Competitive Algorithms for Unbounded One-Way Trading

Francis Y. L. Chin[1⋆], Bin Fu[2], Minghui Jiang[3],
Hing-Fung Ting[1⋆⋆], and Yong Zhang[1⋆⋆⋆]

[1] Department of Computer Science, The University of Hong Kong, Hong Kong
{chin,hfting,yzhang}@cs.hku.hk
[2] Department of Computer Science, University of Texas-Pan American,
Edinburg, TX 78539, USA, bfu@utpa.edu
[3] Department of Computer Science, Utah State University,
Logan, UT 84322, USA, mjiang@cc.usu.edu

Abstract. In the one-way trading problem, a seller has some product to be sold to a sequence σ of buyers $u_1, u_2, \ldots, u_\sigma$ arriving online and he needs to decide, for each u_i, the amount of product to be sold to u_i at the then-prevailing market price p_i. The objective is to maximize the seller's revenue. We note that most previous algorithms for the problem need to impose some artificial upper bound M and lower bound m on the market prices, and the seller needs to know either the values of M and m, or their ratio M/m, at the outset. Moreover, the performance guarantees provided by these algorithms depend only on M and m, and are often too loose; for example, given a one-way trading algorithm with competitive ratio $\Theta(\log(M/m))$, its actual performance can be significantly better when the actual highest to actual lowest price ratio is significantly smaller than M/m.

This paper gives a one-way trading algorithm that does not impose any bounds on market prices and whose performance guarantee depends directly on the input. In particular, we give a class of one-way trading algorithms such that for any positive integer h and any positive number ϵ, we have an algorithm $A_{h,\epsilon}$ that has competitive ratio $O(\log r^*(\log^{(2)} r^*)\ldots(\log^{(h-1)} r^*)(\log^{(h)} r^*)^{1+\epsilon})$ if the value of $r^* = p^*/p_1$, the ratio of the highest market price $p^* = \max_i p_i$ and the first price p_1, is large and satisfy $\log^{(h)} r^* > 1$, where $\log^{(i)} x$ denotes the application of the logarithm function i times to x ; otherwise, $A_{h,\epsilon}$ has a constant competitive ratio Γ_h. We also show that our algorithms are near optimal by showing that given any positive integer h and any one-way trading algorithm A, we can construct a sequence of buyers σ with $\log^{(h)} r^* > 1$ such that the ratio between the optimal revenue and the revenue obtained by A is at least $\Omega(\log r^*(\log^{(2)} r^*)\ldots(\log^{(h-1)} r^*)(\log^{(h)} r^*))$.

⋆ Research supported by HK RGC grant HKU-711709E and Shenzhen basic research project (NO.JCYJ20120618143038947).
⋆⋆ Research supported by HK RGC grant HKU-716412E.
⋆⋆⋆ Research supported by NSFC 11171086 and Natural Science Foundation of Hebei Province A2013201218.

1 Introduction

The *one-way trading problem*, which was introduced by El-Yaniv *et al.*[10, 11] and Borodin *et al.*[6], involves selling a fixed amount of a product to a sequence of buyers, with the objective of maximizing the seller's revenue. A major difference between this problem and other general revenue maximization problems commonly studied in economics and computer science is that for the general problems, the seller has some control of the prices; he can determine the amount and the price of product to be sold to each buyer. However, for the one-way trading problem, a seller has no control of the prices, and when a buyer arrives, he can only determine the amount of the product to be sold at the then-prevailing market price. There are many applications that can be modeled as a one-way trading problem. One example is money-exchange, in which a seller has some initial asset, say US dollars, and he wants to sell them at the price of some target asset, say yen. In fact, the one-way trading problem is formulated as a money exchange problem in [10].The exchange rate fluctuates everyday. To maximize the amount of yen gained, the seller needs to decide, for each day, the right amount of US dollars to be changed at the exchange rate used on that day. Other applications such as stock selling in a stock market and electricity selling in a power grid can also be modeled naturally as one-way trading problem.

It is easy to solve the offline version of the problem; if the seller knows all the future prices, he can simply wait for the highest price and then sell all his product at that price. However, our problem is online in nature, and without knowledge of future prices, a player cannot be sure whether the current price is the highest. More formally, in our one-way trading problem, there is a seller who has L units of product to be sold, and there is a sequence of buyers $u_1, u_2, \ldots, u_\sigma$ arriving. When a buyer u_i arrives, the then-prevailing unit price p_i is revealed and the seller needs to decide the amount x_i of product to be sold to u_i at price p_i, and the objective is to maximize $\sum_i p_i x_i$ subject to $\sum_i x_i \leq L$. The main features of the problem that make it difficult and interesting include: (1) the seller has no control of the prices, which fluctuates with time, and (2) he does not have any knowledge about the future prices, i.e., when u_i arrives, he does not know any price p_j where $j > i$, and (3) he needs to decide the amount of product to be sold to a buyer u_i as soon as u_i arrives.

Previous results.
After introducing the one-way trading problem, El-Yaniv *et al.* gave in [11] an algorithm for the problem that works under the assumption that there are a lower bound m and an upper bound M on the market prices such that $p_i \in [m, M]$ for all p_i, and that these bounds m and M are known to the algorithm. They proved that their algorithm has competitive ratio $O(\log(M/m))$, and showed that it is optimal by deriving a matching lower bound. They also studied the case when only the ratio M/m is known, and gave an optimal algorithm for this case. Without knowledge of M/m in advance, an algorithm with competitive ra-

tio $O(\log(M/m)\log^{1+\epsilon}(\log(M/m)))$ was given in [11]. [4] More recently, Fujiwara *et al.*[12] have studied the one-way trading problem under the assumption that the input prices follow some given probability distribution. In [9], Chen *et al.* introduced the *planning game problem*, which is similar to the one-way trading problem, and they gave an algorithm for their problem which imposes some different constraint on the prices: instead of assuming that $p_i \in [m, M]$ for some price range $[m, M]$, their algorithm assumes that the difference between any two consecutive prices p_i and p_{i+1} is not too large, or more precisely, they assumed that for any i, $p_i/\beta \leq p_{i+1} \leq \alpha p_i$ for some fixed $\alpha, \beta > 1$. They showed that if there are n buyers, their algorithm has competitive ratio $\frac{n\alpha\beta-(n-1)(\alpha+\beta)+(n-2)}{\alpha\beta-1}$.

In [11], El-Yaniv *et al.* also studied another problem similar to the one-way trading problem, namely the *1-max-search* problem, in which there is a sequence of prices coming online, and when a price arrives, we have to decide immediately whether we accept the price or not. The objective is to accept the highest price. By assuming that all prices fall in the range $[m, M]$ and these bounds m and M are known, they gave an algorithm for this problem with competitive ratio $O(\sqrt{M/m})$, i.e., the ratio of the highest price and the price accepted by the algorithm is $O(\sqrt{M/m})$. In [14], Lorenz *et al.* generalized the 1-max-search problem to the *k-max-search* problem, in which the objective is to accept the k highest prices. By requiring that the bounds m and M are known, they gave an optimal algorithm for the problem, which has competitive ratio $\sqrt[k+1]{k^k(M/m)}$.

For recent related research on revenue maximization that allows price setting, we mention the *auction problem* [4, 13] and the *pricing problem* [1–3, 5, 7, 8]. For the auction problem, there are bidders competing for the products by sending their bids to the auctioneer, and the auctioneer chooses some bidders, and determines the price and amount of products to be sold to each chosen bidder. For the pricing problem, we have studied an interesting version in [16] in which the seller has m units of products to sell and each buyer has a valuation (i.e., price at which he is willing to buy) represented by a function $v(x)$, which gives the valuation per unit if x units are purchased. When the highest valuation v^* is known, we gave an algorithm with competitive ratio $O(\log v^*)$. Moreover, this algorithm was shown to be asymptotically optimal by giving a matching lower bound. We also studied in [17] an extension of this problem, in which there are multiple types of products and each user is interested in a particular bundle of products.

Our Contribution.

In this paper, we consider the unbounded one-way trading algorithm that does not need to impose any constraint on the market prices, and we derive a bound on its competitive ratio that depends directly on the input, or more precisely, depends on $r^* = p^*/p_1$, the ratio of the highest price $p^* = \max_i p_i$ and the

[4] Remark 4 in [11] said that it is possible to achieve an upper bound of $O(\log(M/m)(\log(\log^{(k)}(M/m)))^{1+\epsilon})$, however, this bound contradicts with the general lower bound of the competitive ratio in this paper. Thus, the claimed upper bound in [11] does not hold for general case.

first price p_1 (in fact, our algorithm will treat p_1 as the lowest price and ignore any prices lower than p_1). Furthermore, the algorithm does not make any assumption on the number of prices p_i in the input sequence and an adversary can terminate the sequence at any time by sending buyers with extremely low prices. In fact, we propose a generic one-way trading algorithm whose behavior depends on some given function $f(x)$, which can be any function satisfying the following conditions: (i) It is non-increasing, and (ii) $\int_1^\infty f(t)dt$ is bounded. Roughly speaking, $f(x)$ helps us determine the amount of products the seller should sell at price x. We show that by using $f(x)$ in our generic algorithm, we have a one-way trading algorithm with competitive ratio $O(\frac{1}{r^* f(r^*)})$. Thus, to get a small competitive ratio, it suffices to find a $f(x)$ that satisfies (i) and (ii), and $f(x)$ is as large as possible. We observe that the following class of functions satisfies our requirements:

$$\frac{1}{x \log x (\log^{(2)} x) \dots (\log^{(h-1)} x)(\log^{(h)} x)^{1+\epsilon}},$$

where h is any positive integer and ϵ is any positive real number, and where $\log^{(k)} x$ denotes the function $\log \log \dots \log x$, which applies the logarithm function k times to x. Based on these functions, (a different function for each different value of h and ϵ) our generic algorithm gives us a class of one-way trading algorithms such that for any fixed positive integer h and positive number ϵ, we have an algorithm $A_{h,\epsilon}$ such that when $\log^{(h)} r^* > 1$, $A_{h,\epsilon}$ has competitive ratio $O((\log r^*) \dots (\log^{(h-1)} r^*)(\log^{(h)} r^*)^{1+\epsilon})$; otherwise, its competitive ratio is bounded by some constant Γ_h depending only on h. We also show that the bounds are almost tight by employing the divergence of the same class of function when $\epsilon = 0$ to design an adversary such that, given any online algorithm A for the problem, the adversary gives a sequence of buyers σ such that the ratio between the revenue obtained by an optimal offline algorithm on σ and that obtained by A is $\Omega((\log r^*) \dots (\log^{(h-1)} r^*)(\log^{(h)} r^*))$ for any positive integer h. Moreover, we show that our results still hold if the amount of products sold to each buyer is constrained to be at most a maximum amount specified by the buyer.

2 Upper bound

Since products could be sold fractionally, we may assume, without loss of generality, that the seller has one unit of product to sell. The offline version is easy to solve: the whole product is assigned to the buyer with the highest market price. However, for the online version, we have no information about the future prices, including the bound of the highest market price. If the whole amount of product has been sold by the time a buyer with very high market price arrives, the performance will be poor. Thus, we must keep or reserve some amount in case there is a future buyer with a higher market price. On the other hand, if we reserve too much for the possible buyer with higher market price and assign very

little to the buyers who have come already, the performance will be also poor since the possible buyer with higher price may not come. Thus, to have a good performance, the amount sold and the amount remaining should be balanced nicely.

For the purposes of illustrating the main ideas of our algorithm only, consider the case when all prices are non-negative integers; in general, our algorithm is not restricted to integer prices. We make the following observations.

- Our algorithm should only sell products when the price is strictly higher than the maximum price that we have seen so far. For example, suppose the input sequence of prices is 1, 4, 2, 3, 6, 5, 12. We can ignore the prices 2, 3, 5 and do not sell any at these prices because the optimal offline algorithm will ignore these prices anyway; if our solution is competitive for the input 1, 4, 6, 12, it will also be competitive for the input 1, 4, 2, 3, 6, 5, 12. Therefore, we can focus on handling price sequences that are strictly increasing.
- If we have a good solution for a sequence of strictly increasing and consecutive prices, i.e., for the price sequence $1, 2, 3, ..., p^*$, then we can easily modify it to get a good solution for any price sequences that are strictly increasingly with the highest price p^*. For example, suppose that for the prices 1, 2, 3, 4, our algorithm sells an amount δ_1, δ_2, δ_3, δ_4 of products at prices 1, 2, 3, 4, respectively and thus obtains a revenue of $R = \delta_1 + 2\delta_2 + 3\delta_3 + 4\delta_4$. Then, for the strictly increasing price sequence 1, 3, 4, we can sell an amount of δ_1 at price 1, $\delta_2 + \delta_3$ at price 3, and δ_4 at price 4. Then, the revenue we obtain is $\delta_1 + 3(\delta_2 + \delta_3) + 4\delta_4 \geq R$.

Therefore, our algorithm can focus on strictly increasing and consecutive price sequences. For these sequences, we only need to determine the amount δ_i of products to be sold at price p_i. Since there is only one unit of product, we must have $\sum_{i=1}^{+\infty} \delta_i \leq 1$. Another property that is desirable is that the δ_is should be decreasing, i.e., $\delta_1 > \delta_2 > \delta_3 > ...$[5]; the leading δ_is should be large so that we can sell enough products even if the market crashes very early, i.e., the adversary declares immediately that there are no more buyers, or buyers with extremely low market prices. Then, for any input price sequence with highest value p^*, our algorithm will have revenue at least $\delta_1 + 2\delta_2 + ... + p^*\delta^* \geq (p^*)^2\delta^*/2$, and since no algorithm (including the offline optimal algorithm) can have revenue higher than p^*, the competitive ratio of this algorithm is $O(1/(p^*\delta^*))$ (Lemma 1).

Now we give the algorithm. The algorithm assigns amounts based on a non-increasing function $f(x)$, which computes the value of δ_i such that $\int_0^{+\infty} f(x)dx = 1$, $\int_0^1 f(x)dx = \delta_1$, $\int_1^2 f(x)dx = \delta_2$, ..., $\int_{i-1}^{i} f(x)dx = \delta_i$.

Let $(p_1, p_2, ..., p^*)$ be the sequence of strictly increasing transacted prices, i.e. prices at which the seller sells some (non-zero) amount to the buyer. For ease of analysis, we can normalize this sequence to be $(r_1, r_2, ..., r^*) = (1, p_2/p_1, ..., p^*/p_1)$ where the first price r_1 is 1 and the normalized maximum r^* is the ratio of the

[5] The decreasing of δ_i can be argued easily. WLOG, assume that $p_1 < p_2$ and $\delta_1 \leq \delta_2$, we can show that the competitive ratio can be decreased by moving a small amount from δ_2 to δ_1. This process can continue until $\delta_1 > \delta_2$.

highest transacted price p^* to the lowest transacted price p_1. Any buyers with market price less than p_1 will be ignored. For the sake of simplicity, we shall denote r_i as the normalized price of the i-th buyer and r^* as the highest normalized price. The online selling strategy is described below as Algorithm 1. Note that Algorithm 1 can handle non-integer prices.

Algorithm 1 : Online Selling

Initially, let $cr^* \leftarrow 0$. $\{\}cr^*$ is the current highest normalized price.
repeat
  when a buyer with normalized market price r comes
  if $r > cr^*$ **then**
    Assign $\int_{cr^*}^{r} f(x)dx$ products to this buyer.
    $cr^* \leftarrow r$
  end if
until no buyer comes

Lemma 1. *Suppose r^* is the highest normalized market price, if $f(.)$ is a non-increasing function, the competitive ratio is at most $O(\frac{1}{r^* \cdot f(r^*)})$.*

Proof. The revenue received from Algorithm 1 is

$$r_1 \int_0^{r_1} f(r)dr + r_2 \int_{r_1}^{r_2} f(r)dr + ... + r^* \int_{r^{*-}}^{r^*} f(r)dr \geq \int_0^{r^*} r \cdot f(r)dr$$

where r^{*-} is the second highest normalized market price in the sequence.

Since $f(r)$ is non-increasing, the revenue received from Algorithm 1 is at least

$$\int_0^{r^*} r \cdot f(r)dr \geq f(r^*) \int_0^{r^*} rdr = f(r^*) \cdot \frac{(r^*)^2}{2}.$$

Note that the maximum revenue is r^* given that the seller has only one unit to sell, and therefore, the competitive ratio is at most

$$\frac{r^*}{\int_0^{r^*} r \cdot f(r)dr} = O(\frac{1}{r^* \cdot f(r^*)}).$$

□

In order to get a good performance, we need to find a non-increasing function $f(x)$ such that $\int_0^{\infty} f(x)dx$ converges to 1, or more simply, $\int_0^{\infty} f(x)dx = c$ for some constant c (as we can normalize it to 1 later), and for any $x > 1$, $f(x)$ is as large as possible. After assuming the first market price is 1, we may just analyze the property of $\int_1^{\infty} f(x)dx$. It is well known that $\int_1^{\infty} \frac{1}{x}dx$ diverges and thus $f(x) = 1/x$ is too large. Similarly as $\int_1^{\infty} \frac{1}{x^{1+\epsilon}}dx$ converges for any $\epsilon > 0$, $f(x) = 1/(x \cdot x^{\epsilon})$ is too small. This suggests that $f(x) = 1/(x\xi(x))$ where $\xi(x)$ is

an increasing function and $\xi(x) = o(x^\epsilon)$ for any $\epsilon > 0$. A good candidate for $\xi(x)$ is a poly-log function of x. This motivates us to focus on the class of functions $f(x) = 1/(x \log x \log^{(2)} x \ldots (\log^{(i)} x)^{1+\epsilon})$ where $\epsilon > 0$ and $\log^{(i)} x$ denotes the application of the logarithm function i times to x, where $i \geq 0$. Now we define the class of functions formally.

Definition 1. *Assume real number $\epsilon \geq 0$, integer $i \geq 0$, $b_0 = 1$, and $b_{i+1} = e^{b_i}$, define function $q_{i,\epsilon}(x)$ for $x \geq b_i$ as follows.*

$$q_{i,\epsilon}(x) = \begin{cases} x^{1+\epsilon} & \text{if } i = 0 \\ x \cdot q_{i-1,\epsilon}(\ln x) & \text{if } i > 0 \end{cases}$$

Thus, $q_{1,\epsilon}(x) = x \cdot (\ln x)^{1+\epsilon}$, $q_{2,\epsilon}(x) = x \cdot (\ln x) \cdot (\ln^{(2)} x)^{1+\epsilon}$, and $q_{i,\epsilon}(x) = x \cdot (\ln x) \cdot (\ln^{(2)} x) \cdot \ldots \cdot (\ln^{(i)} x)^{1+\epsilon}$. The following lemma gives the condition when $\int_{b_i}^{+\infty} \frac{1}{q_{i,\epsilon}(x)} dx$ converges.

Lemma 2. *For each integer $i \geq 0$, $\int_{b_i}^{+\infty} \frac{1}{q_{i,\epsilon}(x)} dx$ converges if and only if $\epsilon > 0$, in particular, $\int_{b_i}^{+\infty} \frac{1}{q_{i,0}(x)} dx$ diverges.*

Proof. By induction on i. When $i = 0$, $b_0 = 1$, it is easy to see that $\int_{b_0}^{+\infty} \frac{1}{q_{0,\epsilon}(x)} dx = \int_1^{+\infty} \frac{1}{x^{1+\epsilon}} dx$ converges if and only if $\epsilon > 0$. Assume that the hypothesis is true for $i - 1$. As $b_i = e^{b_{i-1}}$, we have $\int_{b_i}^{+\infty} \frac{1}{q_{i,\epsilon}(x)} dx = \int_{e^{b_{i-1}}}^{+\infty} \frac{1}{x \cdot q_{i-1,\epsilon}(\ln x)} dx = \int_{b_{i-1}}^{+\infty} \frac{1}{q_{i-1,\epsilon}(y)} dy$, where $y = \ln x$. Thus, $\int_{b_i}^{+\infty} \frac{1}{q_{i,\epsilon}(x)} dx$ converges if and only if $\epsilon > 0$. □

The following theorem shows the competitive ratio of Algorithm 1 by constructing $f(x)$ from $q_{i,\epsilon}(x)$, i.e., proving that the area under $f(x)$ when $x > 0$ is bounded and $f(x)$ is non-increasing and defined for all $x > 0$.

Theorem 1. *Suppose r^* is the highest normalized market price, there exists an online algorithm $A_{h,\epsilon}$ for the unbounded one-way trading problem with competitive ratio $O(1)$ if $r^* < b_h$ and $O(q_{h-1,\epsilon}(\log r^*))$ if $r^* \geq b_h$ for any fixed positive integer h and any real number $\epsilon > 0$.*

Proof. For any fixed positive integer h, b_h is a constant such that $\ln^{(h)} b_h = 1$. From Lemma 2, for any real number $\epsilon > 0$, suppose $\int_{b_h}^{+\infty} \frac{1}{q_{h,\epsilon}(x)} dx$ converges to a constant, say c. As $\ln^{(h)}(x) \geq 1$ when $x \geq b_h$, we define function $f_{h,\epsilon}(x)$ as follows.

$$f_{h,\epsilon}(x) = \begin{cases} \frac{1}{b_h + c \cdot q_{h,\epsilon}(b_h)} & \text{if } 0 < x < b_h \\ \frac{q_{h,\epsilon}(b_h)}{b_h + c \cdot q_{h,\epsilon}(b_h)} \cdot \frac{1}{q_{h,\epsilon}(x)} & \text{if } x \geq b_h \end{cases}$$

It can be verified that $\int_0^{+\infty} f_{h,\epsilon}(x) dx = 1$ and $f_{h,\epsilon}(x)$ is non-increasing (since $f_{h,\epsilon}(x) = f_{h,\epsilon}(b_h)$ is a constant when $0 < x < b_h$ and $f_{h,\epsilon}(x)$ is decreasing when $x \geq b_h$), i.e., $f_{h,\epsilon}(x)$, which depends on h and ϵ, satisfies the requirement of Algorithm 1, which gives $A_{h,\epsilon}$. By Lemma 1, we can analyze the competitive ratio w.r.t. the highest market price r^*.

- If $r^* < b_h$, the competitive ratio is $O(\frac{b_h + c \cdot q_{h,\epsilon}(b_h)}{r^*})$, which is $O(1)$.
- If $r^* \geq b_h$, the competitive ratio is $O(\frac{1}{r^* \cdot f_{h,\epsilon}(r^*)})$, which is $O(q_{h-1,\epsilon}(\log r^*))$, i.e., $O(\log r^* \log^{(2)} r^* ... (\log^{(h)} r^*)^{1+\epsilon})$.

□

Now we consider the case where each buyer has a maximum amount of products he wants to buy at the market price. This variant can be regarded as an extension of the previous part. Algorithm 1 assigns products only based on the buyer's market price with no regard for how much the buyer is able to buy, i.e., the buyer's quota. Modify Algorithm 1 by taking into consideration the buyer's quota, we have the following conclusion. (For details, please see appendix.)

Theorem 2. *For the unbounded one-way trading problem, if each buyer has a maximum amount of products he wants to buy at the market price, there is an online selling strategy with competitive ratio $O(1)$ if $r^* < b_h$ and $O(q_{h-1,\epsilon}(\log r^*))$ if $r^* \geq b_h$ for any fixed integer h and any real number $\epsilon > 0$, where r^* is the highest normalized market price.*

3 Lower bound

In this part, we present a lower bound for the competitive ratio of the unbounded one-way trading problem. We will show that the lower bound and the upper bound given in Section 2.1 are almost tight; in another words, Algorithm 1 is near optimal.

To derive a lower bound on the competitive ratio, we give an adversary that determines the sequence of prices $p_1, p_2, p_3 \ldots$, and whenever the seller has sold some products, the adversary checks the total revenue the seller has accumulated so far, and if it is not competitive, the adversary declares immediately that there are no more buyers, or buyers with extremely low market price, i.e., the market "crashes". The prices p_i's grow exponentially, i.e., $p_i = \Theta(e^i)$. The adversary also determines for each i a bound Δ_i, which is the minimum amount of product sold during the first i prices in order to prevent the market crashes. In other words, if the amount of product sold at price $p_1, p_2, \ldots, p_k$ are $s_1, s_2, \ldots, s_k$, respectively, and $s_1 \geq \Delta_1, s_1 + s_2 \geq \Delta_2, \ldots, \sum_{k=1}^{j-1} s_k \geq \Delta_{j-1}$, and $\sum_{k=1}^{j} s_k < \Delta_j$, the market crashes immediately at price p_j. Note that in such case, the seller has sold at most $\Delta_j - \Delta_{j-1}$ unit of product at p_j, and since p_j is much larger than all previous prices, we would be able to show that the total revenue obtained by the seller will be dominated by the last transaction and is $O((\Delta_j - \Delta_{j-1})p_j)$. On the other hand, an offline algorithm can sell the whole unit of product at p_j and gets the maximum revenue p_j. Thus the competitive ratio of the algorithm is $\Omega(\frac{1}{\Delta_j - \Delta_{j-1}})$ if the adversary "crashes" the market after p_j. The challenge for getting a large lower bound is to decide the $\Delta_i's$ such that (i) they are unbounded (i.e., $\Delta_i \to \infty$ when $i \to \infty$) so that the seller will fail eventually to meet the requirement on the minimum amount of product sold, and (ii) $\Delta_i - \Delta_{i-1}$ is as small as possible. The bound $\Delta_i = \frac{1}{e+1} + \frac{1}{e+2} + \ldots + \frac{1}{e+i}$ can be considered as a

good candidate, which will lead us to a lower bound of $\Omega(i)$ or $\Omega(\log p_i)$ when the highest price $p_i = O(e^i)$. Below, we describe some other Δ_i's that will lead us to a substantially larger bound.

From Lemma 2, we know that $q_{h,0}(x)$ is a good candidate such that there is a $b_h > 0$ causing $\int_{b_h}^{+\infty} \frac{1}{q_{h,0}(x)} dx$ to diverge, where $\ln^{(h)} b_h = 1$. The adversary in Algorithm 2 uses $\sum_{k=1}^{j} \frac{1}{q_{h,0}(b_h+k-1)}$ as a candidate for Δ_j as mentioned before and s_j is the amount of products assigned to buyer u_j. Since $1/q_{h,0}(x)$ is monotone decreasing and $\int_{b_h}^{+\infty} \frac{1}{q_{h,0}(x)} dx$ diverges for any fixed integer $h > 0$, the sum $\sum_{k=1}^{\infty} \frac{1}{q_{h,0}(b_h+k-1)}$ diverges. Therefore, Algorithm 2 must be terminated on some buyer since the seller has only one unit of product.

Algorithm 2 : Adversary for online selling

Assume that the seller has one unit of product to sell.
Let $j \leftarrow 0$.
repeat
  Let $j \leftarrow j + 1$.
  Send buyer u_j with market price e^{b_h+j-1} to the seller.
  The seller sells s_j product to buyer u_j.
until $\sum_{k=1}^{j} s_k \leq \sum_{k=1}^{j} \frac{1}{q_{h,0}(b_h+k-1)}$

Assume the adversary stops sending buyers after the arrival of buyer u_j. From Algorithm 2, the total revenue received is $\sum_{k=1}^{j} s_k \cdot e^{b_h+k-1}$, while the maximum offline revenue is e^{b_h+j-1}. The following lemma estimates the total revenue received from Algorithm 2.

Lemma 3. $\sum_{k=1}^{j} s_k \cdot e^{b_h+k-1} = O(\frac{e^{b_h+j-1}}{q_{h,0}(b_h+j-1)})$

Proof. From the adversary's strategy, at any step $j' < j$,

$$\sum_{k=1}^{j'} s_k > \sum_{k=1}^{j'} \frac{1}{q_{h,0}(b_h+k-1)},$$

and in the last step j,

$$\sum_{k=1}^{j} s_k \leq \sum_{k=0}^{j} \frac{1}{q_{h,0}(b_h+k-1)}.$$

Therefore,

$$\sum_{k=1}^{j} s_k \cdot e^{b_h+k-1} \leq \sum_{k=1}^{j} \frac{e^{b_h+k-1}}{q_{h,0}(b_h+k-1)}.$$

In Lemma 4, we show that

$$\frac{e^{b_h+k}}{q_{h,0}(b_h+k)} \cdot \frac{q_{h,0}(b_h+k-1)}{e^{b_h+k-1}} = \frac{e \cdot q_{h,0}(b_h+k-1)}{q_{h,0}(b_h+k)} \geq c$$

for some constant $c > 1$ and any $k \geq 1$. Thus,

$$\sum_{k=1}^{j} \frac{e^{b_h+k-1}}{q_{h,0}(b_h+k-1)} \leq \frac{e^{b_h+j-1}}{q_{h,0}(b_h+j-1)} \cdot \frac{1}{1-1/c} = O(\frac{e^{b_h+j-1}}{q_{h,0}(b_h+j-1)}).$$

□

Lemma 4. *For any integer $h \geq 1$ and $k \geq 1$, there exist a constant $c > 1$, such that*

$$\frac{e \cdot q_{h,0}(b_h+k-1)}{q_{h,0}(b_h+k)} \geq c$$

Proof. Based on the logarithmic characteristic of $q_{h,0}(x)$, i.e., the increasing rate is decreasing with the increase of x, $\frac{q_{h,0}(b_h+k-1)}{q_{h,0}(b_h+k)}$ achieves the lowest value when $k = 1$. Thus, it is sufficient to prove the following inequality for any integer $h \geq 1$.

$$\frac{e \cdot q_{h,0}(b_h)}{q_{h,0}(b_h+1)} \geq c > 1 \tag{1}$$

We prove Inequality (1) by induction on h.

Basis step: $h = 1$. As $b_h = e$,

$$\frac{e \cdot q_{1,0}(b_1)}{q_{1,0}(b_1+1)} = \frac{e \cdot b_1 \cdot \ln b_1}{(b_1+1) \cdot \ln(b_1+1)} \approx 1.513 > 1$$

Induction step: Assume Inequality (1) is true for h,

$$\begin{aligned}\frac{e \cdot q_{h,0}(b_h)}{q_{h,0}(b_h+1)} &= \frac{e \cdot b_h \cdot \ln b_h \cdot ... \cdot \ln^{(h)} b_h}{(b_h+1) \cdot \ln(b_h+1) \cdot ... \cdot \ln^{(h)}(b_h+1)} \\ &= \frac{e \cdot b_h}{b_h+1} \prod_{h'=1}^{h} \frac{\ln^{(h')} b_h}{\ln^{(h')}(b_h+1)} \geq c > 1 \end{aligned} \tag{2}$$

Then for $h+1$,

$$\begin{aligned}\frac{e \cdot q_{h+1,0}(b_{h+1})}{q_{h+1,0}(b_{h+1}+1)} &= \frac{e \cdot b_{h+1} \cdot \ln b_{h+1} \cdot ... \cdot \ln^{(h+1)} b_{h+1}}{(b_{h+1}+1) \cdot \ln(b_{h+1}+1) \cdot ... \cdot \ln^{(h+1)}(b_{h+1}+1)} \\ &= \frac{e \cdot b_{h+1}}{b_{h+1}+1} \prod_{h'=1}^{h+1} \frac{\ln^{(h')} b_{h+1}}{\ln^{(h')}(b_{h+1}+1)} \end{aligned} \tag{3}$$

We shall prove that $\frac{e \cdot b_{h+1}}{b_{h+1}+1} \prod_{h'=1}^{h+1} \frac{\ln^{(h')} b_{h+1}}{\ln^{(h')}(b_{h+1}+1)}$ as given in Equation (3) is larger than

$\frac{e \cdot b_h}{b_h+1} \prod_{h'=1}^{h} \frac{\ln^{(h')} b_h}{\ln^{(h')}(b_h+1)}$ as given in Equation (2) term by term. Since $\ln b_{h+1} = b_h$, we have $\ln(b_{h+1}+1) < b_h + 1$. Thus, for any $2 \le h' \le h+1$,

$$\frac{\ln^{(h')} b_{h+1}}{\ln^{(h')}(b_{h+1}+1)} > \frac{\ln^{(h'-1)} b_h}{\ln^{(h'-1)}(b_h+1)}$$

As for the first few terms, because $\ln b_{h+1} = b_h$ and $\ln(b_{h+1}+1) = b_h + \delta$, where $\delta \ll 1$, we have

$$\frac{b_{h+1} \cdot \ln b_{h+1}}{(b_{h+1}+1) \cdot \ln(b_{h+1}+1)} \ge \frac{b_h}{b_h+1}$$

Thus, we have shown that

$$\frac{e \cdot q_{h,0}(b_h+k)}{q_{h,0}(b_h+k+1)} \ge c > 1.$$

□

Based on the above analysis, Theorem 3 gives the lower bound on the competitive ratio of the unbounded one-way trading problem.

Theorem 3. *The competitive ratio of the unbounded one-way trading problem is at least* $\Omega(q_{h,0}(\log r^*)) = \Omega(\log r^* \cdot \log^{(2)} r^* \cdot ... \cdot \log^{(h+1)} r^*)$ *where* r^* *is the highest normalized market price and* $h > 0$ *is any fixed integer.*

Proof. Assume that Algorithm 2 terminates on some buyer u_j. As mentioned before, the revenue received from Algorithm 2 is $\sum_{k=1}^{j} s_k \cdot e^{b_h+k-1} = O(\frac{e^{b_h+j-1}}{q_{h,0}(b_h+j-1)})$ (Lemma 3), and the maximum offline revenue is e^{b_h+j-1} by assigning the whole product to buyer u_j with the market price e^{b_i+j-1}. As $p_j = e^{b_h+j-1}$, the performance ratio is at least $\Omega(q_{h,0}(b_h+j-1)) = \Omega(\log p_j \log^{(2)} p_j ... \log^{(h+1)} p_j)) = \Omega(\log r^* \log^{(2)} r^* ... \log^{(h+1)} r^*))$ since b_h can be regarded as a constant and $r^* = p_j/p_1 = e^{j-1}$. □

4 Conclusion

There are many real applications where the market price fluctuates and cannot be controlled by the seller. It is a problem of practical interest to find a good revenue-maximizing (or profit-maximizing) selling strategy for the seller in such a situation. This paper has made an attempt towards this direction. However, the strategy prescribed in this paper may not be too practical in the sense that, for example, products are not sold when the market price decreases. The reality is that, in practice, the seller may have a fixed time-frame to sell and cannot wait forever for the buyer with the highest price to arrive, and price movements from one moment to the next may not be drastic or arbitrary. Additional assumptions and/or constraints to the unbounded one-way trading problem to reflect such practical realities will be studied in our next attempt and hopefully could lead to more practical selling strategies.

References

1. Moshe Babaioff, Shaddin Dughmi, Robert Kleinberg, Aleksandrs Slivkins, Dynamic Pricing with Limited Supply. In Proceedings of the 13th ACM Conference on Electronic Commerce, pp. 74-91, 2012.
2. Ashwinkumar Badanidiyuru, Robert Kleinberg, Yaron Singer. Learning on a budget: posted price mechanisms for online procurement In Proc. of the 13th ACM Conference on Electronic Commerce, pp. 128-145, 2012.
3. Maria-Florina Balcan, Avrim Blum, Yishay Mansour. Item pricing for revenue maximization. In Proceedings of the 9th ACM Conference on Electronic Commerce, pp. 50-59, 2008.
4. Avrim Blum, Jason D. Hartline. Near-optimal online auctions. In Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1156-1163, 2005.
5. Avrim Blum, Anupam Gupta, Yishay Mansour, Ankit Sharma. Welfare and Profit Maximization with Production Costs. In Proceedings of 52th Annual IEEE Symposium on Foundations of Computer Science, pp. 77-86, 2011.
6. Allan Borodin and Ran El-Yaniv. Online Computation and Competitive Analysis Cambridge University Press, 1998.
7. Tanmoy Chakraborty, Eyal Even-Dar, Sudipto Guha, Yishay Mansour, S. Muthukrishnan. Approximation Schemes for Sequential Posted Pricing in Multi-unit Auctions. In Proc. of the 6th International Workshop on Internet and Network Economics, pp. 158-169, 2010.
8. Tanmoy Chakraborty, Zhiyi Huang, Sanjeev Khanna. Dynamic and non-uniform pricing strategies for revenue maximization. In Proceedings of 50th Annual IEEE Symposium on Foundations of Computer Science, pp. 495-504, 2009.
9. Gen-Huey Chen, Ming-Yang Kao, Yuh-Dauh Lyuu, Hsing-Kuo Wong. Optimal buy-and-hold strategies for financial markets with bounded daily returns. SIAM J. Compt. 31(2) 447-459, 2001. A preliminary version appeared in STOC 1999, pp. 119-128.
10. Ran El-Yaniv, Amos Fiat, Richard M. Karp, G. Turpin. Competitive analysis of financial games. In Proceedings of 50th Annual IEEE Symposium on Foundations of Computer Science, pp. 372-333, 1992.
11. Ran El-Yaniv, Amos Fiat, Richard M. Karp, G. Turpin. Optimal search and one-way trading online algorithms. Algorithmica 30(1), 101-139 (2001)
12. Hiroshi Fujiwara, Kazuo Iwama, Yoshiyuki Sekiguchi. Average-case competitive analyses for one-way trading. Journal of Combinatorial Optimization, 21(1) pp. 83-107, 2011.
13. Elias Koutsoupias, George Pierrakos. On the Competitive Ratio of Online Sampling Auctions. In Proceedings of the 6th International Workshop on Internet and Network Economics, pp. 327-338, 2010.
14. Julian Lorenz, Konstantinos Panagiotou, Angelika Steger. Optimal algorithms for k-search with application in option pricing. Algorithmica (2009) 55:311-328. A preliminary version appeared in ESA 2007, pp. 275-286.
15. Roger B. Myerson. Optimal auction design. Mathematics of Operations Research, 6:58-73, 1981.
16. Yong Zhang, Francis Y.L. Chin, Hing-Fung Ting. Competitive Algorithms for Online Pricing. In Proceedings of the 17th Annual International Computing and Combinatorics Conference, pp. 391-401, 2011.
17. Yong Zhang, Francis Y.L. Chin, Hing-Fung Ting. Online pricing for bundles of multiple items. Journal of Global Optimization 58(2), pp. 377-387.