Online Adwords Allocation

Shoshana Neuburger

May 6, 2009

1 Overview

Many search engines auction the advertising space alongside search results. When Google interviewed Amin Saberi in 2004, their advertisement model became the focus of a new research area. Although he did not land the job, the interview was an important stimulus in his academic career. Upon his return to Georgia Tech, Saberi shared the question with another graduate student, Aranyak Mehta, and their thesis advisor, Vijay Vazirani.

Vazirani recognized a connection between this new problem and the online bipartite matching problem, a problem he had solved 14 years earllier. When the Vazirani brothers and Karp proved a solution to it in 1990, the researchers saw online matching as a beautiful research problem. Umesh Vazirani says. "At the time, we had no idea it would turn out to have practical value." [7]

Based on previous algorithms for online allocation, Mehta et. al. devised an innovative deterministic algorithm. They prove the optimality of their algorithm in the general case when there is no prior knowledge about the input queries. The primary assumption of the algorithm is that bids are small compared to an advertiser's daily budget. [5]

Together with Gagan Goel, Mehta explored the Adwords problem when assumptions can be made about the distribution of search queries. They showed that a simple greedy algorithm achieves the same competitive ratio as the more complex algorithm of Mehta et. al. in the random input model, as wall as in i.i.d. input models. This factor is $1 - \frac{1}{e}$. [1]

2 Online Bipartite Matching

The online bipartite matching problem is classically described with the following problem. A matchmaker and n boys are gathered in a room. n girls appear, one at a time. Each girl has a list of boys who are acceptable to her, which she reveals to the matchmaker as she appears. The matchmaker immediately matches the new girl to one of the boys on her list, if any of them are available. The goal is to maximize the number of matches. Vazirani, Vazirani, and Karp devised a randomized algorithm that achieves the optimal competitive ratio of $1 - \frac{1}{e}$.

Online bipartite matching can also be described as the task of finding a matching of minimum weight in a bipartite graph. An edge connects a server node to a request it can handle. The set of

server nodes is known, while the set of requests arrive one at a time. As a node arrives, the weights of all edges incident on the node, i.e., servers that can fulfill the request, are divulged. Each server can service a single request. The task is to find a minimum weight matching of requests to servers online as each node arrives. [3]

The effectiveness of an online algorithm is measured by *competitive analysis*. We say an algorithm ALG is α-competitive if the ratio between its performance and the optimal offline performance, OPT, is bounded by α. In other words,

$$\frac{ALG(I)}{OPT(I)} \geq \alpha$$

for any instance of the problem I.

Karp et. al. describe an optimal randomized algorithm for online bipartite matching in [4]. The input is represented as adjacency matrix B for G(U,V,E), a bipartite graph on $2n$ vertices that contains a perfect matching. We can let the rows represent the nodes in U, the boys, and the columns represent the nodes in V, the girls. Then, an entry in the adjacency matrix B, b_{ij}, is a boolean value.

$$b_{ij} = \begin{array}{l} 1, \text{ if boy i is acceptable to girl j,} \\ 0, \text{ otherwise.} \end{array}$$

The online matching is constructed while the matrix is revealed column-by-column.

We can consider a simple greedy algorithm. GREEDY matches a girl whenever possible. GREEDY achieves a maximal matching of size at least $n/2$. An adaptive adversary can limit any deterministic algorithm to a matching of size $n/2$. As an example, let the first $n/2$ columns contain all ones and the last $n/2$ columns contain ones only in those rows which are matched by the deterministic algorithm in the first $n/2$ steps. Thus, the competitive ratio of GREEDY is $\frac{1}{2}$.

The randomized algorithm RANKING of [4] achieves a competitive ratio of $1 - \frac{1}{e}$. The algorithm fixes a random permutation of the first set of vertices, assigning a rank to each boy. As each girl arrives, she is matched to the eligible boy (if any) of highest rank. Karp et. al. proved that this algorithm is optimal for online bipartite matching.

Online bipartite matching is a special case of the Adwords allocation problem in which advertisers are restricted to unit bids and unit daily budgets. For this special case, the RANKING algorithm matches ads as they arrive with a competitive factor of $1 - \frac{1}{e}$.

3 Online b-matching

The work of Karp, Vazirani, and Vazirani was extended to b-matching by Bala Kalyanasundaram and Kirk Pruhs in [2]. In a b-matching prolem, each server can be matched to several requests within a predetermined service limit. This is unlike the assignment problem in which a server vertex is matched only once. A b-matching problem seeks to allocate requests to servers so that the maximum number of requests are serviced. In effect, the unused service is kept to a minimum. Kalyanasundaram and Pruhs developed a deterministic algorithm called BALANCE. Their

algorithm assigns each request to the server site with the largest remaining number of servers. Kalyanasundaram and Pruhs demonstrated that their deterministic algorithm is optimal, achieving a competitive ratio of $1 - \frac{1}{e}$.

The Adwords problem corresponds to b-matching if the advertisers can specify any budget, just as each server site has the capacity to fill any number of requests. Although an advertiser can limit his interest to specific keywords, bids must be offered at a set rate. The BALANCE algorithm assigns each keyword to the advertiser with largest remaining budget.

The adwords matching problem is a generalization of online matching and b-matching to the case of varying budgets and bids. The input that arrives online, the girls or requests for service, are queries. The boys or serveres are the advertisers, with their respective bids and daily budgets. The goal is to maximize the revenue from allocated ads, while respecting the daily budget of each advertiser. Initially, the problem was simplified, with the assumption that only one ad is allocated at each query. Later, the solution was generalized to the case of ranking multiple ads for each query.

4 Adwords and Generalized Online Matching

Many search engines display ads alongside query results. Although they are based on the same keywords, the ads are displayed separately from the query results. The search engine collects profit from the advertisements. Each advertiser can place a bid for a keyword that relates to his line of business. The bid amount can differ among advertisers as they have different valuations for a keyword. The advertisement of the winning bidder is displayed and he is charged the bid price for the ad. This is an assignment problem since the ads are displayed as each query arrives. It is online since the search engine cannot anticipate future queries. The Adwords problem is to design an allocation algorithm that maximizes the search engine's daily revenue.

An initial algorithmic solution to the Adwords problem was presented at FOCS 2005 [5]. The journal version appeared in JACM 2007, [6]. The advertisers' daily budgets play a role in the algorithm. It is in the search engine's interest to ensure that each advertiser remains solvent as long as possible. Thus, the algorithm weighs the remaining fraction of each advertiser's budget against the amount of its bid.

We can state the Adwords problem formally:

- N bidders
- Each bidder i has daily budget b_i
- Each bidder i specifies a bid c_{iq} for query word $q \in Q$
- A sequence of query words $q_1 q_2 \cdots q_M$, $q_j \in Q$, arrive online during the day.
- Each query q_j must be assigned to some bidder i as it arrives; the revenue is c_{iq_j}

Objective: maximize total daily revenue while respecting daily budget of bidders.

4.1 Greedy Algorithm

The overall goal of the Adwords problem is to maximize revenue. It is natural to consider an algorithm that maximizes profit per query. We can consider a greedy algorithm, GREEDY, that chooses the ad that offers the highest bid for a keyword that arrives. Bids are only considered from an advertiser who has not already spent his entire budget. GREEDY achieves a competitive ratio of $\frac{1}{2}$.

To examine GREEDY, an adaptive adversary might present the following worst-case scenario. There are two advertisers, $Bidder_1$ and $Bidder_2$, whose bids are in the following table.

	$Bidder_1$	$Bidder_2$
Bicycle	\$1	\$0.99
Flowers	\$1	\$0
Budget	\$100	\$100

Suppose 100 queries arrive for Bicycle, followed by 100 queries for Flowers. GREEDY allocates 100 Bicycles to $Bidder_1$ accruing \$100. On the other hand, the optimal offline algorithm allocates 100 Bicycles to $Bidder_2$ and 100 Flowers to $Bidder_1$ for a total revenue of \$199.

The algorithm proposed by Mehta et. al. in [5] achieves a competitive ratio of $1 - \frac{1}{e} \approx .63$. This is not only an improvement over GREEDY. This factor is, in fact, optimal.

4.2 MSVV Algorithm

A natural algorithm for the Adwords problem might be to greedily assign queries to the highest bidder and then break ties with the largest remaining budget. Mehta et. al. provide an example to demonstrate that this algorithm achieves a factory strictly less than $1 - \frac{1}{e}$. The more intricate algorithm that they suggest is $1 - \frac{1}{e}$-competitive.

The algorithm of Mehta et. al. [5] is developed for a simplified version of the Adwords problem. A bidder pays as soon as his ad is displayed, not when it is clicked. A bidder pays his own bid; this is not a second price auction. Only one ad is displayed per search query. These assumptions can easily be relaxed later. However, bids must be small relative to budgets for the algorithm to be effective.

There are several assumptions that simplify the proofs in the paper, which are later relaxed. Budgets are all of equal, unit size. The best offline algorithm will completely exhaust each bidder's entire budget. That is, we assume the optimal revenue is exactly \$N.

The algorithm weighs the remaining fraction of each advertiser's budget against the amount of his bid. There is a tradeoff function, ψ, between these two parameters. $\psi(x) = 1 - e^{-(1-x)}$. The algorithm gives the ad to the bidder who maximizes bid $bid \times \psi(fraction\ of\ budget\ spent)$. A graph of the tradeoff function ψ is shown in the figure.

Mehta et. al. present a new proof for BALANCE that is obtained by a factor-revealing linear program. The linear program is then modified to accommodate arbitrary bids using a tradeoff-revealing linear program. The dual of this linear program reaveals the tradeoff function ψ. Thus,

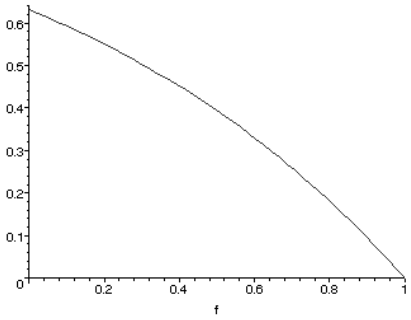

Figure 1: Tradeoff Function

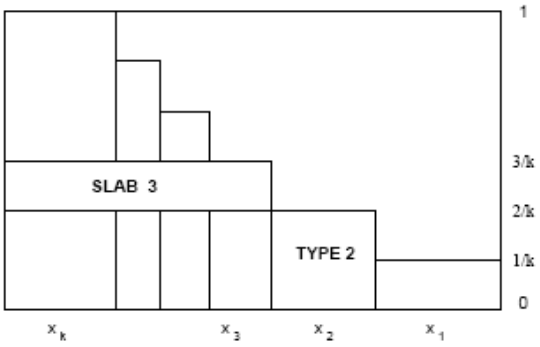

Figure 2: Discretization of Budget

the new algorithm also has a competitive ratio of $1 - \frac{1}{e}$. The tradeoff function is the basis of the new algorithm. In the special case when all bids are equal, the new algorithm is precisely BALANCE provided by [2].

The budget of each advertiser is discretized into k equal parts numbered 1 through k. k can be any large integer. A bidder spends money in slab j before proceeding to slab $j+1$. A bidder is classified by the fraction of his budget that is spent at the end of the algorithm. A bidder is of type j if the fraction of his budget spent at the end of the algorithm is in $(\frac{j-1}{k}, \frac{j}{k}]$. Then, we can define x_i as the number of bidders of type i. Alternatively, x_i is the number of bidders who spend i/k of their budget at the end of BALANCE. x_i is defined for $1 \leq i \leq k$. The slabs are depicted in the accompanying image.

Let $\psi_k : [1 \cdots k] \rightarrow R^+$ be the following (monotonically decreasing) function: $\psi_k(i) = 1 - e^{-(i-i/k)}$ Note: $\psi_k \rightarrow \psi$ as $k \rightarrow \infty$.

$$\text{Revenue} = \sum_{i=1}^{k} x_i \frac{i}{k}$$

The factor revealing LP bounds the number of bidders of type j for small j. We can examine the area of each rectange in the figure to identify the constraints. The first constraint is: $x_1 \leq \frac{N}{k}$. Similarly, the second constraint is: $x_1 + x_2 \leq 2\frac{N}{k} - \frac{x_1}{k}$. Rearranging the terms reveals the following generalization of x_i: $\forall i, 1 \leq i \leq k-1$: $\sum_{j=1}^{i} (1 + \frac{i-j}{k}) x_j \leq \frac{i}{k} N$.

The following LP, L, gives a lower bound on BALANCE. i ranges from 1 to $k-1$.

$$\begin{aligned} \text{Maximize} \quad & \sum_{i=1}^{k-1} \frac{k-i}{k} x_i \\ \text{such that} \quad & \forall i : \sum_{j=1}^{i} (1 + \frac{i-j}{k}) x_j \leq \frac{i}{k} N \\ & \forall i : x_i \geq 0 \end{aligned}$$

The dual LP, D, follows.

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{k-1} \frac{i}{k} N y_i \\ \text{such that} \quad & \forall i : \sum_{j=i}^{k-1} (1 + \frac{j-i}{k}) y_j \geq \frac{k-i}{k} \\ & \forall i : y_i \geq 0 \end{aligned}$$

The primal LP, L, can be written as

Max $c \cdot x$ s.t. $Ax \leq b$ $x \geq 0$.

The dual LP, D, can be written as

Min $b \cdot y$ s.t. $A^T y \geq c$ $y \geq 0$.

The linear program is solved by finding the optimum primal and dual solution. The optimal solution is $x_i = \frac{N}{k}(1-\frac{1}{k})^{i-1}$, which tends to $N(1-\frac{1}{e})$ as $k \to \infty$. Thus, BALANCE achieves a factor of $1-\frac{1}{e}$.

In the more general Adwords problem, there is a subtle tradeoff between bids and unspent budget. Thus, it is difficult to extend results obtained via the factor-revealing LP to the case of arbitrary bids. We generalize the primal LP L and its dual D using a series of linear programs, $L(\pi, \psi)$. For every monontically decreasing tradeoff function ψ and every instance π of the Adwords problem we write a new LP $L(\pi, \psi)$ for our algorithm.

There is a tradeoff-revealing family of LPs for the Adwords problem. We can formulate the LPs, $L(\pi, \psi)$, and their duals, $D(\pi, \psi)$. The only change from L to $L(\pi, \psi)$ is in the introduction of a new term, Δ, in the right-hand side of the constraints. The dual's constraints are identical, but the objective function of $D(\pi, \psi)$ includes a new Δ term.

$L(\pi, \psi)$:

Max $c \cdot x$ s.t. $Ax \leq b + \Delta(\pi, \psi)$

$D(\pi, \psi)$:

Min $b \cdot y + \Delta(\pi, \psi) \cdot y$ s.t. $y^T A \geq c$

The authors made the following observation: For every ψ, the dual LP achieves its optimal value at the same vertex. Furthermore, there is a way to choose ψ so that the objective function does not decrease. Thus, the competitive factor remains $1-\frac{1}{e}$. This competitive factor is optimal.

Theorem: The competitive ratio of BALANCE is exactly $1-\frac{1}{e}$.

Proof. We know the size of the matching is $\geq N - \Phi - \frac{N}{k}$, so it tends towards $N(1-\frac{1}{e})$. We take it for granted that the optimal offline allocation achieves a revenue of \$N. Thus, the competitive ratio is $\geq 1-\frac{1}{e}$.

There is an instance of the problem that at the end of the algorithm all inequalities are tight. One example is provided in [2]. □

4.3 More Realistic Models

The algorithm introduced in [5] generalizes to

- Advertisers with different daily budgets.
- The optimal allocation does not exhaust all budgets.
- Several ads per search query.
- Cost-per-Click
- Second Price

These generalizations follow easily from the analysis of the LPs and the tradeoff function.

5 Random input models

Mehta continued working on the Adwords problem with Gagan Goel . They presented their results at SODA 2008 [1]. GREEDY is a natural algorithm for the Adwords problem that selects the highest bidder for each query. GREEDY has a competitive ratio of 1/2 when considered for all worst-case sequences of queries. In specific input models, GREEDY achieves the optimal competitive ratio of the more complex algorithm of Mehta et. al. That is, GREEDY is $1 - \frac{1}{e}$-competitive.

Mehta and Goel present proofs for the simple GREEDY algorithm in the random input and the i.i.d. models. The random input model allows the set of queries to be arbitrary, while the order of queries is random. In the i.i.d. model, occurrences of queries follow a fixed distribution function. Mehta and Goel proved that GREEDY has competitive ratio $1 - \frac{1}{e}$ in the random permutation input model, as well as in the i.i.d. model.

Goel et. al. modify the online bipartite matching algorithm of Karp et. al. In particular, they fill a hole that was found in a proof of the original randomized algorithm. Then, the results are extended and applied to the Adwords problem.

Although the input to the problem is identical to that considered previously, the revenue of the algorithm is $\sum_{i=1}^{m} min \left\{ B_i, \sum_{q \in Q} bid_{iq} \right\}$. This change is due to the property that a bidder's budget can be exceeded by the last bid he wins. Since bids are small compared to an advertiser's budget, this loss is negligible.

In online bipartite matching, whether or not an item it matched depends on its position in the random permutation of online input. If it is high enough in the permutation, then it is matched. Otherwise, by the time it arrives, its desirable match may already be allocated.

We can consider the permutations of online input from the perspective of an item p that the algorithm receives to match. Permutations can be classified into those in which p remains unmatched, *miss*, and those in which p is matched by the end of the algorithm. There are subclasses for the permutations in which p gets matched, *good* and *bad*, which depend on the structure of the match. In a *good* match, the correct match is available when p arrives. In a *bad* match, the correct match is allocated prior to the arrival of p.

There are several properties of GREEDY that are simple observations in online bipartite matching. The Prefix property states that if two permutations begin with an identical sequence of t items, then the allocation of GREEDY will be the same up to time t. The Monotonicity property describes the effect of moving an item to the front of the permutation: all items that were matched by time t are now matched by time $t + 1$. The Partition property tells us which variable is 1 for each permutation of the input sequence. With these properties, inequalities bound the sizes of *miss*, *bad*, and *good*. A factor revealing LP maximizes the loss of revenue over these inequalities. The LP proves a competitive ratio of $1 - \frac{1}{e}$ in the random input model of online bipartite matching.

There can be several different bids for the same query in the Adwords problem. Not every mismatch can be reversed easily; a tagging procedure is used to generalize the results. Thus, the three basic properties are invariant in Adwords allocation. The tagging method works on permutations of the input.

Consistent Tie-Breaking is another property that is taken for granted in online bipartite matching but needs to be formalized for the Adwords problem.

The monotonicity of the allocation algorithm is an important property. If a query is moved to the front of the permutation, each bidder would spend at least as much money as he did before. This property holds in the GREEDY algorithm, provided we allow a bidder to overspend his budget in the last bid he wins. The overspending is not counted towards revenue. This variant of GREEDY is monotonic and thus easier to analyze.

The tagging procedure is also used to develop the Partition property of the Adwords problem. Thus, the lemmas of the online bipartite matching GREEDY algorithm have direct counterparts in Adwords allocation. The relationship between the sets *miss*, *bad*, and *good* remains the same and the same inequalities hold. After some rearrangement, the same factor revealing LP maximizes the loss of revenue over the inequalities. The same factor of $1 - \frac{1}{e}$ is achieved in the random input model of the Adwords problem.

Although GREEDY was modified to allow overspending, the revenue is almost the same as that of pure GREEDY. This relies on the assumption that bids are small compared to budgets.

The competitive factor of GREEDY in both the random-permutation input model and independent distribution input model (i.i.d.) is exactly $1 - \frac{1}{e}$. This bound is tight.

6 Open Problems

Independent of the Adwords motivation, the problem is a general combinatorial allocation problem. The algorithms in [5] and [1] assume that bids are small compared to an advertiser's budget. Can we generalize their work to problems in which this assumption does not hold?

A search engine must publicize the algorithm that is used for Adwords allocation. How would an advertiser react to this knowledge under the algorithms we have considered? Would they change their bids in any way? From my analysis, it seems that the bidders would devalue the keywords in all their bids.

References

[1] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to adwords. In Shang-Hua Teng, editor, *SODA*, pages 982–991. SIAM, 2008.

[2] Kalyanasundaram and Pruhs. An optimal deterministic algorithm for online b-matching. *TCS: Theoretical Computer Science*, 233, 2000.

[3] B. Kalyanasundaram and K. Pruhs. Online weighted matching. *Journal of Algorithms*, 14(3):478–488, May 1993.

[4] Karp, Vazirani, and Vazirani. An optimal algorithm for on-line bipartite matching. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 1990.

[5] Mehta, Saberi, Vazirani, and Vazirani. Adwords and generalized on-line matching. In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2005.

[6] Mehta, Saberi, Vazirani, and Vazirani. Adwords and generalized online matching. *JACM: Journal of the ACM*, 54, 2007.

[7] Sara Robinson. Computer scientists optimize innovative ad auction. *SIAM News*, 38, April 2005.