Online Search with Best-Price and Query-Based Predictions

Spyros Angelopoulos,[1,2] **Shahin Kamali**[3]**, and Dehou Zhang**[3]

[1] Centre National de la Recherche Scientifique (CNRS)
[2] Sorbonne Université, Laboratoire d'informatique de Paris 6, LIP6, Paris, France
[3] University of Manitoba, Winnipeg, Manitoba, Canada
spyros.angelopoulos@lip6.fr, shahin.kamali@umanitoba.ca, zhangd37@myumanitoba.ca

Abstract

In the online (time-series) search problem, a player is presented with a sequence of prices which are revealed in an online manner. In the standard definition of the problem, for each revealed price, the player must decide irrevocably whether to accept or reject it, without knowledge of future prices (other than an upper and a lower bound on their extreme values), and the objective is to minimize the competitive ratio, namely the worst case ratio between the maximum price in the sequence and the one selected by the player. The problem formulates several applications of decision-making in the face of uncertainty on the revealed samples.

Previous work on this problem has largely assumed extreme scenarios in which either the player has almost no information about the input, or the player is provided with some powerful, and error-free advice. In this work, we study learning-augmented algorithms, in which there is a potentially erroneous prediction concerning the input. Specifically, we consider two different settings: the setting in which the prediction is related to the maximum price in the sequence, as well as well as the setting in which the prediction is obtained as a response to a number of binary queries. For both settings, we provide tight, or near-tight upper and lower bounds on the worst-case performance of search algorithms as a function of the prediction error. We also provide experimental results on data obtained from stock exchange markets that confirm the theoretical analysis, and explain how our techniques can be applicable to other learning-augmented applications.

1 Introduction

The *online (time series) search* problem formulates a fundamental setting in decision-making under uncertainty. In this problem, a player has an indivisible asset that wishes to sell within a certain time horizon, e.g., within the next d days, without knowledge of d. On each day i, a *price* p_i is revealed, and the player has two choices: either accept the price, and accrue a *profit* equal to p_i, or reject the price, in which case the game repeats on day $i + 1$. If the player has not sold by day d (i.e., has rejected all prices $p_1, \ldots, p_{d-1}$), then the last price p_d is accepted by default.

This problem was introduced and studied in (El-Yaniv et al. 2001) by means of *competitive analysis*. Namely, the competitive ratio of the player's strategy (or algorithm) is

defined as the worst case ratio, over all price sequences, of the maximum price in the sequence divided by the price accepted by the player. Thus, the competitive ratio provides a worst-case guarantee that applies even to price sequences that are adversarially generated. Since the problem formulates a basic, yet fundamental transaction setting, a player that follows a competitively efficient algorithm has a safeguard against any amount of volatility with respect to prices.

(El-Yaniv et al. 2001) gave a simple, deterministic algorithm that achieves a competitive ratio equal to $\sqrt{M/m}$, where M, m are upper and lower bounds on the maximum and minimum price in the sequence, respectively, and which are assumed to be known to the algorithm. This bound is tight for all deterministic algorithms. Randomization can improve the competitive ratio to an asymptotically tight bound equal to $O(\log(M/m))$. See also the surveys (El-Yaniv 1998; Mohr, Ahmad, and Schmidt 2014).

Online search is a basic paradigm in the class of online financial optimization problems. Several variants and settings have been studied through the prism of competitive analysis; see, e.g., (Damaschke, Ha, and Tsigas 2009; Lorenz, Panagiotou, and Steger 2009; Xu, Zhang, and Zheng 2011; Clemente et al. 2016). The problem has also been studied as a case study for evaluating several performance measures of online algorithms, including measures alternative to competitive analysis (Boyar, Larsen, and Maiti 2014; Ahmad, Pirron, and Schmidt 2021). Extensions of online search such as *one-way trading* and *portfolio selection* have also been studied extensively both within competitive analysis; e.g., (El-Yaniv et al. 2001; Fujiwara, Iwama, and Sekiguchi 2011; Borodin, El-Yaniv, and Gogan 2000), as well as from the point of view of regret minimization; e.g., (Hazan and Kale 2015; Uziel and El-Yaniv 2020; Das, Johnson, and Banerjee 2014). We refer also to the survey (Li and Hoi 2014).

Previous work on competitive analysis of online financial optimization problems, including online search, has largely assumed a status of almost complete uncertainty in regards to the input. Namely, the algorithm has either no knowledge, or very limited knowledge concerning the input. This models a scenario that is overly pessimistic: indeed, in everyday financial transactions, the players have some limited, albeit potentially erroneous information on the market.

This observation illustrates the need for a study of online financial optimization problems using the framework

of *learning-enhanced* competitive algorithms (Lykouris and Vassilvitskii 2018; Purohit, Svitkina, and Kumar 2018). Such algorithms have access to some machine-learned information on the input which is associated with a *prediction error* η. The objective is to design algorithms whose competitive ratio degrades gently as the prediction error increases, but also quantify the precise tradeoff between the performance and the prediction error. Several online optimization problems have been studied in this setting, including caching (Lykouris and Vassilvitskii 2018; Rohatgi 2020), ski rental and non-clairvoyant scheduling (Purohit, Svitkina, and Kumar 2018; Wei and Zhang 2020), makespan scheduling (Lattanzi et al. 2020), rent-or-buy problems (Banerjee 2020; Anand, Ge, and Panigrahi 2020; Gollapudi and Panigrahi 2019), secretary and matching problems (Antoniadis et al. 2020b; Lavastida et al. 2020), and metrical task systems (Antoniadis et al. 2020a). See also the survey (Mitzenmacher and Vassilvitskii 2020).

To our knowledge, there is no previous work on competitive analysis of online financial optimization problems in the learning-enhanced model. Note that this is in contrast to analysis based on regret minimization, which inherently incorporates predictions as "experts" (Cover and Ordentlich 1996; Hazan and Megiddo 2007).

1.1 Contribution

We present the first results on competitive online search in a setting that provides predictions related to the price sequence. We show that the obtained competitive ratios are optimal under several models. We also introduce new techniques for leveraging predictions that we argue can be applicable to other learning-augmented online problems. More precisely, we study the following two settings:

The prediction is the best price Here, the prediction is the best price that the player is expected to encounter. We further distinguish between the model in which no other information on this prediction is available to the player, which we call the *oblivious* model, and the model in which an *upper bound* to the prediction error is known, which we call the *non-oblivious* model. In the latter, the player knows that the error is bounded by some given value H, i.e., $\eta \leq H$. The oblivious model is more suitable for markets with very high volatility (e.g., cryptocurrencies), whereas the non-oblivious model captures less volatile markets (e.g., fiat currencies), in which we do not expect the prices to fluctuate beyond a (reasonable) margin. For both models, we give optimal (tight) upper and lower bounds on the competitive ratio as function of the prediction error. A novelty in the analysis, in comparison to previous work, is that we perform an asymmetric analysis in regards to the error, namely we distinguish between *positive* and *negative* error, depending on whether the best price exceeds the prediction or not. This distinction is essential in order to prove the optimality of our results.

The prediction is given as response to binary queries In this model, the prediction is given as a response to n binary queries, for some fixed n. For example, each query can be of the form "will a price at least equal to p appear in the sequence?". This model captures settings in which the predictions define *ranges* of prices, as opposed to the single-value prediction model, and was introduced recently in the context of a well-known resource allocation problem in AI, namely the *contract scheduling* problem (Angelopoulos and Kamali 2021). The prediction *error* is defined as the number of erroneous responses to the queries, and we assume non-oblivious algorithms which know an upper bound $H < n$ on the error. Online search was previously studied under an error-free query model in (Clemente et al. 2016), however their proposed solution is non-robust: a single query error can force the algorithm to accept a price as bad as the smallest price in the sequence.

We present two different algorithms in this model, and prove strict upper bounds on their competitive ratios, as functions of n and H. The first algorithm uses the n queries so as to choose a price from n suitably defined intervals, then accepts the first price in the sequence that is at least as high as the chosen price; moreover, its performance is guaranteed as long as at most half of the query responses are correct. We then present an algorithm based on *robust binary search*, which allows to select a suitable price from a much larger space of 2^n intervals, thus leading to improved performance, at the expense of a relatively smaller (but still high) tolerance to errors (i.e., the theoretical analysis assumes that $H < n/4$). This result is the main technical contribution of this work, and we expect that it can find applications in many other settings in which we must identify a "winner" from a large set of candidates, in the presence of errors. We give such a concrete application in Section 5. We complement the robust binary-search upper bound with a theoretical lower bound on the competitive ratio in this query model.

For both models, we evaluate experimentally our algorithms on real-world data, in which the prices are the exchange rates for cryptocurrencies or fiat currencies. Our experimental results demonstrate that the algorithms can benefit significantly from the predictions,in both models, and that their performance decreases gently as the error increases.

1.2 Notation and definitions

Let $\sigma = (\sigma_i)_{i=1}^d$ be a sequence of prices revealed on days $1, d$. Given an algorithm A, $A(\sigma)$ is the *profit* of A on σ; since σ and A are often implied from context, we simply refer to the profit of the algorithm as its accepted price. We denote by p^* the optimal price in the input. Given an algorithm A, we denote its competitive ratio by CR(A). Recall that m, M are the known lower and upper bounds on the prices in the sequence, respectively. We denote by ON* the optimal online algorithm without predictions, i.e., the algorithm of competitive ratio $\sqrt{M/m}$.

A *reservation* algorithm with price q is an algorithm that accepts the first price in the sequence that is at least q. For example, it is known that ON* can be described as a reservation algorithm with price $\sqrt{M/m}$.

Due to space limitations, we omit several technical proofs. All omitted details can be found in (Angelopoulos, Kamali, and Zhang 2021)

2 Algorithms with Best-Price Prediction

In this setting, the prediction is the highest price that will appear in the sequence. In the remainder of the section, we denote this prediction with $p \in [m, M]$, and recall that p^* is the optimal price. The prediction p is associated with an error η, defined as follows. If $p^* \leq p$, we define η to be such that $1 - \eta = p^*/p$, and we call this error *negative*, in the sense that the best price is no larger than the predicted price. Note also that the negative error ranges in $[0, (M - m)/M]$, that is, $\eta < 1$, in this case. If $p^* > p$, we define η to be such that $1 + \eta = p^*/p$, and we call the error *positive*, in the sense that the best price is larger than the predicted price. Since $1 < p^*/p \leq M/m$, the positive error ranges in $(0, (M - m)/m]$. Naturally, the online algorithm does not know neither the error value, nor its parity. The parity is a concept that we introduce for the benefit of the analysis.

Depending on the volatility of the market, the positive and negative error can fluctuate within a certain range. Let H_n, H_p denote upper bounds on the negative and positive errors, respectively, i.e., $H_n \leq (M - m)/M$, and $H_p \leq (M - m)/m$. We distinguish between non-oblivious and oblivious algorithms, namely between algorithms that know H_n and H_p, and algorithms that do not, respectively.

2.1 Oblivious algorithms

We first study oblivious algorithms, and show matching upper and lower bounds on the competitive ratio. Given algorithm A with prediction p, define the function $s_A(p, m, M) \in [m, M]$ as the smallest price revealed on day 1 (i.e., the smallest value of p_1) such that A accepts that price on day 1. Define also $r_A = s_A(p, m, M)/p$. We first show a lower bound on the competitive ratio.

Theorem 1. *For any algorithm A with prediction p,*

$$\text{CR}(A) \geq \begin{cases} (1 - \eta)/r_A, & \text{if } \eta \leq 1 - r_A \\ (1 - \eta)M/m, & \text{if } \eta > 1 - r_A, \end{cases}$$

if the error is negative, and

$$\text{CR}(A) \geq \begin{cases} (1 + \eta)/r_A, & \text{if } \eta \geq r_A - 1 \\ M/m, & \text{if } \eta < r_A - 1, \end{cases}$$

if the error is positive.

Proof. Case 1: η is negative, i.e., $p^* = p(1 - \eta)$. The adversary chooses $p^* = (1 - \eta)M$, which implies that $p = M$.

Suppose first that $r_A \leq 1 - \eta$, hence $r_A \cdot p \leq p^*$. The adversary presents the sequence $r_A \cdot p, p^*, \ldots, p^*$. From the definition of r_A, A accepts the price on day 1, and $\text{CR}(A) \geq p^*/(r_A \cdot p) = (1 - \eta)/r_A$.

Next, suppose that $r_A > 1 - \eta$. We have $r_A \cdot p > (1 - \eta) \cdot p = p^*$. The adversary presents the sequence $p^*, m, \ldots, m$. By definition, A rejects the price on day 1, hence its profit is m, and $\text{CR}(A) \geq p(1 - \eta)/m = (1 - \eta)M/m$.

Case 2: η is positive, i.e., $p^* = p(1 + \eta)$. The adversary chooses $p^* = M$, which implies that $p = M/(1 + \eta)$.

Suppose first that $r_A \leq 1 + \eta$, that is, $r_A \cdot p \leq p^*$. The adversary chooses the sequence of prices $r_A \cdot p, p^*, p^*, \ldots, p^*$. By definition, A accepts on day 1, therefore $\text{CR}(A) \geq p^*/(r_A \cdot p) = (1 + \eta)/r_A$.

Next, suppose that $r_A > 1 + \eta$, which implies $r_A \cdot p > (1 + \eta) \cdot p = p^* = M$. The intuition here is that A does not accept on day 1 a price equal to M, which is clearly a bad decision. The adversary chooses the sequence of prices $M, m, \ldots, m$, and thus $\text{CR}(A) \geq M/m$. $\square$

Next, we show a class of algorithms whose competitive ratio matches Theorem 1. For any $r > 0$, define the *oblivious reservation algorithm*, named ORA_r as the algorithm with reservation price $r \cdot p$, given the prediction p.

Theorem 2. *The algorithm ORA_r (with reservation price $r \cdot p$) has competitive ratio*

$$\text{CR}(\text{ORA}_r) \leq \begin{cases} (1 - \eta)/r, & \text{if } \eta \leq 1 - r \\ (1 - \eta)M/m, & \text{if } \eta > 1 - r, \end{cases}$$

if the error is negative, and

$$\text{CR}(\text{ORA}_r) \leq \begin{cases} (1 + \eta)/r, & \text{if } \eta \geq r - 1 \\ M/m, & \text{if } \eta < r - 1, \end{cases}$$

if the error is positive.

Figure 1 illustrates the competitive ratio of ORA_r, $\text{CR}(\text{ORA}_r)$, as function of η, for different values of the parameter r. First, we observe that there is no value of r^* such that ORA_{r^*} dominates ORA_r with $r \neq r^*$. More precisely, for any pair of r_1 and r_2, there are some values of η for which ORA_{r_1} has a better competitive ratio while for other values of η, ORA_{r_2} has a better competitive ratio.

For positive error, the competitive ratio degrades linearly in the error with slope $1/r$. Note however that if $r = 1.5$, we have $\text{CR}(\text{ORA}_r) = M/m(= 10)$ for $\eta < r - 1 = 0.5$, since for positive error in $(0, 0.5)$, ORA_r does not trade at day 1 even if the trading price is M. For the other values of r, we have $r \leq \eta + 1$ for which $\text{CR}(\text{ORA}_r) \leq (1 + \eta)/r$, and hence the algorithm performs better when r becomes larger.

For negative but *small* values of error, we have that $\text{CR}(\text{ORA}_r) = (1 - \eta)/r$, thus the performance improves linearly in error, again with slope $1/r$. For larger values, i.e., if $\eta > 1 - r$, there is a "jump" in the competitive ratio, which increases from 1 to $(1 - r)M/m$. This is because when $\eta > 1 - r$, the reservation price becomes larger than the best price, and the algorithm is forced to trade for the minimum price at the last day in the worst case. Following this jump, $\text{CR}(\text{ORA}_r)$ improves linearly with the error, this time with slope M/m. This improvement corresponds to the smaller upper bound value of the best price (the optimal profit) as the negative error increases.

Even though ORA_r is optimal according to Theorems 1 and 2, its competitive ratio may be worse than ON^* for certain ranges of error (namely, for small negative or large positive error). However, this is unavoidable: the next corollary shows that there is no oblivious algorithm with best-price prediction that improves upon ON^* for all values of the error, i.e., cannot dominate ON^* for all values of error.

Corollary 1. *For any oblivious algorithm A with best-price prediction, there exists some range of error η for which $\text{CR}(A) > \sqrt{M/m}$.*
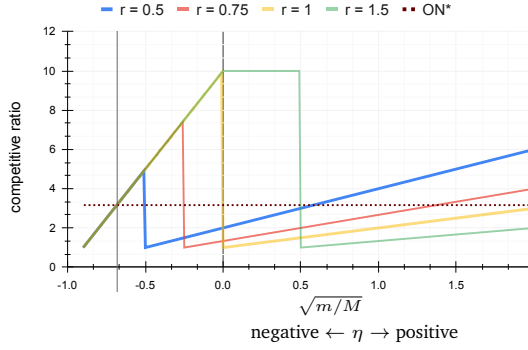
Figure 1: The competitive ratio of ORA_r as function of the error η, and the parameter r. Here we choose $M/m = 10$, and thus the ranges of negative and positive error are $[0, 0.9]$ and $(0, 9)$, respectively.

2.2 Non oblivious algorithms

In this section we show matching upper and lower bounds on the competitive ratio, in the setting in which the algorithm knows upper bounds H_n and H_p on the negative and positive error, respectively.

We call an algorithm A *robust* if for all values of M and m, and all values of η (negative or positive), $\text{CR}(A) \leq \sqrt{M/m}$. In light of Corollary 1, without knowing H_n and H_p, no online algorithm can be robust. In what follows, we will show that there exist robust non-oblivious algorithms. In particular, we define the algorithm ROBUST-MIX which works as follows. If $(1 + H_p)/(1 - H_n) > \sqrt{M/m}$, then ROBUST-MIX ignores the prediction and applies ON^*. Otherwise, i.e., if $(1 + H_p)/(1 - H_n) \leq \sqrt{M/m}$, ROBUST-MIX is an algorithm with reservation price equal to $p' = p(1 - H_n)$. Note that since $H_n \leq (M - m)/M$, we have $p' \geq mp/M$.

Theorem 3. $\text{CR}(\text{ROBUST-MIX}) \leq$

$$\begin{cases} \min\{(1-\eta)/(1-H_n), \sqrt{M/m}\}, & \textit{for negative error} \\ \min\{(1+\eta)/(1-H_n), \sqrt{M/m}\}, & \textit{for positive error}. \end{cases}$$

We also prove a matching bound which establishes that ROBUST-MIX is the optimal non-oblivious algorithm:

Theorem 4. *Any robust non-oblivious algorithm has competitive ratio at least*

$$\begin{cases} \min\{(1-\eta)/(1-H_n), \sqrt{M/m}\}, & \textit{for negative error} \\ \min\{(1+\eta)/(1-H_n), \sqrt{M/m}\}, & \textit{for positive error}. \end{cases}$$

Proof. Let A denote a robust non-oblivious algorithm. Let $s'_A(p, m, M, H_n, H_p) \in [m, M]$ be the smallest price for which A accepts on day 1. Define $r_A = s'_A(p, m, M, H_n, H_p)/p$.

We use the assumption that A is robust to establish that $r_A \leq 1 - H_n$. By way of contradiction, suppose that $H_n > 1 - r_A$. Fix a value of positive error η_p and let $\epsilon > 0$ be any small positive value such that $\epsilon < \min\{H_n - 1 + r_A, \eta_p\sqrt{m/M}\}$. By Theorem 1, for a value of negative error equal to $\eta_n = 1 - r_A + \epsilon$, the competitive ratio of A must

be at least $(1 - \eta_n)M/m = (r_A - \epsilon)M/m$. Given that A is robust, it follows that $(r_A - \epsilon)M/m \leq \sqrt{M/m}$, that is, $r_A \leq \sqrt{m/M} + \epsilon$. Now by Theorem 1, for the positive error η_p, we have

$$\begin{aligned} \text{CR}(A) &\geq \min\{(1 + \eta_p)/r_A, M/m\} \geq \\ &\min \{(1 + \eta_p)/(\sqrt{m/M} + \epsilon), M/m\} > \sqrt{M/m}. \end{aligned}$$

Since $r_A \leq 1 - H_n$, for any values of negative error η, we have $\eta \leq 1 - r_A$ and by Theorem 1, $\text{CR}(A) \geq (1 - \eta)/r_A \geq (1 - \eta)/(1 - H_n)$. For values of positive error η, $\text{CR}(A) \geq \min\{(1+\eta)/r_A, M/m\} \geq (1+\eta)/(1-H_n)$. □

3 Query-based Predictions

In this section, we study the setting in which the prediction is in the form of responses to n binary queries $Q_1, \ldots, Q_n$, for some fixed n. Hence, the prediction P is an n-bit string, where the i-th bit is the response to Q_i. We assume that the algorithm knows an upper bound H on η. Therefore, the responses to least $k - H$ queries are guaranteed to be *correct*, and the responses to at most H queries may be *incorrect* (or *wrong*). We assume the setting of non-oblivious algorithms in this model. This is because without an upper bound on the error, the algorithm is in a state of complete lack of knowledge concerning the truthfulness of the queries, and it is not obvious how to use them in a meaningful way.

We present two algorithms, both of which use *comparison queries* concerning the best price p^*. That is, the queries are in the form of "Is $p^* \leq b$, for some given value b?". In our first algorithm, the values of b form a strictly increasing sequence, which allows us to narrow p^* within an interval (range) from a candidate set of n intervals. The second algorithm implements a *robust* version of binary search, in which the candidate set of intervals is exponential in n, hence we can narrow p^* within an interval of much smaller size, and thus obtain a much better estimate on p^*.

3.1 Robust Linear Interval Search algorithm

Define $m = a_0, a_1, \ldots, a_n = M$ so that $r_n = a_1/a_0 = a_2/a_1 = \ldots = a_n/a_{n-1}$, which implies that $r_n = (M/m)^{1/n}$. Define the n intervals $E_1, \ldots, E_n$, where $E_i = [m, a_i)$ (we have $1 \leq i \leq n$). Query Q_i asks whether the best price p^* is in E_i or not, and the response denoted by "1" or "0", respectively.

Consider the n-bit string P formed by responses to $Q_1, \ldots, Q_n$. If all responses were correct, then P would consist of j 0s followed by $(n - j)$ 1s for some $j \in [1, n]$. This would prove that p^* is in the range $[a_j, a_{j+1})$. The algorithm then could use a_j as its reservation price, which yields a competitive ratio of at most $\frac{a_{j+1}}{a_j} = (M/m)^{1/n}$.

We describe the algorithm *Robust Linear Interval Search* (RLIS), that works in the presence of error. From the way queries are defined, it may be possible to detect and correct some of the wrong responses in P as follows. Suppose the response to Q_i is 1, while the response to at least $H + 1$ queries Q_j that come after Q_i (that is, $j > i$) are 0. Given that the number of incorrect responses cannot exceed H, we

infer that the response to Q_i must be incorrect. With a similar argument, if the response to Q_i is 0 and the responses to at least $H+1$ queries that come before Q_i are 1, then the response to Q_i must be incorrect. Thus RLIS starts with a *pre-processing phase* which corrects these two types of incorrect responses. This results in an updated prediction string P' in which every 1-bit is followed by at most H 0-bits and every 0-bit is preceded by at most H 1-bits.

If all responses in P' are 0, RLIS sets its reservation price to a_{n-H+1}. Otherwise, let i_1 denote the index of the first 1 in P', and let $\alpha \leq H$ be the number of 0s after index i_1. Define $l = \max\{0, i_1 - (H+1-\alpha)\}$, then RLIS sets its reservation price to a_l.

Theorem 5. *Algorithm* RLIS *has competitive ratio at most* $(M/m)^{2H/n}$.

Proof. First suppose all responses in P' are 0. There can only be a suffix of at most H 0-responses in P' which are incorrect, that is, p^* is in the range $[a_{n-(H-1)}, M]$. Given that RLIS has reservation price $a_{n-(H-1)}$, and $p^* \leq M$, its competitive ratio is at most $M/a_{n-(H-1)} = (M/m)^{(H-1)/n}$.

Next, suppose that P' contains a 1-bit. We consider two cases, depending on the presence or absence of a 0-bit in P' after index i_1. If there is no 0-response after index i_1 (that is $\alpha = 0$), then $p^* \leq a_{i_1+H-1}$ (because at most H queries can be wrong), while the profit of RLIS is at least a_l. The competitive ratio is thus at most $a_{i_1+H-1}/a_l \leq (M/m)^{2H/n}$.

Next, suppose that there is a 0-response after index i_1, and let $j_0 > i_1$ denote the index of the last such 0 in P'. We will show that $p^* \geq a_l$, where l is the reservation price of RLIS. Suppose first that the response to Q_{i_1} is incorrect. Then, $p^* \geq a_{i_1}$. Suppose next that the response to Q_{i_1} is correct. In this case, the α 0-responses after index i_1 must be wrong. Since there can be up to H errors, at most $(H-\alpha)$ 0-responses that immediately precede index i_1 can be wrong. Therefore, $p^* \geq a_l$ where $l = \max\{0, i_1 - (H+1-\alpha)\}$. This also implies that RLIS has profit at least a_l.

To finish the proof, we need an upper bound on p^*. If the response to Q_{j_0} is incorrect, then $p^* \leq a_{j_0}$. Otherwise, there are $j_0 - i_1 - \alpha$ wrong 1-responses before j_0, from the definition of j_0. Therefore, up to $H - (j_0 - i_1 - \alpha)$ 1-responses that follow j_0 can also be wrong. That is, p^* can be as large as $a_{j'}$, where $j' = \min\{n, j_0 + H - (j_0 - i_1 - \alpha)\} = \min\{n, H + i_1 + \alpha\}$. Given that the reservation price is a_l, the competitive ratio of the algorithm is therefore at most $a_{j'}/a_l \leq (r_n)^{2H} = (M/m)^{2H/n}$. $\square$

3.2 Robust Binary Interval Search algorithm

Algorithm RLIS uses the queries so as to select a reservation price from n candidate intervals. We will now show how to increase this number to 2^n using a new *Robust Binary Search* algorithm (RBIS). Partition the interval $[m, M]$ into 2^n intervals $L_1, L_2, \ldots, L_{2^n}$, where $L_i = (a_{i-1}, a_i]$, $a_0 = m$, and $a_{2^n} = M$. We define the a_i's so that $\rho = a_1/a_0 = a_2/a_1 = \ldots = a_{2^n}/a_{2^n-1}$, where $\rho = (M/m)^{1/2^n}$.

Suppose that $L_1, \ldots, L_{2^n}$ correspond to the 2^n leaves of a binary tree T of height n, and that the best price p^* is in the interval L_x for some $x \in [1, 2^n]$. With perfect queries (zero

error), it is possible to find L_x using binary search on T, which leads to a competitive ratio $a_x/a_{x-1} = (M/m)^{1/2^n}$, by choosing a reservation price equal to a_{x-1}. This is the approach of (Clemente et al. 2016). Unfortunately this simple approach is very inefficient even if a single error occurs (e.g., if Q_1 receives a wrong response, the search will end in a leaf L_y, where $|x - y|$ is as large as $2^{n/2}$.)

Searching with erroneous queries is a well-studied topic, see e.g., the book (Cicalese 2013). A related problem to ours was studied in (Rivest et al. 1980) and (Disser and Kratsch 2017), however there are important differences with respect to our setting. First, these works consider a "dual" problem to ours in which the objective is to minimize the number of queries so as to locate an *exact* leaf in the binary tree. Second, there are certain significant implementation issues that need to be considered. Specifically, (Disser and Kratsch 2017) assumes that when reaching a leaf, an oracle can respond whether this is the sought element or not (in other words, the algorithm receives an error-free response to a query of the form "is an element exactly equal to x"). For our problem and, arguably, for many other problems with query-based predictions, this assumption cannot be made. Moreover (Rivest et al. 1980) does not have an efficient implementation, specifically in comparison to (Disser and Kratsch 2017). We propose a new algorithm using some ideas of (Disser and Kratsch 2017) that is applicable to our problem, and has an efficient implementation.

Algorithm description   Recall that T is a binary search tree with leaves $L_1, L_2, \ldots, L_{2^n}$ and that we search for the leaf L_x. We denote by $l(v)$, $r(v)$ the left and right child of v, respectively, and by T_v the subtree rooted at v.

We describe the actions of the algorithm. Suppose the algorithm is at node v at the beginning of iteration i (in the first iteration, v is the root of T). The algorithm first asks a *main query*, defined as follows: "Is $x \leq q$?", where q is such that L_q is the rightmost leaf of the left subtree of v. We denote by `main(v)` the response to this query. As we discuss shortly, the search may visit the same node multiple times, so we emphasize that `main(v)` is the response to the most recent main query at v. Next, the algorithm finds the first ancestor of v in T, say w, for which `main(w)` $\neq$ `main(v)`. We denote this ancestor of v by $anc(v)$, if it exists, and define $anc(v) = \emptyset$, otherwise. The algorithm continues by asking a *checkup query* which is a repetition of the main query asked for w. We denote the response to the checkup query as `check(v)`. The algorithm continues by taking one of the following actions, after which iteration $i+1$ begins:

- **Move-down:** If $anc(v) = \emptyset$ or `check(v)` = `main(anc(v))`, RBIS moves one level down in T. That is, if `main(v)` is Yes (respectively No), RBIS moves to $l(v)$ (respectively $r(v)$).

- **Move-up**: If `check(v)` $\neq$ `main(anc(v))`, RBIS moves one level up to the parent of v. In this case, RBIS increments a counter `mu`, which is originally set to 0.

The algorithm continues as described above until it exhausts its number n of queries. Suppose the search stops at some node u, and let a_u denote the $(H - mu_{end})$-th ancestor

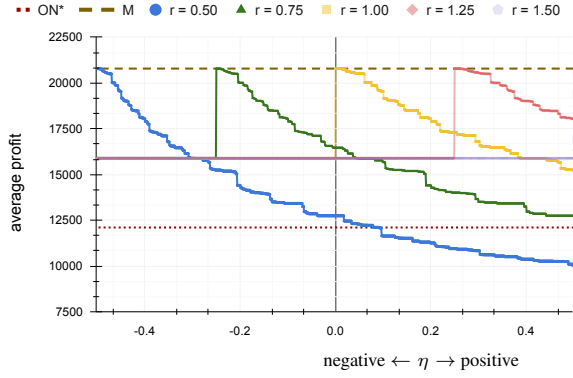Figure 2: Average profit of ORA_r as a function of the error η and the parameter r.

Figure 3: Average profit of ROBUST-MIX as a function of the error η and the bound H.

of u (or the root, if such an ancestor does not exist), where mu_{end} is the content of mu at the end of the search. Let L_l be the leftmost leaf in T_{a_u}, i.e the subtree rooted at a_u. Then RBIS returns this leftmost leaf in T_{a_u}. In particular, for the online search problem, the algorithm sets its reservation price to a_{l-1}.

Analysis We first show the following useful lemma.

Lemma 1. *The following hold: (i) Node a_u is at depth at least $\lfloor n/2 \rfloor - 2H$ in T; and (ii) L_x is a leaf of T_{a_u}.*

Theorem 6. *For every $H \le n/4$, RBIS has competitive ratio at most $(M/m)^{2^{2H-n/2}}$.*

Proof. Let $L_l = [a_{l-1}, a_l]$ and $L_r = [a_{r-1}, a_r]$ denote the leftmost and rightmost leaves in the subtree rooted at a_u. Recall that the algorithm selects a_{l-1} as its reservation price, while Lemma 1 guarantee ensures that L_x, and thus p^* is located in the subtree rooted at a_u, that is, $p^* < a_r$. Therefore, the competitive ratio of RBIS is at most $a_r/a_{l-1} = \rho^{r-l+1}$. Moreover, by Lemma 1, since a_u is at depth at least $d = \lfloor n/2 \rfloor - 2H$ of T, the number of leaves in the subtree rooted at a_u is at least $2^{n-d} < 2^{n/2+2H}$, and thus $\text{CR}(Rbis) \le \rho^{2^{n-d}} < (M/m)^{2^{n/2+2H}/2^n} = (M/m)^{2^{2H-n/2}}$. ☐

Lower bounds We can complement Theorem 6 with the following impossibility result, assuming comparison-based queries over a binary search tree.

Theorem 7. *The competitive ratio of any online search algorithm with n comparison-based queries over a binary tree is at least $(M/m)^{2^{2H-n}}$.*

4 Experimental evaluation

In this section we present an experimental evaluation of the performance of our algorithms.

4.1 Benchmarks and input generation

We evaluate our algorithms on benchmarks generated from real-world currency exchange rates, which are publicly available on several platforms. Specifically, we rely on (EA
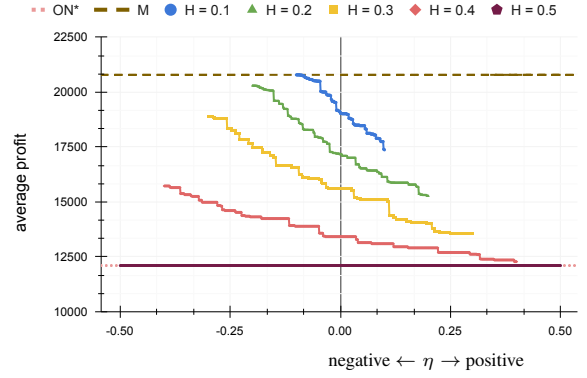
Trading Academy 2021). We used two currency exchange rates (Bitcoin-to-USD and Ethereum-to-USD) and two fiat currency exchange rates (Euro-to-USD and Yen-to-CAD). In all cases, we collected the closing daily exchange rates for a time horizon starting on January 1st, 2018 and ending on September 1st, 2021, which we use as the daily prices. Due to space limitations, we report results on the Bitcoin-to-USD Benchmark, and we refer to (Angelopoulos, Kamali, and Zhang 2021) for additional benchmarks.

For each benchmark, 20 instances $I_1, I_2, \ldots, I_{20}$ of the online search problem are generated as follows. We select 20 *starting days* from the time horizon so that consecutive starting days are evenly distanced. Each starting day and the 199 days that follow it form an instance (of length 200) of the search problem. For each such instance, we select m and M to be respectively the minimum and maximum exchange rates. In all experiments, the reported profits are the average taken over these 20 instances. In particular, we use the average profit of the optimal online algorithm ON^* (without any prediction) as the baseline for our comparisons. Similarly, the average value of the best prices (over all instances) is reported as an upper bound for attainable profits.

4.2 Algorithms with Best-Price Prediction

We test our algorithms using several values of prediction error. We select 500 values of negative error equally distanced in $[0, 0.5]$, as well as 500 equally-distanced values of positive error in $[0, 0.5]$. For each selected value, say η_0, and for each instance I_x of the problem, we test our algorithms for prediction error equal to η_0, that is, the predicted value is generated by applying the error η_0 on the best price in I_x. The average profit of the algorithm over all instances is reported as its average profit for η_0. Choosing $\eta \le 0.5$ implies that the prediction p is at least half and at most twice the best price. For real data, such as currency exchange prices, this range of error is sufficient to capture all instances.

Oblivious algorithms We evaluate ORA_r with different values of the parameter $r \in \{0.5, 0.75, 1.0, 1.25, 1.5\}$ (recall that rp is the reservation price of an algorithm in this
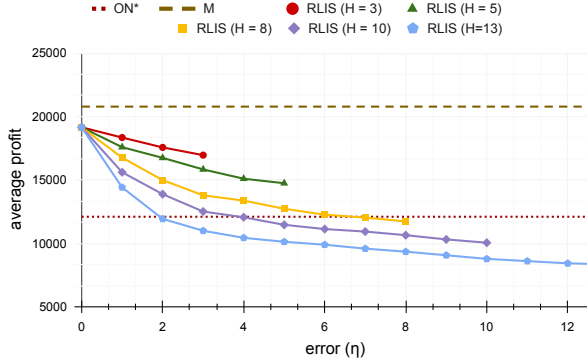
Figure 4: Average profit of RLIS as a function of error (we have $\eta \leq H$).

Figure 5: Average profit of RBIS as a function of error (we have $\eta \leq H$).

class). Figure 2 illustrates the average profit for instances generated from the `Bitcoin-to-USD` benchmark. The findings are consistent with Theorem 2. Specifically, for positive error, for all reported values of r, ORA$_r$ degrades with η (consistently with the linear increase in the competitive ratio in Theorem 2). For small values of negative error, the average profit increases by η, followed by a "drop" when η takes a certain larger value (e.g., when η becomes 0.251 for the algorithm with $r = 0.75$). This follows precisely Theorem 2, as illustrated in Figure 1. For larger values of negative error, the algorithms gain a fixed profit (15890 in the figure), which is the average value of the last-day price. For these values of error, the algorithm sets a reservation price that is too large, and results in the player accepting the last-day price. Last, we note that, as predicted by our competitive analysis, no algorithm dominates another in terms of collected profit.

The results demonstrate that predictions about best price lead to profit gains even for oblivious algorithms. In particular, for all values of error in the reported range (for all $\eta < 0.5$), algorithms with $r \in \{0.75, 1.00, 1.25, 1.50\}$ result in better profit when compared to ON*.

Non-oblivious algorithms We tested ROBUST-MIX with upper bound H on both the positive and negative error, that is, $H = H_n = H_p$ for $H \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. For each such value of H, and for each selected error η, we report the average profit over the 20 instances from the `Bitcoin-to-USD` benchmark (Figure 3). Since the setting is non-oblivious, we only report profits for $\eta \leq H$.

We observe that all algorithms improve as the negative error increases and they degrade as the positive error increases. This is consistent with Theorem 3. Algorithms with smaller H have an advantage over those with larger H, again consistently with Theorem 3. These results demonstrate that non-oblivious algorithms can benefit from best-price predictions.

4.3 Query-based algorithms

In our experiments, we set the number of queries to $n = 25$. We test RLIS and RBIS with H taken from $\{3, 5, 8, 10, 13\}$, and for all values of $\eta \in [0, H]$. Let Alg denote any of our algorithms (RLIS or RBIS for a certain value of H). For each instance I_x from our benchmarks and each selected value of
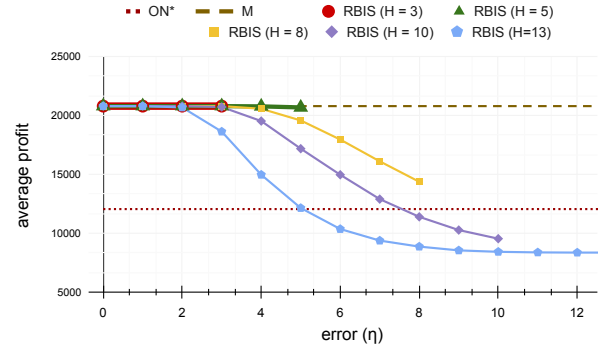
η_0, the following process is repeated 1000 times for Alg. First, the (correct) responses to the 25 queries asked by Alg are generated; then out of these 25 responses, η_0 of them are selected uniformly at random, and flipped. This is the prediction P that is given to Alg; we run Alg with this prediction, and record its profit. After running 1000 tests, the average value of the reported profits is recorded as the average profit of Alg for I_x, for a value of error equal to η_0.

Figures 4 and 5 depict the average profit (as a function of η) for RLIS and RBIS, respectively, for inputs generated from the `Bitcoin-to-USD` benchmark. Since this is a non-oblivious setting, the profit is only reported for values of $\eta \leq H$. We observe that both algorithms attain profit significantly better than ON* for reasonable values of error, and their profit degrades gently with the error. In particular, RBIS with $H \in \{3, 5\}$ accrues an optimal profit. For a fixed value of η, smaller values of H yield to better profit for both algorithms. This is consistent with Theorems 5 and 6, which bound the competitive ratios as an increasing function of H. We also observe that RBIS performs better than RLIS, which is again consistent with Theorems 5 and 6. We also observe that even if H is relatively large (e.g., $H = 8$), RBIS results in better profit in comparison to ON*.

5 Conclusion

We gave the first theoretical study, with supporting experimental evaluation over real data, of a fundamental problem in online decision making, and in a learning-augmented setting. Despite the simplicity of the problem in its standard version, the learning-augmented setting is quite complex and poses several challenges. Future work should expand the ideas in this work to generalizations of online search such such as one-way trading and online portfolio selection.

Our robust binary search algorithm can be useful in other query-based optimization settings, with or without predictions, since it addresses a broad setting: select a "good" candidate, using noisy queries, while maximizing the size of the candidate space (exponential in the number of queries).

Acknowledgments

References

Ahmad, I.; Pirron, M.; and Schmidt, G. 2021. Analysis of threat based algorithm using different performance measures. *RAIRO: Recherche Opérationnelle*, 55: 2393.

Anand, K.; Ge, R.; and Panigrahi, D. 2020. Customizing ML Predictions for Online Algorithms. In *International Conference on Machine Learning (ICML)*, 303–313. PMLR.

Angelopoulos, S.; and Kamali, S. 2021. Contract Scheduling With Predictions. In *35th AAAI Conference on Artificial Intelligence*, 11726–11733. AAAI Press.

Angelopoulos, S.; Kamali, S.; and Zhang, D. 2021. Online Search With Best-Price and Query-Based Predictions. *CoRR*, abs/2112.01592.

Antoniadis, A.; Coester, C.; Eliás, M.; Polak, A.; and Simon, B. 2020a. Online metric algorithms with untrusted predictions. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 345–355.

Antoniadis, A.; Gouleakis, T.; Kleer, P.; and Kolev, P. 2020b. Secretary and Online Matching Problems with Machine Learned Advice. In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*.

Banerjee, S. 2020. Improving Online Rent-or-Buy Algorithms with Sequential Decision Making and ML Predictions. In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*.

Borodin, A.; El-Yaniv, R.; and Gogan, V. 2000. On the competitive theory and practice of portfolio selection. In *Latin American symposium on theoretical informatics*, 173–196. Springer.

Boyar, J.; Larsen, K. S.; and Maiti, A. 2014. A comparison of performance measures via online search. *Theoretical Computer Science*, 532: 2–13.

Cicalese, F. 2013. *Fault-Tolerant Search Algorithms - Reliable Computation with Unreliable Information*. Monographs in Theoretical Computer Science. An EATCS Series. Springer.

Clemente, J.; Hromkovič, J.; Komm, D.; and Kudahl, C. 2016. Advice complexity of the online search problem. In *International Workshop on Combinatorial Algorithms*, 203–212. Springer.

Cover, T. M.; and Ordentlich, E. 1996. Universal portfolios with side information. *IEEE Transactions on Information Theory*, 42(2): 348–363.

Damaschke, P.; Ha, P. H.; and Tsigas, P. 2009. Online search with time-varying price bounds. *Algorithmica*, 55(4): 619–642.

Das, P.; Johnson, N.; and Banerjee, A. 2014. Online portfolio selection with group sparsity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Disser, Y.; and Kratsch, S. 2017. Robust and Adaptive Search. In Vollmer, H.; and Vallée, B., eds., *Proc. 34th Symposium on Theoretical Aspects of Computer Science, STACS*, volume 66 of *LIPIcs*, 26:1–26:14.

EA Trading Academy. 2021. Currency exchange rates. https://eatradingacademy.com/software/forex-historical-data/. Accessed: 2021-09-05.

El-Yaniv, R. 1998. Competitive solutions for online financial problems. *ACM Computing Surveys (CSUR)*, 30(1): 28–69.

El-Yaniv, R.; Fiat, A.; Karp, R. M.; and Turpin, G. 2001. Optimal search and one-way trading online algorithms. *Algorithmica*, 30(1): 101–139.

Fujiwara, H.; Iwama, K.; and Sekiguchi, Y. 2011. Average-case competitive analyses for one-way trading. *Journal of combinatorial optimization*, 21(1): 83–107.

Gollapudi, S.; and Panigrahi, D. 2019. Online algorithms for rent-or-buy with expert advice. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2319–2327.

Hazan, E.; and Kale, S. 2015. An online portfolio selection algorithm with regret logarithmic in price variation. *Mathematical Finance*, 25(2): 288–310.

Hazan, E.; and Megiddo, N. 2007. Online learning with prior knowledge. In *International Conference on Computational Learning Theory*, 499–513. Springer.

Lattanzi, S.; Lavastida, T.; Moseley, B.; and Vassilvitskii, S. 2020. Online scheduling via learned weights. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1859–1877.

Lavastida, T.; Moseley, B.; Ravi, R.; and Xu, C. 2020. Learnable and Instance-Robust Predictions for Online Matching, Flows and Load Balancing. *CoRR*, abs/2011.11743.

Li, B.; and Hoi, S. C. 2014. Online portfolio selection: A survey. *ACM Computing Surveys (CSUR)*, 46(3): 1–36.

Lorenz, J.; Panagiotou, K.; and Steger, A. 2009. Optimal algorithms for k-search with application in option pricing. *Algorithmica*, 55(2): 311–328.

Lykouris, T.; and Vassilvitskii, S. 2018. Competitive Caching with Machine Learned Advice. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 3302–3311.

Mitzenmacher, M.; and Vassilvitskii, S. 2020. Algorithms with Predictions. In Roughgarden, T., ed., *Beyond the Worst-Case Analysis of Algorithms*, 646–662. Cambridge University Press.

Mohr, E.; Ahmad, I.; and Schmidt, G. 2014. Online algorithms for conversion problems: a survey. *Surveys in Operations Research and Management Science*, 19(2): 87–104.

Purohit, M.; Svitkina, Z.; and Kumar, R. 2018. Improving Online Algorithms via ML Predictions. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS)*, volume 31, 9661–9670.

Rivest, R. L.; Meyer, A. R.; Kleitman, D. J.; Winklmann, K.; and Spencer, J. 1980. Coping with Errors in Binary Search Procedures. *J. Comput. Syst. Sci.*, 20(3): 396–404.

Rohatgi, D. 2020. Near-optimal bounds for online caching with machine learned advice. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1834–1845.

Uziel, G.; and El-Yaniv, R. 2020. Long-and Short-Term Forecasting for Portfolio Selection with Transaction Costs. In *International Conference on Artificial Intelligence and Statistics*, 100–110. PMLR.

Wei, A.; and Zhang, F. 2020. Optimal Robustness-Consistency Trade-offs for Learning-Augmented Online Algorithms. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems (NeurIPS)*.

Xu, Y.; Zhang, W.; and Zheng, F. 2011. Optimal algorithms for the online time series search problem. *Theoretical Computer Science*, 412(3): 192–197.