How well can Priceline sell airline tickets ?

Nikhil Bansal[1] Ning Chen[2*] Neva Cherniavsky[2] Atri Rudra[2†]
Baruch Schieber[1] Maxim Sviridenko[1]

[1] IBM T.J. Watson Research Center
Yorktown Heights, NY
{nikhil,sbar,sviri}@us.ibm.com

[2] Department of Computer Science and Engineering
University of Washington
Seattle, WA
{ning,nchernia,atri}@cs.washington.edu

TECHNICAL REPORT UW-CSE-TR-2005-10-01

Abstract

We consider the following simple model for how Priceline sets airline prices. Assume Priceline wants to sell tickets for a single flight (which are assumed to have unlimited supply) over (discrete) time. Each bidder submits a triple (s, e, b) where s is the time when the bidder will arrive, e is the time when the bidder will leave and b is the maximum amount the bidder is willing to pay for a ticket (every bidder is interested in only one ticket). On each day t, Priceline sets a price $p(t)$ and every bidder buys a ticket on the first day t (at price $p(t)$) such that $p(t)$ is less than or equal to its bid value and t is between its arrival and departure time. The goal for the Priceline is to set prices so as to maximize its revenue. We also consider a related model studied by Guruswami et al. [1], where each bidder buys the ticket at the minimum price $p(t')$ (provided it is at most b) where t' is between its arrival and departure time.

We study the above optimization problems in both the offline and online settings. We show that the offline algorithm can be solved exactly in polynomial time. We also give upper and lower bounds for online algorithms (both deterministic and randomized) for this problem.

1 Introduction

We consider the following model in this note. There is an unlimited supply of a single item. Every day $t = 1, 2, \ldots$ the auctioneer sets a price $p(t)$. Each bidder i has an input (s_i, e_i, b_i) where s_i and e_i are the arrival and departure days and b_i is the maximum amount the buyer is willing to pay for a copy of the item. We will refer to the tuple (s_i, e_i, b_i) as the *bid* of buyer i and the quantity b_i

*Supported in part by NSF CCR-0105406.

†This work was done when the author was a summer intern at IBM TJ Watson Research Center, Yorktown Heights, NY.

	Deterministic		Randomized	
	Upper Bound	Lower Bound	Upper Bound	Lower Bound
PL-model	$O(\log h)^{\dagger}$	$\Omega(\sqrt{\log h})$	$O(\log\log h)$	$\Omega(\sqrt{\frac{\log\log h}{\log\log\log h}})$
EF-model	$O(h)^{\dagger}$	$\Omega(h)$	$O(\log h)^{\dagger}$	$\Omega(\sqrt{\frac{\log h}{\log\log h}})$

Table 1: The upper bounds with † are either well known and/or trivial.

as the *bid value*. A buyer buys a copy of the item on the first day $s_i \leq t \leq e_i$ such that $p(t) \leq b_i$. The goal of the auctioneer is to set the prices $\{p(t)\}$ so as to maximize her revenue. We call this model PL-model (where PL stands for Priceline).

We also study the model considered by Guruswami et al. [1]. The only place their model differs from PL-model is that the buyer buys the item at the price $\min_{t\in[s_i,e_i]} p(t)$ (provided of course that this price is less than b_i). Thus, the pricing is Envy-Free [1]– we call this model EF-model.

We also consider both the offline and online models of bidder arrival. In the offline model, the auctioneer receives all the bidders' inputs on day 1. In the online model, on day $t = 1, 2, \ldots$, the auctioneer only knows about the bids of the buyers i for whom $s_i \leq t$.

In the offline model we will aim for polynomial time algorithm to compute the prices for each day such that the optimal revenue is generated. For the online case, we will evaluate algorithms by the standard notion of competitive ratio.

1.1 Previous Work

Guruswami et al. give a polynomial time algorithm to compute the optimal set of prices for the offline version of EF-model in [1].

1.2 Notations

We will use h to denote the ratio of the maximum bid value to the minimum bid value. For the rest of the paper we will assume that the bid values are in the set $\{1, 2, \cdots, h\}$. The total number of bidders will be denoted by n. For every bidder i, the quantity $e_i - s_i$ will sometimes be referred to as the *bid length* of bid i. We will say that a bid i is *alive* at time t if $t \in [s_i, e_i]$ and it has not bought a copy of the item by day $t-1$.

For any set of bids B, $OPT(B)$ denotes the optimal revenue obtainable from B. OPT will denote the optimal revenue from the set of all input bids.

1.3 Our Results

In Section 2, we present a polynomial time algorithm to compute the optimal set of prices for the offline version of the PL-model.

Table 1 tabulates our results and previously knows results for online algorithms.

2 Optimal offline algorithm for PL-model

We have the following result.

Theorem 1 *The optimal set of prices for the offline version of* PL-model *can be computed in polynomial time.*

Proof: We will give a dynamic program to compute the optimal revenue (the set of prices will be a by-product). Let the distinct bid values be denoted by $b_1 \geq b_2 \geq \cdots \geq b_L$ for some $L \leq n$. For any pair of days $s \leq e$, $\ell \in \{0, 1, 2, \cdots, n\}$ and $k \in \{1, 2, \cdots, L\}$ we will use $A_k(s, e, \ell)$ to denote the optimal revenue that can be generated from the set of bids i such that $b_i \geq p_k$, $s_i \in [s, e]$, $\min_{t\in[s,e]} p(t) \geq p_k$ and ℓ bids with bid value at least p_k are still alive on day $e+1$. We also define $C_k(s, e)$ to be the optimal revenue obtainable from the set of bids i such that $b_i \geq p_k$, $s_i \in [s, e]$, $\min_{t\in[s,e-1]} p(t) > p_k$ and $p(e) = p_k$. That is, $C_k(\cdot, \cdot)$ is like $A_k(\cdot, \cdot, \cdot)$ except that the price p_k is used on the last day.

We will use $n^k_{s,t}$ to denote the number of bids i with $s_i \in [s, t]$, $e_i \geq t$ and $b_i = p_k$ and we will use $m^k_{s,t}$ to denote the number of bids i with $s_i \in [s, t]$, $e_i \geq t + 1$ and $b_i = p_k$.

We now spell out the recurrence relation for $A_k(s, e, \ell)$ (assuming $\ell > 0$).

$$A_k(s, e, \ell) = \max\left(A_{k-1}(s, e, \ell - m^k_{s,e}), \max_{t'\in[s,e-1]} \left(C_k(s, t') + A_k(t' + 1, e, \ell)\right)\right)$$

Note that for the optimal revenue $A_k(s, e, \ell)$ there are two options– either only use prices greater than or equal to p_{k-1} or use the price p_k somewhere in the time interval $[s, e]$. The first case is captured by the term $A_{k-1}(s, e, \ell - m^k_{s,e})$– we subtract out $m^k_{s,e}$ from ℓ because by definition the last argument in $A_{k-1}(\cdot, \cdot, \cdot)$ is the number of bids with value greater than p_{k-1} that are still alive on day $e + 1$. In the second case when the price p_k is used, let t' be the first time it is used. This implies that for days in $[s, t'-1]$ the price is at least p_{k-1}. Then by definition, the revenue obtained on the first t' days is $C_k(s, t')$. Note that any bid with value at least p_k that was alive on day t' cannot be alive on day $t'+1$. This implies that the optimal revenue obtainable from days $[t'+1, e]$ such that ℓ bids with bid value greater than p_k are alive on day $e + 1$ is $A_k(t' + 1, e, \ell)$. Of course, for the optimal revenue $A_k(s, e, \ell)$ one has to pick the best possible value of t'. This is obtained by the expression $\max_{t'\in[s,e-1]}(C_k(s, t') + A_k(t' + 1, e, \ell))$.

The reasoning in the above paragraph also explains the following recurrence relation:

$$A_k(s, e, 0) = \max\left(A_{k-1}(s, e, 0), \max_{t'\in[s,e-1]} \left(C_k(s, t') + A_k(t' + 1, e, 0)\right), C_k(s, e)\right)$$

We now give the recurrence relation for $C_k(s, e)$. Note that in this case the minimum price used in the time range $(s, e-1)$ is at least p_{k-1}. If there are ℓ' many bids with value greater than p_{k-1} that are alive on day e, then the maximum revenue obtainable from the days $(s, e - 1)$, by definition, is $A_{k-1}(s, e-1, \ell')$. Further, on day e, $\ell' + n^k_{s,e}$ copies of items are sold at price p_k. Finally optimizing over the choice of ℓ', we get

$$C_k(s, e) = \max_{\ell'\in\{0,1,\cdots,n\}} \left(A_{k-1}(s, e - 1, \ell') + (\ell' + n^k_{s,e})p_k\right)$$

The base cases of the recurrences are pretty simple. For any $s \leq e$ and ℓ

$$A_0(s, e, \ell) = 0$$

$$A_1(s, s, \ell) = 0 \text{ if } \ell \neq 0$$

$$C_1(s,e) = n^1_{s,e} \cdot p_1$$

We are interested in the quantity $A_L(1, \max_{i=1..n} e_i, 0)$. The optimality of the above follows from considering the prices set and the days in non-increasing order. Further, it is easy to see that the prices set by the optimal algorithm have to be among the bid values.

We finally need to show that the dynamic program runs in polynomial time. The number of days considered in the above recurrence relations is $\max_{i=1}^n e_i$ which need not be polynomial in n. However, one can assume w.l.o.g. that for all i, $e_i \leq n$. This is because for every t when $p(t) < \infty$, at least one buyer buys a copy of the item. Further, for all days t such that $t \notin (s_i, e_i)$ for every $i = 1, \ldots, n$, one can set $p(t) = \infty$. Thus, the maximum number of days that needs to be considered by the dynamic program is at most n^2. Since there are at most n different price levels that are considered by the dynamic program, at most n^5 entries need to be considered for $A_k(\cdot,\cdot,\cdot)$ and at most n^4 entries for $C_k(\cdot,\cdot)$. Further, for each level k, only entries in the level $k-1$ need to be accessed. Thus, the above dynamic program runs in polynomial time. ■

3 Online algorithms

For this section, we will assume that all the bid values are powers of 2, that is, they come from the set $\{1, 2, 4, \cdots, h/2, h\}$. In particular there are $\log h + 1$ distinct bid values. This changes the optimal revenue by a factor of 2, which does not affect the results in this section as the competitive ratios are all super-constant.

3.1 Trivial Algorithms

3.1.1 $O(h)$-competitive deterministic online algorithm

Consider the following simple pricing scheme: $p(t) = 1$ for all t. Call this pricing scheme Algo-Price-At-One. We have the following trivial observation.

Proposition 1 Algo-Price-At-One *is an $O(h)$-competitive algorithm.*

Proof: As all bid values lie in $[1, h]$, the revenue obtained from bidder i is at least b_i/h. Thus, the total revenue generated by Algo-Price-At-One is at least $\frac{1}{h} \cdot \sum_{i=1}^n b_i$. The proof is complete by observing that $\sum_{i=1}^n b_i$ is an upper bound on OPT. ■

Note that the above result holds for both PL-model and EF-model.

3.1.2 $O(\log h)$-competitive randomized online algorithm

Consider the following scheme that we call Algo-Stick-At-One-Level. Uniformly at random pick a price level from $\{1, 2, 4, \cdots, h\}$ and set $p(t)$ to be that price for every time t. We have the following observation.

Proposition 2 Algo-Stick-At-One-Level *is an $O(\log h)$-competitive algorithm.*

Proof: If Algo-Stick-At-One-Level chooses price level 2^j then all the bids with bid value at least 2^j buy the item at price 2^j. Thus, the revenue generated is at least

$$\sum_{j=0}^{\log h} \frac{1}{\log h} \sum_{i: b_i = 2^j} b_i = \frac{1}{\log h} \sum_{i=1}^n b_i.$$

The proof is complete by noting that $\sum_{i=1}^{n} b_i$ is an upper bound on OPT. ■

Note that the above result holds for both PL-model and EF-model.

3.1.3 $O(\log h)$-competitive deterministic online algorithm

Consider the following algorithm that we call Algo-Max-Price. For any day t, let B_t denote the set of bids that are still alive on that day. Algo-Max-Price sets $p(t) = \arg\max_{p\in\{1,2,4,\cdots,h/2,h\}} p \cdot |\{b_i | b_i \geq p \wedge i \in B_t\}|$.

We use a standard argument to show the following bound:

Lemma 1 Algo-Max-Price *is a $O(\log h)$-competitive algorithm.*

Proof: Divide the range of prices into intervals $[2^i, 2^{i+1})$ for $i = 0, 1, \cdots, \log h$. Then by the pigeon-hole principle, there exists a level i^* such that

$$\sum_{i\in B_t \wedge b_i \in [2^{i^*}, 2^{i^*+1})} b_i \geq \frac{1}{\log h} \sum_{i\in B_t} b_i.$$

By the choice of $p(t)$, we have

$$\sum_{i\in B_t \wedge b_i \geq p(t)} b_i \geq \sum_{i\in B_t \wedge b_i \in [2^{i^*}, 2^{i^*+1})} b_i.$$

The lemma follows from summing the above relation for all days and recalling that $\sum_{i=1}^{n} b_i$ is an upper bound on OPT. ■

We remark that the analysis of the competitive ratio of the algorithm only works for the PL-model.

3.2 $O(\log\log h)$-competitive algorithm for PL-model

In this section we give a randomized $O(\log\log h)$-competitive algorithm for PL-model. The following lemma lies at heart of this result.

Lemma 2 *Let k be a fixed integer, and suppose the instance is such that the duration of every bid lies between $2k$ and $4k$. If k is known in advance, then there is a randomized algorithm (which we call* Alg(k)*) that is $O(1)$-competitive.*

Proof: We divide time into intervals of size k. In particular, for $i \geq 1$, let T_i denote the interval $[(i-1)k+1, ik]$. Let $V_i(j)$ denote the total value of the bids that have bid value 2^j and that arrive during T_i. Let $j_1(i), j_2(i), \ldots, j_k(i)$ be the k price levels with the k highest values of $V_j(i)$. We assume that they are indexed such that $j_1(i) > j_2(i) > \ldots > j_k(i)$. Let $\mathcal{V}(i)$ denote the set of these k indices $j_1(i), \ldots, j_k(i)$. Finally, let $R(i)$ denote the value $V_{j_1}(i) + V_{j_2}(i) + \ldots + V_{j_k}(i)$.

Consider the following algorithm that we call Alg$_{even}(k)$. For $i = 2, 4, 6, \ldots,$ during T_i it sets the prices from the set $\mathcal{V}(i-1)$ in decreasing order of the price levels. That is, on the l^{th} day of interval T_i (i.e. day $(i-1)k + l$), it sets the price 2^{j_l}. On the other days during $T_1, T_3, T_5, \ldots$, the prices are set to infinity. Note that Alg$_{even}(k)$ is a well-defined online algorithm, as $\mathcal{V}(i-1)$ is known at the start of T_i. Also, as each bid has duration at least $2k$, every T_i has length k and as the prices during T_{i-1} are set to infinity, the bids that arrive during T_{i-1} are all alive at the start of T_i. Finally, as the prices set during T_i are in a decreasing order, it follows that the algorithm collects a profit of at least $R(i-1)$ during T_i. Thus the total profit of this algorithm is at least $R(1)+R(3)+\ldots$.

Analogously, we define the algorithm $\mathsf{Alg}_{odd}(k)$ that sets infinite prices during $T_2, T_4, \ldots$ and for odd i, sets prices in $\mathcal{V}(i-1)$ during T_i. It is easy to see that the total profit of $\mathsf{Alg}_{odd}(k)$ is at least $R(2)+R(4)+\ldots$.

Our randomized online algorithm simply tosses one coin in the beginning and either executes $\mathsf{Alg}_{odd}(k)$ or $\mathsf{Alg}_{even}(k)$. We call this algorithm $\mathsf{Alg}(k)$. Clearly, the expected profit of this algorithm is at least $1/2\sum_{i\geq 1} R(i)$.

We now show that any offline algorithm can get a total profit of at most $\sum_{i\geq 1} 10R(i)$. Consider the period T_i for some $i \geq 1$. Since each bid has duration at most $4k$, the profit obtained during T_i can only be due to bids that arrived during T_{i-4} or later. Thus it suffices to show that for $q = i-4, \ldots, i$, the profit that can be obtained during T_i due to bids that arrive during T_q is at most $2R(q)$. Let $j'_1 < j'_2 < \ldots < j'_{l-1} < j'_l$, where $l \leq k$, denote the distinct price levels that the offline algorithm sets during T_i. As the prices are powers of 2, the profit obtained by setting price level j due to bids that arrive during T_q is at most $\sum_{s\geq 0} V_{j-s}(q)/2^s$. Thus, the total profit due to bids that arrive during T_q is at most

$$\sum_{r=1}^{l}\sum_{s\geq 0}\frac{V_{j'_r-s}(q)}{2^s} = \sum_{s\geq 0}\frac{1}{2^s}\left(\sum_{r=1}^{l} V_{j'_r-s}(q)\right) \leq \sum_{s\geq 0}\frac{1}{2^s}R(q) = 2R(q)$$

The inequality follows from the fact that $R(q)$, by definition, is the maximum total profit contained in any k levels in bids that arrive during T_q. ■

Additionally, we need the following simple facts:

1. If all bids have duration 1, then $\mathsf{Algo\text{-}Max\text{-}Price}$ is optimal. We rename this algorithm $\mathsf{Alg}(0)$.

2. If all bids have duration greater than $2\log h$, then the randomized algorithm described in the proof of Lemma 2 with $k = \log h$ obtains in expectation a profit that is least $1/2$ of the sum of the bid values in the instance. As sum of the bid values is an upper bound on the optimal revenue, the algorithm is 2-competitive. We call this algorithm $\mathsf{Alg}(\log k)$.

Thus our overall algorithm is the following: Choose k uniformly at random among the set $\{0, 1, 2, 4, \log h/2, \log h\}$ of cardinality $\log\log h + 2$. Run the algorithm $\mathsf{Alg}(k)$.

By Lemma 2 and the observations above, it is easy to see that the algorithm above is $O(\log\log h)$ competitive.

4 Lower bounds for online algorithms

4.1 Lower bounds for deterministic online algorithms

4.1.1 $\Omega(h)$ lower bound for EF-model

Theorem 2 *Any deterministic online algorithm $\mathcal{A}$ for* EF-model *must have a competitive ratio of $\Omega(h)$.*

Proof: Consider the following game that the adversary will play with $\mathcal{A}$. On day 1, there are h^2 bids that announce $(1, h^2, h)$ and h^2 bids that announce $(1, 2, 1)$. If on any day $t \geq 1$, $\mathcal{A}$ chooses $p(t) = 1$, then the game stops. Otherwise, the adversary introduces h^2 new bids that announce $(t+1, t+2, 1)$. If h^2 days have elapsed, the game stops.

Let t^* be the day that the game stops and let B be the set of bids introduced in the game (if $\mathcal{A}$ never chooses price level of 1 then define $t^* = h^2 + 1$). Note that the optimal offline algorithm has the choice of either sticking with the price level h or sticking with the price level 1 on all the t^* days. Thus,

$$OPT(B) \geq \max\left(h^3, \min(t^* h^2, h^4)\right)$$

On the other hand, if $t^* = h^2 + 1$ then $\mathcal{A}$ gets a revenue of h^3; otherwise, it gets a revenue of $h^2 + h^2 = 2h^2$. Thus, the competitive ratio of $\mathcal{A}$ is at least

$$\min\left(\frac{h^4}{h^3}, \frac{\max(h^3, t^* h^2)}{2h^2}\right) = \frac{h}{2}.$$

This completes the proof. ■

4.1.2 $\Omega(\sqrt{\log h})$ lower bound for PL-model

In the proof of theorem 2, the adversary provides the algorithm with two choices and the algorithm always makes the wrong choice in hindsight. The crucial property exploited there is that one has to keep setting high prices to get the full revenue from bids with long bid lengths. This of course will not work as it is in PL-model. However, a simple variation does the trick.

Theorem 3 *Any deterministic online algorithm $\mathcal{A}$ for* PL-model *must have a competitive ratio of $\Omega(\sqrt{\log h})$.*

Proof: Consider the following game that the adversary will play with $\mathcal{A}$. On day 1, there are 2^i bids (for every $i = 0, 1, 2, 4, \cdots, \log h - 1$) that announce $(1, \log h, h/2^i)$. There are $h\sqrt{\log h}$ bids that announce $(1, 2, 1)$. If on any day $t \geq 1$, $\mathcal{A}$ chooses $p(t) = 1$, then the game stops. Otherwise, the adversary introduces $h\sqrt{\log h}$ new bids that announce $(t+1, t+2, 1)$. If $\log h$ days have elapsed, the game stops. Note that because of the way the bids with bid value greater than 2 are set up, picking any of the price levels $2, 4, \cdots, h$ will result in a revenue of at most $2h$.

Let t^* be the day that the game stops and let B be the set of bids introduced in the game (if $\mathcal{A}$ never chooses price level of 1 then define $t^* = \log h + 1$). Note that the optimal offline algorithm has the choice of either sticking with the price level 1 or getting all the bids with bid value greater than 1. Thus,

$$OPT(B) \geq \max\left(h \log h, \min\left(ht^* \sqrt{\log h}, h(\log h)^{3/2}\right)\right)$$

Also, the revenue generated by $\mathcal{A}$ is at most $2(t^* - 1)h + h\sqrt{\log h}$. Thus, the competitive ratio of the algorithm is at least

$$\frac{\max(h \log h, \min(ht^* \sqrt{\log h}, h(\log h)^{3/2}))}{2t^* h + h\sqrt{\log h}} \geq \frac{\sqrt{\log h}}{2}$$

This completes the proof. ■

4.2 Lower bounds for randomized online algorithms

We will use Yao's min-max theorem ([2]) to show a lower bound on the competitive ratio of any online algorithm for the models we are considering. To do this, we will need to come up with a set of bid instances $I_1, I_2, \cdots, I_N$ for some N and a probability distribution D on them. By Yao's principle the lower bound will be $\frac{\mathbb{E}_{I \leftarrow D} OPT(I)}{\mathbb{E}_{I \leftarrow D} Rev_A(I)}$ where $Rev_P(I)$ is the revenue generated by the set of prices P on instance I and $A(\cdot)$ is the *fixed* set of prices that has the best expected revenue.

4.2.1 Preliminaries

Geometric distribution will be a key building block in our lower bound constructions. Let $G(p)$ denote the discrete distribution on $m = 1, 2, 3, \cdots$ such that

$$\Pr(m) = (1-p)^{m-1} p$$

We need the following well known facts about the distribution.

Fact 1 *A random variable X drawn from $G(p)$ has the following properties:*

1. *The mean is given by $\mathbb{E}(X) = \frac{1}{p}$*
2. $\Pr[X \le m] = 1 - (1-p)^m$.
3. $\mathbb{E}(X | X \ge m) = m + \mathbb{E}(X)$*, that is, the geometric distribution is memoryless.*

We will also need the following technical claim.

Claim 1 *Let c be some integer. Consider the following recurrence relation for any $1 < k < c$:*

$$x_i = 1 + x_{i+1} \left(1 - \frac{1}{c}\right)^{\frac{c}{x_{i+1}}} - (1 + x_{i+1}) \left(1 - \frac{1}{c}\right)^{c^2 - c}$$

$$x_k = 1.$$

Then

$$x_0 > \frac{\sqrt{k}}{4}.$$

To prove the above claim we will need the following result, which follows from binomial expansion and simple algebraic manipulation.

Proposition 3 *For any integers a and b such that $a \ge 2$ and $ab < 1$, the following holds:*

$$(1-b)^a \ge 1 - ab + \frac{a^2 b^2}{4}$$

Proof of Claim 1. We first note that the function $y(1-\frac{1}{c})^{c/y} - (y+1)(1-\frac{1}{c})^{c^2-c}$ is an increasing function. Thus, by induction[1] the claim can be proved if one can show that

$$\ell(j) = 4 + \sqrt{k-j} \left(1 - \frac{1}{c}\right)^{\frac{4c}{\sqrt{k-j}}} - \frac{4 + \sqrt{k-j}}{4} \left(1 - \frac{1}{c}\right)^{c^2 - c} > \sqrt{k-j+1}.$$

[1]In the base case, $x_k = 1$ and the hypothesis $x_j > \frac{\sqrt{k-j}}{4}$ for $j = k, k-1, \cdots, 0$ is trivially true.

Using Proposition 3, we have

$$\begin{aligned}\ell(j) &\geq 4+\sqrt{k-j}\left(1-\frac{4}{\sqrt{k-j}}+\frac{4}{k-j}\right)-\frac{4+\sqrt{k-j}}{4}\left(1-\frac{1}{c}\right)^{c^2-c}\\ &\geq \sqrt{k-j}+\frac{2}{\sqrt{k-j}}.\end{aligned}$$

The last inequality follows from the fact that $\frac{2}{\sqrt{k-j}} \geq \frac{4+\sqrt{k-j}}{4}\left(1-\frac{1}{c}\right)^{c^2-c}$. Squaring both sides and dropping some terms on the right hand side gives us:

$$\begin{aligned}\ell(j)^2 &\geq k-j+2\cdot\sqrt{k-j}\cdot\frac{2}{\sqrt{k-j}}\\ &= k-j+4\\ &> k-j+1.\end{aligned}$$

The proof is complete. ■

4.2.2 $\Omega\left(\sqrt{\frac{\log h}{\log\log h}}\right)$ lower bound for EF-model

We are now ready to present the set of instances for the lower bound. We will not specify the instances explicitly but will describe a procedure that will implicitly describe both the instances and the probability distribution over them.

We will have $\Theta\left(\frac{\log h}{\log\log h}\right)$ distinct bid values: $h, h/\log h, h/(\log h)^2, \cdots, 1$. Let L be the number of distinct bid values. We will say bids with bid value $h/(\log h)^i$ are at level i. Let c denote the quantity $\log h$.

For the ease of exposition, we will describe the construction in terms of a tree with L levels (with the root having level 0) such that each node at level i is a bid with bid length h^2/c^{2i} and bid value h/c. Every node v at level $0 \leq i < L-1$ has m_v many children where m_v is chosen from $G(1/c)$. However, if m_v exceeds c^2 then it is truncated to c^2. If u is the j^{th} child of node v (which is at level i), then the bid corresponding to node u is $(s_v+(j-1)\frac{h^2}{c^{2i}}, s_v+j\frac{h^2}{c^{2i}}-1, \frac{h}{c^{i+1}})$, where s_v is the start day of the bid corresponding to v. The root node has the bid $(1, h^2, h)$.

We will denote a typical bid set by I and denote the induced distribution on the instances by D. By definition, every bid instance has exactly L levels.

Note that by construction, for any node v at level i, if w is the m_v^{th} child of v and if e_v and e_w are the end times of the bids corresponding to v and w, then $e_w \leq e_v$.

We first estimate the expected revenue of the optimal fixed set of prices.

Lemma 3 *The expected revenue generated by the optimal fixed set of prices is $O(h)$ (where fixed means that the set of prices remains the same for all bid instances).*

Proof: We first show that the expected revenue generated by setting $p(t) = h/c^i$ for some $i = 0, \cdots, L-1$ is at most h. This will follow if we can show that the expected number of bids at level i is at most c^i. Let s_i be the random variable denoting the number of nodes at level i in a randomly constructed tree. We will show by induction that for all i, $\mathbb{E}_{[0,i]}(s_i) \leq c^i$, where $\mathbb{E}_{[a,b]}(\cdot)$ is the expectation over all the randomness used in the levels $a, a+1, \cdots, b$. The base case is true as $s_0 = 1$. Now assume that for all $i \leq j$, $\mathbb{E}_{[0,i]}(s_i) \leq c^i$. Let the s_j nodes at level j be denoted by $v_1, v_2, \cdots, v_{s_j}$. Then

$$s_{j+1} = \sum_{i=1}^{s_j} m_{v_i}$$

By linearity of expectation we have

$$\mathbb{E}_{[0,j+1]}(s_{j+1}) = \mathbb{E}_{[0,j]}\left(\sum_{i=1}^{s_j} \mathbb{E}_{[j+1,j+1]}(m_{v_i})\right)$$

Since for any node v in the tree, m_v is chosen from $G(1/c)$ and then conditionally truncated to c^2, $\mathbb{E}_{[j+1,j+1]}(m_v) \leq c$ (as the mean of $G(1/c)$ is c by Fact 1). Thus, we have

$$\mathbb{E}_{[0,j+1]}(s_{j+1}) \leq \mathbb{E}_{[0,j]}\left(\sum_{i=1}^{s_j} c\right) = c\mathbb{E}_{[0,j]}(s_j) \leq c^{j+1}$$

The equality follows from linearity of expectation and the last inequality follows from the induction hypothesis. Similarly, one can show that the revenue generated by setting $p(t) = h/c^i$ is at least $h/2$ (it is an easy calculation to show that for any node v at level j, $\mathbb{E}_{[j,j]}(m_v) \geq c/2$).

Now to complete the proof, we will show that setting $p(t) = h/c^i$ will generate a revenue which is within a factor of 2 of the optimal fixed set of prices. If the optimal indeed chooses the same price throughout then by the discussion in the last paragraph we are done. So there must exist a time t such that $p(t) \neq p(t+1)$. First assume that $p(t) < p(t+1)$. This implies that the price "moves" from one node v (such that the associated bid has value $p(t)$) to another node u (such that the associated bid has value $p(t+1)$) in the tree. We first note that u cannot be an ancestor of v as that will mean that the bid corresponding to u was alive at time t and thus, the buyer at u will not pay more than $p(t)$ for a copy of the item (recall that any two price levels in the construction differ by a factor of at least c). If it were the case that $p(t) > p(t+1)$ then again the same arguments show that the node associated with $p(t)$ is not an ancestor of the node associated with $p(t+1)$. A similar argument shows that any time t' when the price changes, has to be after the end day of some bid. Further, the arguments in the previous paragraph also show that if we consider any subtree rooted at a level j node, then the revenue generated from the bids in that subtree by setting $p(t) = h/c^i$ for any $i \geq j$ is $\Theta(h/c^j)$. Thus, for all time $t' \geq t+1$ such that $p(t') = p(t+1) \neq p(t)$, setting $p(t') = p(t)$ will change the revenue by a factor of at most 2.

Applying the argument in the last paragraph for all time t' such that $p(t'+1) \neq p(t)$, we can come up with a new price function $p'(t) = h/c^i$ for some i that generates an expected revenue within a factor of 2 of the expected revenue of the optimal fixed set of prices. The proof is complete. ■

We now bound the expected value of $OPT(I)$ where I is chosen according to D.

Lemma 4

$$\mathbb{E}_{I \leftarrow D}[OPT(I)] = \Omega\left(h\sqrt{\frac{\log h}{\log\log h}}\right)$$

Proof: Given an instance I, $OPT(I)$ can be computed recursively starting from the leaves. In particular, if in the recursion one is considering node v at level i then one can either set the prices that give the optimal revenue for each of its children or it can set the price to be h/c^i throughout the time when the bid at v is alive.

Let $Rev(v)$ denote the optimal revenue obtainable from the subtree rooted at v. Note that as the tree is constructed from a random process, $Rev(v)$ is a random variable. By the discussion in the previous paragraph, if v is at level i (let its m_v children be denoted by $u_1, u_2, \cdots, u_{m_v}$), then

$$Rev(v) \geq \max\left(\frac{h}{c^i}, \sum_{j=1}^{m_v} Rev(u_j)\right). \tag{1}$$

Note that $\mathbb{E}_{I \leftarrow D}[OPT(I)] = \mathbb{E}(Rev(r))$ where r is the root. By definition of expectation,

$$\begin{aligned}\mathbb{E}(Rev(v)) &= \mathbb{E}\left(Rev(v) \mid m_v \leq \frac{c}{\mathbb{E}(Rev(u_1))}\right) \Pr\left[m_v \leq \frac{c}{\mathbb{E}(Rev(u_1))}\right] \\ &+ \mathbb{E}\left(Rev(v) \mid m_v > \frac{c}{\mathbb{E}(Rev(u_1))}\right) \Pr\left[m_v > \frac{c}{\mathbb{E}(Rev(u_1))}\right]\end{aligned}$$

From (1),

$$\mathbb{E}(Rev(v)) \geq (h/c^i)\Pr\left[m_v \leq \frac{c}{\mathbb{E}(Rev(u_1))}\right] + \left(\sum_{j=1}^{m_v} \mathbb{E}(Rev(u_j))\right) \Pr\left[m_v > \frac{c}{\mathbb{E}(Rev(u_1))}\right]$$

Further, note that since the random coin tosses in subtrees rooted at the children $u_1, \cdots, u_{m_v}$ are independent, $\mathbb{E}(Rev(u_1)) = \mathbb{E}(Rev(u_2)) = \cdots = \mathbb{E}(Rev(u_{m_v}))$.

$$\begin{aligned}\mathbb{E}(Rev(v)) &\geq (h/c^i)\Pr\left[m_v \leq \frac{c}{\mathbb{E}(Rev(u_1))}\right] \\ &+ \mathbb{E}(Rev(u_1)) \cdot \mathbb{E}\left(m_v \mid m_v > \frac{c}{\mathbb{E}(Rev(u_1))}\right) \cdot \Pr\left[m_v > \frac{c}{\mathbb{E}(Rev(u_1))}\right]\end{aligned}$$

To simplify notation, we will x_i to denote the expected optimal revenue generated from any node at level i when the bid values are normalized such that the bid value at level i is 1. That is, for any node v at level i

$$x_i = \frac{c^i Rev(v)}{h}$$

Note that by the above definition, $\mathbb{E}(Rev(u_1)) = x_{i+1}h/c^{i+1}$. Define q to be $(1-1/c)$. Now, by Fact 1, we know that $\Pr[m_v \leq c/x_{i+1}] = 1 - q^{c/x_{i+1}}$. Similarly, $\Pr[m_v > c/\mathbb{E}(x_{i+1})] \geq \Pr[c/\mathbb{E}(x_{i+1}) < m_v \leq c^2 - c] = q^{c/x_{i+1}} - q^{c^2-c}$ and in this case by the memoryless property, $\mathbb{E}(m_v|m_v > c/x_{i+1}) = c/x_{i+1} + c$. Thus,

$$\begin{aligned}x_i &\geq (1 - q^{c/x_{i+1}}) \cdot 1 + (q^{c/x_{i+1}} - q^{c^2-c})\left(\frac{c}{x_{i+1}} + c\right)\frac{x_{i+1}}{c} \\ &= 1 + x_{i+1}q^{c/x_{i+1}} - (x_{i+1}+1)q^{c^2-c} \\ &= 1 + x_{i+1}\left(1 - \frac{1}{c}\right)^{c/x_{i+1}} - (x_{i+1}+1)\left(1 - \frac{1}{c}\right)^{c^2-c}\end{aligned}$$

The above recursion is the same as the one in Claim 1. Thus, we have $x_0 > \frac{\sqrt{L}}{4}$ or $\mathbb{E}(Rev(r)) > h \cdot \sqrt{L}/4$ (where r is the root), which proves the lemma. ■

Thus, by Lemma 3 and 4 we have the following result.

Theorem 4 *Any randomized online algorithm for* EF-model *has competitive ratio of* $\Omega\left(\sqrt{\frac{\log h}{\log\log h}}\right)$

4.2.3 $\Omega\left(\sqrt{\frac{\log\log h}{\log\log\log h}}\right)$ lower bound for PL-model

We have the following result.

Theorem 5 *Any randomized online algorithm for* PL-model *has competitive ratio of* $\Omega\left(\sqrt{\frac{\log\log h}{\log\log\log h}}\right)$

We will show how to modify the construction of Section 4.2.2. The analysis to show that the construction proves Theorem 5 is similar to the proof of Theorem 4 and the details are omitted.

The idea is similar to the way the construction for deterministic online algorithms for EF-model in Theorem 2 was modified for construction for deterministic online algorithms for PL-model in Theorem 3. In the construction of Section 4.2.2, each node in the tree corresponded to one bid. We will now modify the construction such that now each node (except the leaves) have multiple bids at different bid values (all having the same start and end days) such that setting the price at any of the bid values gives the same revenue (within a factor of 2).

We will now make the construction more precise. Define $d = \log\log h$ and let k be such that $2^{(d^{2k+2}-1)/(d^2-1)} = h$. Note that $k = \Omega\left(\sqrt{\frac{\log\log h}{\log h \log h \log h}}\right)$. As in the construction of EF-model we will consider the bid instances as a tree. There will be $k+1$ levels with the root being at level 0. A node v at level i consists of bids at $d^{2(k-i)}$ distinct bid values. In particular for $\ell = 0, \cdots, d^{2(k-i)}-1$, there are $(2^{\sum_{j=0}^{i-1} d^{2(k-j)}+\ell})\cdot d^i$ many bids that announce $(s_u, s_u + d^{2(k-i)}-1, h/(2^{\sum_{j=0}^{i-1} d^{2(k-j)}+\ell}))$ where s_u is the start time of parent u of v. Further, each v at level i has m_v children where m_v is chosen from $G(1/d)$. However, if m_v exceeds d^2, then it is truncated to d^2. Finally the level 0 node has d^{2k} distinct price levels. For every $\ell = 0, \cdots, d^{2k}-1$, there are 2^ℓ bids that announce $(1, l^{2k}-1, h/2^\ell)$.

Using arguments similar to the ones used in Section 4.2.2, it can be shown that if one stays at the same level (now instead of having a single price per node, we will have $l^{2(k-i)}$ different prices at level i node to get revenue of hd^{2k-i} from all bids in that node) then the expected revenue obtained is $O(hd^{2k})$. Further, one can show that this is within a constant factor of the expected revenue of the optimal set of fixed prices. Finally, one can show that expected value of the optimal revenue from each bid instance is $\Omega\left(hd^{2k}\sqrt{\frac{\log\log h}{\log\log\log h}}\right)$ which proves Theorem 5.

Acknowledgments

The authors would like to thank Anna Karlin and Tracy Kimbrel for helpful discussions.

References

[1] V. Guruswami, J. Hartline, A. Karlin, D. Kempe, C. Kenyon, and F. McSherry. On profit-maximizing envy-free pricing. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA)*, pages 1164–1173, 2005.

[2] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.