International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 06 | June-2016 www.irjet.net p-ISSN: 2395-0072

MULTIPLE-ITEM B-BOUNDED ONLINE AUCTIONING

Harish Kumar Singh[1] Dr.Rajesh Singh[2]

[1]M.Tech Scholor , Dept. of CSE, B. S. A. College of Engg. & Tech., Mathura
[2]A.P., Dept. of CSE B. S. A. College of Engg. & Tech., Mathura
[1]harishsingh200785@gmail.com ,[2]rajesh.singh@bsacet.org

*****

ABSTRACT - *Given the interactive nature of Internet auctioning today, we feel it most appropriate to study auctioning strategies from an online algorithms perspective. That is, algorithms must make immediate decisions based on existing, incomplete information, and are not allowed to delay responses to wait for future offers. This has the additional advantage of obviating the requirement for a closing time which leads to various inefficiencies. Moreover, given that existing auctioning web sites must implement what are essentially algorithmic rules for accepting or rejecting bids, in this paper we focus on algorithmic strategies for sellers. Even so, we restrict our study to strategies that are honest and fair to buyers.*

Key Words: Offline Auctions, Online Auctions, Lower bound, Price and Pack, Optimal offline algorithm.

INTRODUCTION

Auctions are among the oldest forms of economic activity known to mankind. Of late there has been a renewed interest in auctioning as the Internet has provided a forum for economic interaction on an unprecedented scale. A number of web sites have been created for supporting various kinds of auctioning mechanisms. For example, at www.priceline.com, users present bids on commodity items without knowledge of prior bids, and the presented bids must be immediately accepted or rejected by the seller. Alternately, web sites such as www.ebay.com and www.ubid.com allow bidding on small lots of non-commodity items, with deadlines and exposure of existing bids. The rules for bidding vary considerably, in fact, even in how equal bids for multiple lots are resolved. Interestingly, it is a simple exercise to construct bidding sequences that result in suboptimal profits for the seller. For example, existing rules at www.ubid.com allow a $100 bid on 10 of 14 items to beat out two $70 bids on 7 items each. Thus, we feel there could be considerable interest in algorithmic strategies that allow sellers to maximize their profits without compromising fairness.

EXISTING TECHNIQUES AND WORK

Offline scenarios for auctioning, where all bids are collected at one time, such as in sealed bid auctions, have been studied and understood in terms of knapsack problems, for which the algorithms community has produced considerable previous work [12, 8, 9].

The general area of online algorithms [5] studies combinatorial optimization problems where the problem instance is presented interactively over time but decisions regarding the solution must be made immediately. Even though such algorithms can never know the full problem instance until the end of the sequence of updates, it might not even know when the sequence has ended, online algorithms are typically compared to optimal offline algorithms. We say that an online algorithm is c-competitive with respect to an optimal offline algorithm if the solution determined by the online algorithm differs from that of the offline algorithm by at most a factor of c in all cases. The goal, therefore, in online algorithm design is to design algorithms that are c-competitive for small values of c. Often, as will be the case in this chapter, we can prove worst-case lower bounds on the competitive ratio, c, achievable by an online algorithm. Such proofs typically imply an adversary who constructs input sequences that lead online algorithms to make bad choices. In this paper, we restrict our attention to oblivious adversaries, who can have knowledge of the online algorithm we are using, but cannot have access to any random bits that it may use.

In work that is somewhat related to online auctioning, Awerbuch, Azar and Plotkin [3] study online bandwidth allocation for throughput competitive routing in networks. Their approach can be viewed as a kind of bidding strategy for bandwidth. Leonardi and Marchetti Spaccamela [9] generalize the result of Awerbuch et al.

Work for online call control [1, 13, 9] is also related to the problems we consider. In online call control, bandwidth demands made by phone calls must be immediately accepted or rejected based on their utility and on existing phone line usage. In fact, our work uses an adaptation of an algorithmic design pattern developed by Awerbuch et al.

[4] and Lipton [10], which Awerbuch et al. call "classify-and-select." In applying this pattern to an online problem, one must find a way to partition the optimization space into q classes such that, for each class, one can construct a c-competitive algorithm (all using the same value of c). Combining all of these individual algorithms gives an online algorithm with a competitive ratio that is O(cq). Ideally, the individual c-competitive algorithms should be parameterized versions of the same algorithm, and the values c and q should be as small as possible. Indeed, the classify-and-select pattern is best applied to problems that can be shown to require competitive ratios that are Ω(cq) in the worst case against an oblivious adversary.

OUR WORK

1.1 MULTIPLE-ITEM B-BOUNDED ONLINE AUCTIONING

In this section we introduce the multiple-item B-bounded online auctioning problem. We have n instances of the item on sale and the bids which come in for them offer varying benefit per item. Each bid can request any number of items and offer a given benefit for them. The objective is to maximize the profit that can be earned from the sequence of bids with the additional requirement that the seller accept or reject any given bid before considering any future bids, if they exist.

The price density of a bid is defined as the ratio of the benefit offered by the bid to the number of instances of the item that the bid wants to buy. In other words the price density is the average price per item the bidder is willing to pay. The range of possible price densities that can be offered is between 1 and B, inclusively. This restriction is made without loss of generality in any scheme for single-item bidding that has bounded bid magnitude, as we can alternately think of B as the ratio of the highest and lowest bids that can possibly be made on this item. A sequence of bids need not contain the two extreme values, 1 and B, and any bid after the first need not be larger than or equal to the previous bid.

We assume that the algorithm knows the value of B. We discuss at the end of this section how this assumption can be dispensed with. For this problem we propose an algorithm that uses an adaption of a random choice strategy of Awerbuch and Azar [2] together with the "classify and select" technique of [4], where we break the range of possible optimization values into groups of ranges and select sets of bids which are good in a probabilistic sense based on these ranges. Our algorithm is described in Figure 1.1.

Algorithm Price And Pack

- Select i uniformly at random from the integers 0 to log B − 1.
- If i is 0 then set $pd_r = 1$ else set $pd_r = 2^{i-1}$.
- Define a bid as legitimate if it has a price density of at least pd_r
  - Toss a fair coin with two outcomes before any bid comes in.
  - If the coin has landed heads then wait for a legitimate bid on more than n/2 items to come in rejecting all smaller bids and all illegitimate bids.
  - Else keep accepting legitimate bids till there is capacity to satisfy them. Reject all illegitimate bids.

Figure 1.1: Auctioning multiple items with bids of varying benefit.

Theorem 2.2.1 Price And Pack is an O(log B) competitive algorithm for the multiple item B-bounded online auctioning problem.

Proof. Let the optimal offline algorithm OPT achieve profit density p on a given input sequence I. So if the optimal algorithm sells $n' \le n$ items, its total profit is $n'p$. Let j be the largest integer such that $2^j \le 4p/5$. Define $\alpha = 2^j/p$. We say that Price and Pack chooses i correctly, if the chosen value of i equals j. It is easy to see that i is chosen correctly with probability 1/ log B. In that event, bids of price density greater than pα are legitimate while the rest are not. Note that α (2/5, 4/5].

Let I_p be a subset of I, comprising all bids in I which have price density greater than pα.

Lemma 2.2.2 The sum of the revenues obtained by the optimal algorithm running on I_p is no less than $n'p(1 - \alpha)$ where p is the profit density of OPT on I and n' is the number of items it sells.

Proof: Suppose that OPT sells some $n_{lt} \le n'$ instances to bids in $I - I_p$, and let rev_{ge} be the revenue earned by OPT from items which were sold to bids in I_p. Clearly,

$$rev_{ge} + n_{lt} .p\alpha \ge n'p$$

this gives us

$$rev_{ge} \geq n'p - n_{lt}.p\alpha$$

and since $n_{lt} \leq n'$ we get

$$rev_{ge} \geq n'p(1 - \alpha)$$

Since rev_{ge} is the revenue obtained from a subset of the bids in I_p, the result follows.

We consider the following three cases, and show that in each case the expected revenue ofPrice And Pack is at least np /10 log B .

Case 1: There is a bid of size greater than n/2 in I_p.

With probability at least 1/logB, Price And Pack chooses i correctly. With probability 1/2 Price And Pack chooses to wait for a bid of size greater than size n/2.Thus, with probability at least 1/2 log B Price And Pack will accept a bid of size at least n/2 and price density at least αp.So in this case the expected revenue of Price And Pack is at least npα /4 log B. Since the revenue earned by OPT is np, and α > 2/5, in this case Price And Pack is 10 log B competitive with OPT.

Case 2: There is no bid of size greater than n/2 in I_p, and the total number of items demanded by the bids in I_p is more than n/2.

With probability 1/2 Price And Pack will choose to accept bids of any size. If it also chooses i correctly (the probability of which is 1/ log B), it will sell at least n/2 instances, and earn a revenue of at least pα units for every item sold. Thus, with probability 1/2 log B, Price And Pack sells at least n/2 instances to bids whose price densities are no smaller than pα. This means that, in this case, the expected revenue of Price And Pack is at least npα /4 log B> np /10 log B, which makes it 10 log B competitive with OPT.

Case 3: There is no bid of size greater than n/2 in I_p, and taken together the bids in I_p demand no more than n/2 instances.

Again, with probability ½ Price And Pack decides to accept all bids, and with probability 1/ log B, i is chosen correctly. Thus, with probability 1/2 log B our algorithm accepts all bids in Ip, and, by Lemma 2.2.2, earns a revenue no smaller than $n'p(1 - \alpha)$ where n' is the number of items sold by $n'p(1 - \alpha)$ /2 log B≥ $n'p$ /10 log B , which makes it 10 log B competitive with OPT in this case.

An important thing to note is that here the algorithm has to know the range of the input i.e. the algorithm has to be aware of the value of B. It is possible to dispense with this assumption to get a slightly weaker result following [4]. In other words, it is possible to give an $O((\log B)^{1+\epsilon})$ competitive algorithm, for any $\epsilon > 0$, which does not know the value of B beforehand. We do not detail it here because it does not provide any further insight into the problem of auctioning. In the next section we give lower bounds which will show that Price And Pack gives the best possible competitive ratio for the this problem.

1.2 LOWER BOUNDS FOR THE ONLINE AUCTIONING PROBLEM

We consider the version of the online auctioning problem in which there is only one item to be auctioned and the range of possible prices that can be offered for this item is between 1 and B, inclusively. We call this the single item B-bounded online auctioning problem. We give lower bounds for this problem. Upper bounds for this problem are given in [5]. It is easy to see that a lower bound on any algorithm for the single-item problem is a lower bound for the multiple-item problem as well.

If Price And Pack accepts all bids in I_p, it sells at least n/2 instances. If it rejects any bid in I_p, it must not have enough capacity left to satisfy it. But then at least n/2 instances must have been sold, since any bid in I_p in particular the rejected bid is of size no more than n/2.

In this section we first prove that no deterministic algorithm can in the worst case have a competitive ratio better than the maximum for the single-item problem. More precisely, we show that every deterministic algorithm must have a worst-case competitive ratio that is $\Omega(B)$. This lower bound is based on the fact that a seller does not know in advance how many bids will be offered. Even so, we also show that even if the seller knows in advance the number of bids in the input sequence, any deterministic algorithm is limited to a competitive ratio that is $\Omega(\sqrt{B})$ in the worst case.

Theorem 1.2.1 Any deterministic algorithm for the single-item B-bounded auctioning problem has a competitive ratio that is $\Omega(B)$ in the worst case.

Proof: For a given deterministic algorithm A we construct an adversarial input sequence I_A in the following way: Let the first bid in I_A be of benefit 1. If A accepts this bid, then I_A is the sequence {1, B}. In this case, on the sequence I_A, the deterministic algorithm A gets a benefit of 1 unit while the offline optimal algorithm would pick up the second bid thereby earning a benefit of B units. If A does not accept this first bid, then I_A is simply the sequence {1}. In this case A earns 0 units of revenue while the optimal offline algorithm would accept the bid of benefit 1.

Of course, B is the worst competitive ratio that is possible for this problem, so this theorem implies a rather harsh constraint on deterministic algorithms. Admittedly, the above proof used the fact, perhaps unfairly, that the seller does not know in advance the number of bids that will be received. Nevertheless, as we show in the following theorem, even if the number of bids is known in advance, one cannot perform much better.

Theorem 1.2.2 Any deterministic algorithm for the single-item B-bounded online auctioning problem, where the number of bids is known in advance, has a competitive ratio that is $\Omega(\sqrt{B})$ in the worst case.

Proof: Consider the input sequence $I_{base} = \{1, 2, 4, \ldots 2^i \ldots B/2, B\}$. For any deterministic algorithm A we construct our adversarial sequence I_A based on what A does with I_{base}.

We recall here that since we are considering the single-item problem, any deterministic algorithm essentially picks at most one of the bids in the input sequence.

Suppose A accepts some bid $2^i \leq \sqrt{B}$. Then we choose I_A to be the same as I_{base}. In this case A's benefit is less than $\sqrt{B}$, whereas an optimal offline algorithm would earn B units thereby making A an $\Omega(\sqrt{B})$ competitive algorithm.

If A accepts some bid $2i > \sqrt{B}$, on the other hand, then we choose IA to be $\{1, 2, 4, \ldots 2i{-}1, 1, 1, \ldots\}$, i.e., we stop increasing the sequence just before A accepts and then pad the rest of the sequence with bids of benefit 1. 3 This way A can get no more than 1 unit of benefit while the optimal offline algorithm gets $2i{-}1$ which we know is at least $\sqrt{B}$.

If A accepts none of the bids in I_{base} then it is not a competitive algorithm at all (i.e. it earns 0 revenue while the optimal offline algorithm earns B units) and so we need not worry about it at all.

It is easy to see that the deterministic algorithm that either picks up a bid of benefit at least $\sqrt{B}$ or, if it does not find such a bid, picks up the last bid, whatever it may be, succeeds in achieving a competitive ratio of $O(\sqrt{B})$.

Theorem 1.2.1 tells us that no deterministic algorithm can effectively compete with an oblivious adversary in the worst case, if the number of bids is not known in advance. Indeed, although the proof used a sequence that consisted of either one bid or two, the proof can easily be extended to any sequence that is either of length n or n + 1. This bleak outlook for deterministic algorithms is not improved much by knowing the number of bids to expect, however, as shown in Theorem 1.2.2.

Furthermore we show that even randomization does not help us too much. We can use Yao's principle [15] to show that no randomized algorithm can be more competitive against an oblivious adversary than Price and Pack.

1.3 EXPERIMENTAL RESULTS

In order to give an idea of the efficacy of Price And Pack we present the results of simulated auctions which use this algorithm.

The input sequences were generated by selecting each bid from a given probability distribution. The three distributions used were: Normal, Poisson and Uniform. Both the number of items being bid on and the price density offered by the bid were chosen from the same distribution.

We chose three different combinations of n and B and generated 100 input sequences for each combination. To get a good approximation to the average benefit of Price And Pack we ran the algorithm 1000 times on each instance and averaged the benefit over all these runs.

We determined a lower bound on the amount of revenue obtained by our algorithm compared to the maximum possible revenue. To do this we implemented an offline algorithm which has been shown to be a 2 approximation [7]. By dividing the revenue obtained by Price And Pack by 2 times the revenue obtained by the offline algorithm we were able to provide a number which is effectively a lower bound on the actual ratio.

The numbers in Table 1.1 show that in practice Price And Pack performs quite well compared to the optimal offline algorithm and significantly better than the bound of O(log B) would suggest. We see that in the two distributions which tend to cluster sample points near the mean, i.e. Normal and Poisson, the algorithm does especially well. However these distributions provide fairly regular input instances. The real power of Price And Pack is on view when the input instances have widely varying bids.

(n, B)	Distribution	Expected ratio (1/ log B)	Ratio
(50,1024)	Uniform	0.1	0.31
	Normal		0.69
	Poisson		0.61
(2000,1024)	Uniform	0.1	0.34
	Normal		0.62
	Poisson		0.7
(2000,2048)	Uniform	0.09	0.34