Seller-Focused Algorithms for Online Auctioning

Amitabha Bagchi[1], Amitabh Chaudhary[1], Rahul Garg[2], Michael T. Goodrich[1], and Vijay Kumar[3⋆]

[1] Dept of Computer Science, Johns Hopkins University, 3400 N Charles St, Baltimore MD 21218, USA
[2] IBM India Research Lab, Hauz Khas, New Delhi 110016, India
[3] Amazon.com, 605 5th Avenue South, Seattle, WA 98104, USA

Abstract. In this paper we provide an algorithmic approach to the study of online auctioning. From the perspective of the seller we formalize the auctioning problem as that of designing an algorithmic strategy that fairly maximizes the revenue earned by selling n identical items to bidders who submit bids online. We give a randomized online algorithm that is $O(\log B)$-competitive against an oblivious adversary, where the bid values vary between 1 and B per item. We show that this algorithm is optimal in the worst-case and that it performs significantly better than any worst-case bounds achievable via deterministic strategies. Additionally we present experimental evidence to show that our algorithm outperforms conventional heuristic methods in practice. And finally we explore ways of modifying the conventional model of online algorithms to improve competitiveness of other types of auctioning scenarios while still maintaining fairness.

1 Introduction.

Although auctions are among the oldest forms of economic activity known to mankind, there has been a renewed interest in auctioning as the Internet has provided a forum for economic interaction on an unprecedented scale. Indeed, 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 for 10 of 14 items to beat out two \$70 bids for 7 items each. Thus, we feel there could be considerable interest in algorithmic strategies that allow sellers to maximize their profits without compromising fairness.

Since Internet auctioning requires a great deal of trust, notions of fairness and legitimacy are fundamental properties for an auction [22]. Indeed, there is considerable previous mathematical treatments of auctioning that classify various types of auctions,

⋆ A significant portion of this work was done while this author was at IBM's India Research Lab.

together with evolving criteria of fairness (e.g., see [23, 19, 9]). Still, the algorithmic aspects of auctioning have been largely neglected.

The Topic of Interest. 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.

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. For example, we would consider as unacceptable a strategy that uses a fictitious external bidder that causes a real bidder to offer more than he would had there been less bidding competition. Besides being unethical, dishonest or unfair strategies are ultimately detrimental to any Internet auction house anyway, since their discovery drives away bidders.

Previous Related 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 [20, 12, 13]. We are not aware of much previous work on online auctioning strategies, however.

The general area of online algorithms [7] 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 (whose arrival itself might not even be known to the online algorithm), 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.[1] 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 paper, 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, but differs from our study, since in bandwidth allocation the problem of communication path determination is at least as difficult as that of bandwidth capacity management. Leonardi and Marchetti-Spaccamela ([14]) generalize the result of Awerbuch *et al.*, but in a way that is also not directly applicable to online auctioning, since, again, there is no notion of path determination in online auctioning.

[1] As a matter of convention we can never have a c-competitive algorithm for $c < 1$ (such algorithms would instead be called $1/c$-competitive)

Work for online call control [1, 4, 15] 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 [15], 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 $\Omega(cq)$ in the worst case against an oblivious adversary.

Our Results. We consider several algorithmic issues regarding online auctioning, from the viewpoint of the seller, in this paper. We begin, in Section 2, by defining the *multiple-item B-bounded online auctioning problem* in which bidders bid on multiple instances of a single item with each bidder allowed to bid for as many items as he or she wants to. We present an online algorithm for this problem that is $O(\log B)$-competitive with an oblivious adversary. The upper bound result presented in this sections is based on adaptations of the classify-and-select design pattern [4] to the specific problem of online auctioning.

In Section 3 we show that it is not possible for any deterministic algorithm to provide a satisfactory competitive ratio for this problem. Moreover, we show that the algorithm we give in Section 2 is "optimal" in the sense that no randomized algorithm can achieve a competitive ratio of $o(\log B)$. To do this we derive lower bounds, based on novel applications of Yao's "randomness shifting" technique [24] , that show the competitive ratios for our algorithm is worst-case optimal.

In order to show that our algorithm performs well in practice we undertook a number of experiments. The results, detailed in Section 4, demonstrate that our algorithm handles different types of input sequences with ease and is vastly superior to other online strategies in the difficult case where the bids vary greatly in size and benefit.

Finally, in Section 5 we discuss a possible modification of our model with an eye towards making it more flexible as regards its online nature so as to gain in terms of competitiveness. In particular, we allow the algorithm to "buffer" a certain number of bids before making a decision. We show that by buffering only $O(\log B)$ bids it is possible to be c-competitive with an oblivious adversary for the case in which we are selling a single item, for a constant c.

2 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.

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 for 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.

Fig. 1. Price_And_Pack: Auctioning multiple items with bids of varying benefit.

Theorem 1. Price_And_Pack *is an* $O(\log B)$ *competitive algorithm for the multiple item* B*-bounded online auctioning problem.*

The proof of this theorem is in Appendix A. ■

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.

3 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 [7]. 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.

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 2. *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 3. *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 $I_{base} = \{1, 2, 4, \dots 2^i \dots B/2, B\}$. 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 $2^i > \sqrt{B}$, on the other hand, then we choose I_A to be $\{1, 2, 4, \ldots 2^{i-1}, 1, 1, \ldots\}$, i.e., we stop increasing the sequence just before A accepts and then pad the rest of the sequence with bids of benefit 1. [2] This way A can get no more than 1 unit of benefit while the optimal offline algorithm gets 2^{i-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 2 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 algorithm is not improved much by knowing the number of bids to expect, however, as shown in Theorem 3.

Furthermore we show that even randomization does not help us too much. We can use Yao's principle [24] to show that no randomized algorithm can be more competitive against an oblivious adversary than *Sell_One*.

Theorem 4. *Any randomized algorithm for the single-item B-bounded online auctioning problem is $\Omega(\log B)$-competitive in the worst case.*

The proof is sketched in Appendix B. ■

4 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 for 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

[2] Bids are not always increasing. For a single item there is no issue at all if the bids are always increasing and the number of bids is known. Just wait for the last bid.

algorithm which has been shown to be a 2 approximation [11]. 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.

(n, B)	Distribution	Expected ratio ($1/\log B$)	Ratio
(50, 1024)	Uniform Normal Poisson	.1	.31 .69 .61
(2000, 1024)	Uniform Normal Poisson	.1	.34 .62 .7
(2000, 2048)	Uniform Normal Poisson	.09	.34 .61 .69

Fig. 2. Price_And_Pack v/s the optimal offline algorithm.

The numbers in Figure 2 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.

To demonstrate this we compared the performance of a simple *Greedy* heuristic with the performance of *Price_And_Pack*. *Greedy* simply accepts bids while it has the capacity to do so. In Figure 3 we present the results in terms of the percentage extra revenue *Price_And_Pack* is able to earn over the *Greedy* heuristic.

Distribution	(n, B)	Average %
Uniform	(50, 1024) (2000, 1024) (2000, 2048)	25 28.5 27.1
Normal	(50, 1024) (2000, 1024) (2000, 2048)	0.5 0.7 0.5
Poisson	(50, 1024) (2000, 1024) (2000, 2048)	1.4 0.1 0.3

Fig. 3. Price_And_Pack over **Greedy**: % advantage.

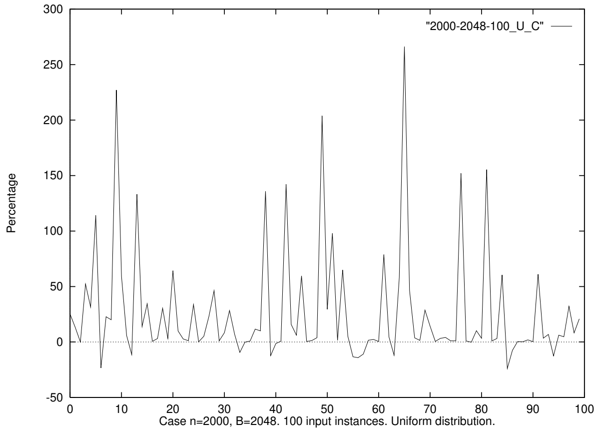

Fig. 4. Price_And_Pack over **Greedy**: % advantage in 100 individual runs.

We see that when the bids are comparable to each other (i.e. when they are generated by the Normal or Poisson distribution) then *Price_And_Pack* does not do significantly better than *Greedy* but when the bids vary widely in size (i.e. when they are generated by the Poisson distribution) then *Price_And_Pack* definitely outperforms *Greedy*.

In Figure 4 we graph the percentage extra revenue earned by *Price_And_Pack* in 100 different input instances for a given choice of n and B. It is clear from the graph that *Price_And_Pack* consistently outperforms *Greedy*.

5 Improving Competitiveness by Giving Intermediate Information.

In this section we look at a way of modifying the auctioning model to improve competitiveness. Taxonomies of auctions (for eg. [23], [19]) have classified auctions along three broad categories: bid types, clearance mechanisms and intermediate information policies. We look at this lattermost classification to help us improve competitiveness.

In the preceding sections we saw that in the conventional online model, where every bid has to be accepted or rejected before the next bid comes in, we are limited to a competitive ratio of $\Omega(\log B)$. However it is possible to do better if we relax the online model slightly. In the model under consideration so far every bid has to be accepted or rejected immediately, or, more precisely, before the next bid comes in. However in real life auctions this is not always the case. Most auctions do release some intermediate information. For example in the outcry type of auction the current highest bid is announced. This amounts to informing those who bid at that level that their bid is still under consideration, although it might yet be beaten out by a better bid.

The problem with the outcry auction is that, in the case of a monotonically increasing sequence of bids, each bid is asked to hold on and then rejected i.e. $O(B)$ bids are made to wait till a new bid comes in before being rejected. This is clearly unacceptable since from the point of view of a bidder an intermediate notification that the bid is still under consideration is tantamount to saying that this bid has a reasonable chance of success. However if the bidder knows that $O(B)$ bids could be asked to hold then he might not consider this chance of success reasonable.

So, the model we propose is that only a certain small number of bids can be asked to wait without a definite notification of acceptance or rejection. We can think of these bids being buffered in a buffer which will need to contain only one item at a time and will not be allowed to hold more than a certain small number of items in the course of the auction. We call this structure a *k-limited access buffer* or a k-LAB. We denote the highest bid held in the LAB by $H(LAB)$.

That this relaxation is useful becomes immediately evident when we consider that a $\log B$-LAB allows us to become constant competitive deterministically with the optimal algorithm in the case where we want to sell one item and we know the number of bids. We give the algorithm for this in Figure 5. We have to view this in light of the fact that in Theorem 3 we showed that in a purely online setting it is not possible to do better than $\Omega(\sqrt{B})$ deterministically for this problem.

Algorithm LAB_Sell_One_N

- $LAB \leftarrow b_0$
- For each bid b_i for i going from 1 to N do
  - if $b_i > 2.H(LAB)$ then $LAB \leftarrow b_i$ else reject b_i
- Accept the bid in the LAB.

Fig. 5. LAB_Sell_One_N: Auctioning a single item with a buffer when the number of bids is known.

Theorem 5. LAB_Sell_One_N *is* $\frac{1}{2}$ *competitive with the optimal.*

Proof: It is quite easy to see that the highest bid b_{off} which is the benefit of the offline algorithm would not be put in the buffer only if a bid which was at least half its benefit were already in there. Since a bid of at least half its benefit is already in the online algorithm's buffer therefore its benefit will be at least half of the online's. ∎

Acknowledgements. The authors would like to thank Leslie Hall and two anonymous referees.

References

1. R. Adler and Y. Azar. Beating the logarithmic lower bound: randomized preemptive disjoint paths and call control algorithms. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–10, 1999.
2. B. Awerbuch and Y. Azar. Blindly competitive algorithms for pricing and bidding. Manuscript.

3. B. Awerbuch, Y Azar, and S. Plotkin. Throughput competitive on-line routing. In *Proceedings of the 34th Annual Symposium on Foundations of Computer Science*, pages 32–40. IEEE, 3–5 November 1993.
4. B. Awerbuch, Y. Bartal, A. Fiat, and A. Rosen. Competitive non-preemptive call-control. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, 23-25 Jan 1994.
5. A. Bagchi, A. Chaudhary, R. Garg, M. T. Goodrich, and V. Kumar. Online algorithms for auctioning problems. Technical report, IBM India Research Lab, 2000.
6. S. Ben-David and A. Borodin. A new measure for the study of on-line algorithms. *Algorithmica*, 11:73–91, 1994.
7. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
8. D. Dooly, S. Goldman, and S. Scott. Tcp dynamic acknowledgment delay: Theory and practice. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing,*, pages 289–298, May 1998.
9. R. Engelbrecht-Wiggans, M. Shubik, and R. M. Stark, editors. *Auctions, Bidding, and Contracting: Uses and Theory*. New York University Press, 1983.
10. E. F. Grove. Online bin packing with lookahead. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 430–436, 1995.
11. Dorit S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems.* PWS Publishing Company, Boston, MA., 1997.
12. O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and subset sum problems. *Journal of the ACM*, 22:463–468, 1975.
13. E. L. Lawler. Fast approximation algorithms for knapsack problems. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, pages 206–213, 1977.
14. S. Leonardi and A. Marchetti-Spaccamela. On-line resource management with applications to routing and scheduling. In *Proceedings of the 22nd International Colloquium on Automata, Languages and Programming*, pages 303–314, 1995.
15. R. J. Lipton and A. Tomkins. Online interval scheduling. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 302–11, 23-25 Jan 1994.
16. G. Lueker. Average-case analysis of off-line and on-line knapsack problems. *Journal of Algorithms*, 29(2):277–305, November 1998.
17. A. Marchetti-Spaccamela and C. Vercellis. Stochastic on-line knapsack problems. *Mathematical Programming*, 68(1):73–104, Jan 1995.
18. R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
19. M. H. Rothkopf, A Pekec, and R. M. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.
20. S. Sahni. Approximation algorithms for the 0/1 knapsack problem. *Journal of the ACM*, 22:115–124, 1975.
21. T. Sandholm. An algorithm for optimal winner determination in combinational auctions. Technical Report WUCS-99-01, Department of Computer Science, Washington University, January 1999.
22. Charles W. Smith. *Auctions: The Social Construction of Value*. The Free Press, 1989.
23. P. R. Wurman, M. P. Wellman, and W. E. Walsh. A parametrization of the auction design space. *Games and Economic Behaviour*, To appear.
24. A. C-C. Yao. Probabilistic computations: Towards a unified measure of complexity. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977.
25. T. Yeh, C. Kuo, C. Lei, and H. Yen. Competitive analysis of on-line disk scheduling. *Theory of Computing Systems*, 31:491–506, 1998.

A Proof of Theorem 1

Let the optimal offline algorithm OPT achieve profit density p on a given input sequence I. So if the optimal algorithm sells $n' \leq n$ items, its total profit is $n'p$. Let j be the largest integer such that $2^j \leq 4p/5$. Define $\alpha = \frac{2^j}{p}$. 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 $p\alpha$ are legitimate while the rest are not. Note that $\alpha \in (2/5, 4/5]$.

Let I_p be a subset of I, comprising all bids in I which have price density greater than $p\alpha$.

Lemma 1. *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 $n_{lt} \leq n'$ 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,

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

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. ■

Proof of Theorem 1:

We consider the following three cases, and show that in each case the expected revenue of *Price_And_Pack* is at least $\frac{np}{10\log B}$.

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

With probability at least $1/\log B$, *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 $\frac{1}{2\log B}$, *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 $\frac{np\alpha}{4\log B}$. Since the revenue earned by OPT is np, and $\alpha > 2/5$, 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 [3], and earn a revenue of at least $p\alpha$ 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 $p\alpha$. This means that, in this case, the expected revenue of *Price_And_Pack* is at least $\frac{np\alpha}{4\log B} > \frac{np}{10\log B}$, 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.

[3] 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 $n/2$.

Again, with probability $1/2$ *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 I_p, and, by Lemma 1, earns a revenue no smaller than $n'p(1-\alpha)$ where n' is the number of items sold by OPT. So its expected revenue is at least $\frac{n'p(1-\alpha)}{2\log B} \geq \frac{n'p}{10\log B}$, which makes it $10\log B$ competitive with OPT in this case. ■

B Proof Sketch for Theorem 4

We use Yao's Principle [24, 18] to show a lower bound for all randomized algorithms. To do this we give a probability distribution over the input space and determine the expected benefit of the best possible deterministic algorithm on this probabilistic input. The competitiveness of this expectation against the expected benefit of the optimal offline algorithm for this probabilistically distributed input will be, by Yao's Principle, a lower bound on the competitiveness of any randomized algorithm for this problem. Due to lack of space we simply describe the probability distribution on the inputs here. The entire proof is available in [5].

Consider the following input sequence which we will be calling the base sequence or I_{base}: $\{1, 2, 4, \ldots B/2, B\}$. Our set of input sequences will be derived from I_{base} by truncating it at a given point and substituting bids of revenue 1 for the tail of the input sequence. All other inputs occur with probability 0. The set of inputs, $\mathcal{I} = \{I_1, I_2 \ldots I_{\log B}\} \cup \{I_f\}$ and associated probabilities are described as:

- $I_i = \{1, 2, 4, \ldots 2^i, 1 \ldots 1\}$ occurs with probability $P_i = \frac{1}{2^{i+1}}$. Each I_i has $\log B + 1$ bids.
- $I_f = \{1\}$ occurs with probability $P_f = \frac{B+1}{2B}$.

■