

For simplicity, assume all values $v(\pi, j)$ are distinct numbers and therefore $\text{Cluster}(\pi, \mathcal{S})$ for $\pi \in \mathcal{S}$ cluster the set $\mathcal{K}$ (if some values are tied, one can arbitrarily break the ties). It is not hard to use an argument similar to the one used in the previous section for the deterministic case to prove that a c-approximation algorithm for $P1$ gives a c-approximation algorithm for the clustering problem.

Denote the distribution $\mathcal{D}$, restricted to the cluster $\text{Cluster}(\pi, \mathcal{S})$ by $\mathcal{D}(\pi, \mathcal{S})$. The objective function can be rewritten as

$$\text{value}(\mathcal{S}) = \sum_{\pi \in \mathcal{S}} \text{Exp}_{j \leftarrow \mathcal{D}(\pi, \mathcal{S})}[v(\pi, j)] = \sum_{\pi \in \mathcal{S}} \sum_{i=1}^{\ell} (q_i - q_{i+1}) \text{Exp}_{j \leftarrow \mathcal{D}(\pi, \mathcal{S})}[v(\pi_i, j)].$$

Lemma 4. *Let $\mathcal{S}$ denote the optimal solution to the problem $P1$. Then for every $\pi \in \mathcal{S}$ the value $\text{Exp}_{j \leftarrow \mathcal{D}(\pi, \mathcal{S})}[v(\pi_i, j)]$ is a non-increasing function of i.*

Proof. Assume this is not true, i.e., for some $\pi \in \mathcal{S}$ and some i, we have

$$\text{Exp}_{j \leftarrow \mathcal{D}(\pi, \mathcal{S})}[v(\pi_i, j)] < \text{Exp}_{j \leftarrow \mathcal{D}(\pi, \mathcal{S})}[v(\pi_{i+1}, j)].$$

Then by swapping π_i and π_{i+1}, we can strictly increase the value of the collection $\mathcal{S}$, contradicting its optimality. □

Step 2 (Truncating permutations). For a permutation π, we define an index $i^*(\pi)$ as follows: $i^*(\pi)$ is the largest index i for which $q_i \geq 1/2$. Equivalently, it is the largest i for which $\sum_{r=1}^{i-1} p_{\pi_r} \leq 1 - 2^{-1/m}$. We now define

$$v'(\pi, j) = \sum_{i=1}^{i^*(\pi)} (q_i - q_{i+1}) v(\pi_i, j). \tag{1}$$

Similar the above, we define $\text{value}'(\mathcal{S})$ and $\text{Cluster}'(\pi, \mathcal{S})$, and define the problem $P2$ as the problem of finding a collection $\mathcal{S}$ that maximizes $\text{value}'(\mathcal{S})$. Note that since elements of π that come aftr $i^*(\pi)$ do not affect the values $v'(\pi, j)$, the problem $P2$ is equivalent to finding a collection $\mathcal{S}$ maximizing $\text{value}'(S)$ from the set Π' of permutations $(\pi_1, \ldots, \pi_s)$ of subsets of $\{1, \ldots, \ell)$ satisfying $\sum_{r=1}^{s-1} p_{\pi_r} \leq 1 - 2^{-1/m}$.

Lemma 5. *Any α-approximate solution $\mathcal{S}$ to problem $P2$ is a 2α-approximate solution to problem $P1$.*

Proof. Let OPT (OPT', respectively) be the optimal solution of the problem $P1$ ($P2$, respectively). By our assumption, $\text{value}'(\mathcal{S}) \geq \frac{1}{\alpha}\text{value}'(OPT') \geq \frac{1}{\alpha}\text{value}'(OPT)$. By definition of value$'$, it is clear that $\text{value}(\mathcal{S}) \geq \text{value}'(\mathcal{S})$. Furthermore, we claim that by Lemma 4, $\text{value}'(OPT) \geq \frac{1}{2}\text{value}(OPT)$. □

Step 3 (Greedy algorithm for problem $P2$). The final step is to solve the problem $P2$ using the greedy submodular function maximization algorithm. To do this, first, we need to prove that the function value$'$ is non-decreasing and submodular. The proof of this fact is straightforward and is omitted here. We also need to show that the space Π' is of polynomial size. This follows from our assumption on the instance. Recall that we had $\min_t p_t \geq \frac{1}{cm}$ for a constant c. Therefore, the length of a permutation in Π' is at most

$$\frac{1 - 2^{-1/m}}{\min_t p_t} \leq cm(1 - 2^{-1/m}) = cm(1 - e^{-(ln2)/m}) \leq cm \cdot \frac{\ln 2}{m} = c \ln 2.$$

Therefore, there are at most polynomially many permutations in Π'. By exhaustively searching these permutations and using sampling as described in the deterministic case, we can solve each iteration of the greedy algorithm in polynomial time. This completes the proof of Theorem 2.

5 Experimental results

In this section, we describe the results of experiments with the greedy algorithm for computing a k-clustering of a set of keywords in the single-slot model. The results of the experiment show that the social welfare of the k-clustering computed by the algorithm is a significant fraction of the total social welfare, even for values of k that are orders of magnitude smaller than the total number of keywords. The resulting k-clusterings are natural and partition the keywords into subsets of roughly the same meaning. Lastly, we show that the k-clusterings obtained by the algorithm are stable over time; the partition of keywords computed from a data set at time t_1 still obtains a large fraction of the social welfare at time $t_2 > t_1$.

5.1 Dataset

The dataset we used in the experiments was a matrix M where the rows are keywords and the advertisers are columns, with M_{ij} being equal to the monthly spend of advertiser j on keyword i. The number of keywords in the dataset was roughly 50,000. We interpreted the spend of advertiser j on keyword i to be the value of the keyword i to advertiser j. Because we are working with the single-slot model, we consider the total social welfare to be the total value obtained when each keyword is assigned to the advertiser with maximum value, i.e.

$$\mathsf{total\ social\ welfare} = \sum_i \max_j M_{ij}$$

5.2 Experiments

We ran the greedy algorithm for computing a k-clustering on our dataset for $k = 1$ to $5,000$. For each value of k, we computed the percentage of total social welfare that is obtained by the k-clustering. The plot of percentage of total social welfare vs. k is the solid line in Figure 2. When $k = 1,000$, we obtain nearly 80% of the total social welfare, while reducing the complexity of the bidding language (the number of items an advertiser can bid on) by a factor of 50. We also ran the greedy algorithm with the row-sums of the matrix M normalized to 1. By normalizing the value of each keyword to 1, we ensure that the values of the keywords are not the primary reason that a k-clustering obtains a large fraction of the total social welfare. The plot of percentage of total social welfare vs. k for the normalized case is the dotted line. While the dotted line is dominated by the solid line, we still see a significant reduction in the complexity of the bidding language while obtaining a significant fraction of social welfare (e.g. at $k = 2,500$, the k-clustering obtains roughly 80% of the total social welfare).

We also compared the social welfare v at time t_2 of the k-clustering C_1 computed on a dataset from time t_1 to the social welfare w at time t_2 of the k-clustering C_2 computed from t_2 for various values of k. The plot of v/w can be found in Figure 3. For small values of k, v/w is quite volatile, but for large enough k, C_1 obtains more than 80% of the social welfare of C_2.

A sample of keywords from the clusters in the k-clustering for $k = 1,000$ can be seen in Table 1.

6 Discussion

We have introduced the idea of a clustering-based bidding language for sponsored search. Clustering-based bidding languages reduce the complexity of bidding for the advertiser. In addition, they avoid the undesirable economic properties of *broad matching*, where advertisers

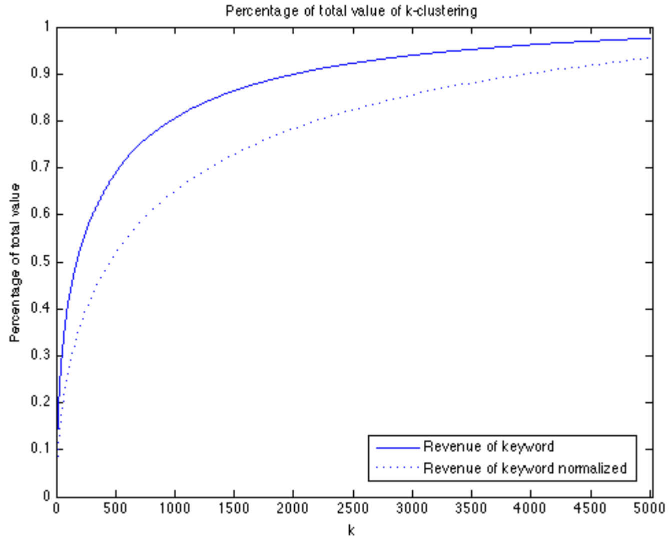

Figure 2: Percentage of total social welfare vs. k

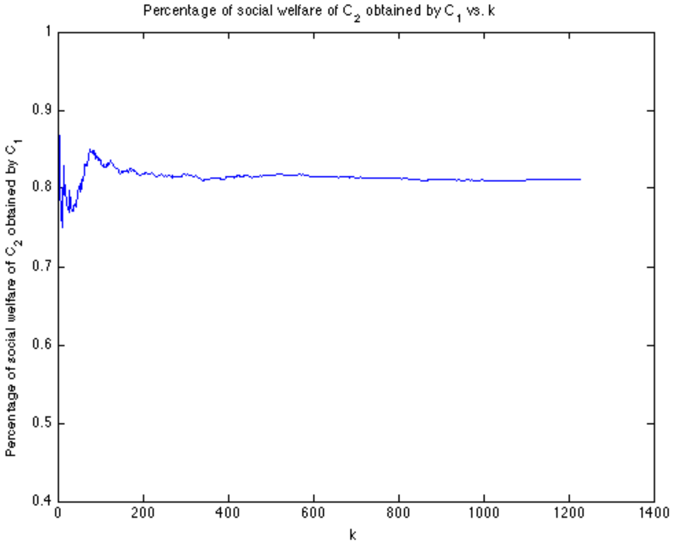

Figure 3: Time stability of k-clustering

Cluster 1	**Cluter 2**	**Cluster 3**	**Cluster 4**
computer fax	septic system	alcohol addiction	brooch pin
download free fax software	septic tank	alcohol addiction treatment	gem stone ring
electronic fax	storm shelter	drug addiction	gem stone
email by fax	timberline shingles	substance abuse	indian jewelry
free computer fax software	tree service	substance abuse treatment	tanzanite jewelry

Table 1: Sample keywords and clusters from a $1,000$-clustering

can bid on overlapping sets of keywords. We gave approximation algorithms for computing a k-clustering that maximizes social welfare to within a constant factor in both a deterministic and stochastic formulation of the problem. Experimental results show that a k-clustering obtains a large fraction of the total social welfare when k is several orders of magnitude smaller than the total number of keywords.

It is worth noting that the greedy algorithm we give for the deterministic formulation of the problem with single slots can be extended to the case of multiple slots. The algorithm remains largely the same – instead of greedily choosing the advertiser that obtains the largest incremental gain in social welfare, we choose the *slate* of advertisers that obtain the largest incremental gain. This algorithm gives a $(1-1/e)$ approximation, since the underlying function we are maximizing is still submodular. When the number of slots is a constant, this gives us a polynomial-time approximation algorithm. However, if the number of slots is not a constant, we do not know how to implement the greedy step in polynomial time.

While we showed experimentally that the k-clustering is relatively stable over time, eventually the bidding language would need to change. The problem of *evolving* a clustering-based bidding language, both in theory and in practice, is an interesting one.

In theory, increasing the power (and complexity) of a bidding language can only increase social welfare. In practice, there is a tradeoff – as bidding languages become more expressive, social welfare can actually decrease. It would be interesting to both study this experimentally and come up with suitable models.

References

[1] Gagan Aggarwal, Ashish Goel, and Rajeev Motwani. Truthful auctions for pricing search keywords. In *EC '06: Proceedings of the 7th ACM conference on Electronic commerce*, pages 1–7, New York, NY, USA, 2006. ACM.

[2] Yoram Barzel. Measurement cost and the organization of markets. *Journal of Law and Economics*, 1982.

[3] Christian Borgs, Jennifer Chayes, Nicole Immorlica, Kamal Jain, Omid Etesami, and Mohammad Mahdian. Dynamics of bid optimization in online advertisement auctions. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 531–540, New York, NY, USA, 2007. ACM.

[4] Christian Borgs, Jennifer Chayes, Nicole Immorlica, Mohammad Mahdian, and Amin Saberi. Multi-unit auctions with budget-constrained bidders. In *EC '05: Proceedings of the 6th ACM conference on Electronic commerce*, pages 44–51, 2005.

[5] Andrei Z. Broder, Marcus Fontoura, Evgeniy Gabrilovich, Amruta Joshi, Vanja Josifovski, and Tong Zhang. Robust classification of rare queries using web knowledge. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 231–238, New York, NY, USA, 2007. ACM.

[6] Shahar Dobzinski, Ron Lavi, and Noam Nisan. Multi-unit auctions with budget limits. *Annual IEEE Symposium on Foundations of Computer Science*, pages 260–269, 2008.

[7] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 2007.

[8] Eyal Even-dar, Yishay Mansour, Vahab Mirrokni, S. Muthukrishnan, and Uri Nadav. Bid optimization for broad match ad auctions. to appear in WWW'09. available online at `http://arxiv.org/abs/0901.3754`.

[9] Jon Feldman, S Muthukrishnan, Martin Pal, and Cliff Stein. Budget optimization in search-based advertising auctions. In *EC '07: Proceedings of the 8th ACM conference on Electronic commerce*, pages 40–49, New York, NY, USA, 2007. ACM.

[10] David Martin, Johannes Gehrke, and Joseph Halpern. Toward expressive and scalable sponsored search auctions. In *ICDE Conference*, 2008.

[11] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. Adwords and generalized online matching. *J. ACM*, 54(5):22, 2007.

[12] S. Muthukrishnan, M. Pal, and Z. Svitkina. Stochastic models for budget optimization in search-based advertising. In *Proc. Workshop on Internet and Network Economics (WINE)*, 2007.

[13] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - i. *Math. Prog.*, 14:265–294, 1978.

[14] Paat Rusmevichientong and David P. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In *EC '06: Proceedings of the 7th ACM conference on Electronic commerce*, pages 260–269, New York, NY, USA, 2006. ACM.

[15] Sudhir Kumar Singh and Vwani P. Roychowdhury. To broad-match or not to broad-match: An auctioneer's dilemma. In *Workshop on Ad Auctions*, 2008.

[16] H. Varian. Position auctions. *To appear in International Journal of Industrial Organization*, 2006.

[17] Yunhong Zhou, Deeparnab Chakrabarty, and Rajan Lukose. Budget constrained bidding in keyword auctions and online knapsack problems. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 1243–1244, New York, NY, USA, 2008. ACM.