Online Search with Predictions: Pareto-optimal Algorithm and its Applications in Energy Markets

Russell Lee*
University of Massachusetts Amherst
rclee@cs.umass.edu

Bo Sun*
University of Waterloo
bo.sun@uwaterloo.ca

Mohammad Hajiesmaili
University of Massachusetts Amherst
hajiesmaili@cs.umass.edu

John C.S. Lui
The Chinese University of Hong Kong
cslui@cse.cuhk.edu.hk

ABSTRACT

This paper develops learning-augmented algorithms for energy trading in volatile electricity markets. The basic problem is to sell (or buy) k units of energy for the highest revenue (lowest cost) over uncertain time-varying prices, which can framed as a classic online search problem in the literature of competitive analysis. State-of-the-art algorithms assume no knowledge about future market prices when they make trading decisions in each time slot, and aim for guaranteeing the performance for the worst-case price sequence. In practice, however, predictions about future prices become commonly available by leveraging machine learning. This paper aims to incorporate machine-learned predictions to design competitive algorithms for online search problems. An important property of our algorithms is that they achieve performances competitive with the offline algorithm in hindsight when the predictions are accurate (i.e., consistency) and also provide worst-case guarantees when the predictions are arbitrarily wrong (i.e., robustness). The proposed algorithms achieve the Pareto-optimal trade-off between consistency and robustness, where no other algorithms for online search can improve on the consistency for a given robustness. Further, we extend the basic online search problem to a more general inventory management setting that can capture storage-assisted energy trading in electricity markets. In empirical evaluations using traces from real-world applications, our learning-augmented algorithms improve the average empirical performance compared to benchmark algorithms, while also providing improved worst-case performance.

CCS CONCEPTS

• **Theory of computation** → **Online algorithms**; *Online learning algorithms*; • **Hardware** → **Energy generation and storage**; *Batteries*;

KEYWORDS

Learning-augmented algorithms, online algorithms, energy storage, Pareto-optimality

ACM Reference Format:
Russell Lee, Bo Sun, Mohammad Hajiesmaili, and John C.S. Lui. 2024. Online Search with Predictions: Pareto-optimal Algorithm and its Applications in Energy Markets . In *The 15th ACM International Conference on Future and Sustainable Energy Systems (E-Energy '24), June 04–07, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 22 pages. https://doi.org/10.1145/3632775.3639590

1 INTRODUCTION

With the increasing penetration of renewable energy in the supply side and distributed energy resources in the demand side, the electricity market has become increasingly volatile. To address the uncertainties underlying both electricity prices and energy demands in modern smart grids, competitive algorithms [5] have been widely used for optimizing worst-case performances, such as energy scheduling in Microgrids [21], storage management [22], electric vehicle pricing and scheduling [6], and beyond.

In this paper, we study competitive algorithms for energy trading problems based on a classic online search model. In this problem, an online decision-maker aims to sell (buy) $k \geq 1$ units of energy for the highest revenue (lowest cost) over a sequence of time-varying prices. At each step, a price is observed, and the decision-maker wants to find how many units to sell (buy) at the current price without knowing the future prices. In online decision-making, a key challenge is to balance the revenue (cost) of trading at the current price, and deferring for future better prices at the risk that those prices never arrive. The online search problem is foundational for modeling online decision-making in volatile markets such as asset trading in financial markets [8, 18, 24], energy arbitrage in electricity markets [27], and revenue management in flights market [4].

The online search problem has been tackled previously under the framework of competitive analysis [5]. In this framework, the ultimate goal is to design online algorithms with the best possible *competitive ratio*, defined as the ratio between the revenue (cost) of the offline optimum and an online algorithm. Lorenz et al. [18] proposed two online threshold-based algorithms for both maximization and minimization versions of the online search problem. The algorithm predetermines k threshold values and trades the i-th unit of the asset only if the price is at least (at most) equal to the i-th threshold value in the k-max (k-min) search. Then, by optimizing

*Both authors contributed equally to this research.

E-Energy '24, June 04–07, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0480-2/24/06
https://doi.org/10.1145/3632775.3639590

the threshold values, the online algorithms can achieve the optimal competitive ratios for both versions of k-search. The original algorithm in [18] requires that at most one unit asset can be traded in each step. Then the follow-up work [29] extends it to a setting that allows for trading multiple units in each step with a slight modification on the algorithm by Lorenz et al. [18] using the same optimal threshold values. As k goes to infinity, the k-search model in [29] asymptotically becomes a continuous search problem, and has been framed as the one-way trading problem by the seminal work [8]. Our paper focuses on the setting by [29] that includes the continuous trading problem as a special case. Further, we extend the problem to an inventory management setting that is applicable to online search problems with inventory dynamics.

The classic online algorithms designed purely with guarantees of the worst-case performance tend to ignore predictions outright, and thus they often have poor performance in common average-case scenarios. In practice, however, for most application scenarios, abundant historical data could be leveraged by machine learning (ML) tools for generating some predictions of the unknown future input, e.g., prices in k-search. The possibility of incorporating ML predictions in algorithmic design has led to the recent development of learning-augmented online algorithms [19, 23], where the goal is to leverage predictions to improve the performance when predictions are accurate and preserve the robust worst-case guarantees when facing erroneous ML predictions. This high-level idea has led researchers to revisit a wide range of online problems, including but not limited to caching [19], rent-or-buy problems [12, 17, 23, 26], facility location [10, 15], secretary matching [3, 7], metrical task systems [2], and bin packing [1, 28].

Motivated by the above direction of online algorithms with predictions, this paper aims to design learning-augmented algorithms for online search problems. This goal is particularly crucial for applications in volatile markets where ML predictions are useful when they are accurate but can frequently advise the wrong direction, resulting in drastic losses relative to the best trade in hindsight.

We design and analyze a learning-augmented algorithm under the consistency-robustness framework [19], where consistency represents the competitive ratio when the prediction is accurate, and robustness is the competitive ratio regardless of the prediction error. Our goal is to design an algorithm that can achieve the Pareto-optimal trade-off between consistency and robustness, i.e., no other learning-augmented algorithms can simultaneously achieve better consistency and robustness than ours. Although learning-augmented algorithms have been an active research topic in recent years, the majority of prior work focuses on algorithms that can provide bounded consistency and robustness. However, studies about the Pareto-optimality of the trade-off are limited, with a few exceptions, e.g., ski-rental problem [26], online conversion problem [24], online matching problem [16] and single-leg revenue management problem [4]. This paper contributes to this line of research for the online search problem.

1.1 Contributions

Pareto-optimal algorithms for online k-search problems with predictions. We design learning-augmented algorithms for k-max and k-min search, and prove that their robustness and consistency are Pareto-optimal. To achieve this, we start by deriving lower bound results on the trade-off between consistency and robustness of any learning-augmented algorithms. We then use the lower bound trade-off as the target of our algorithms. By leveraging ML predictions, we redesign the threshold values in the classic algorithms to prioritize the trading in cases predicted to occur for achieving good consistency while also carefully reserving sufficient trading opportunities for other cases to guarantee robustness. Finally, we prove the target trade-off is achievable by our design.

We note that a closely related work [24] has reported Pareto-optimal algorithms for 1-max search and one-way trading with predictions. Our proposed algorithm solves a more general k-search problem that includes 1-max search (when $k = 1$) and one-way trading ($k \to \infty$) as two special cases. In addition, we also tackle the minimization setting of k-search, which is demonstrated to have different performance guarantees from k-max search. Thus, this paper studies the full spectrum of k-search problems with predictions. In Figure 1, we show the Pareto boundaries achieved by our algorithms, including the results in [24] as two special curves.

Extensions to online search with inventory dynamics. In many real-world applications, the demand for trading (i.e., k) is unknown and is only revealed to the decision-maker over time. To accommodate the online demand together with time-varying prices, we consider an online search problem assisted by an inventory; however, the inventory dynamics strongly couples the decisions over time, adding extra challenges to online decision-making. The presence of inventory also adds a new design space for buying and storing items in inventory when the price is low and using the stored items when the market price is not attractive. Despite the temporal coupling and more involved design space, we observe that the online search with inventory dynamics can be decoupled "spatially" into multiple virtual online search problems in parallel, and an independent learning-augmented algorithm can solve each of them. Based on this observation, we extend the learning-augmented algorithms to solve this more general inventory management problem and show that the algorithm preserves the Pareto optimality when the inventory capacity is sufficiently large.

Applications in energy markets. We demonstrate the performance of our learning-augmented algorithms with numerical experiments for storage-assisted energy procurement in electricity markets. In the empirical experiments, we use energy data traces from Akamai data centers, renewable outputs from NREL, and energy prices from several ISOs. Our proposed algorithms improve the average empirical performance compared to worst-case optimized algorithms and other baseline learning-augmented algorithms. Moreover, our algorithms are shown to provide improved worst-case performance even when the predictions are with relatively large errors. Thus, our algorithms can potentially achieve best of both worlds.

2 PROBLEM STATEMENT

2.1 Online k-search problem

The k-max (k-min) search problem aims to sell (buy) k units of identical items over a sequence of T prices to maximize the total revenue (to minimize the total cost). In the online setting, the T prices arrive one at a time. Upon the arrival of price p_t, a decision-maker must

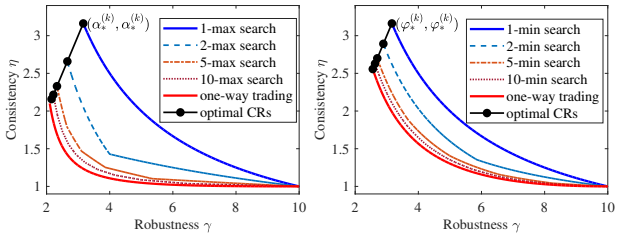


Figure 1: Illustrating Pareto-optimal boundaries of k-max search and k-min search. All curves start from a point $(\alpha_*^{(k)}, \alpha_*^{(k)})$ (or $(\varphi_*^{(k)}, \varphi^{(k)})_*$) with both consistency and robustness equal to the corresponding worst-case optimal competitive ratio (CR), and end at the point $(\theta, 1)$. We set $p_{\max} = 50$, $p_{\min} = 5$ and $\theta = p_{\max}/p_{\min} = 10$, where $p_{\max}$ and $p_{\min}$ are upper and lower bounds of prices over the time horizon.

immediately decide how many items to trade, $x_t \in \{0, 1, \ldots, k\}$, without knowing the future prices. If the decision-maker has only traded m ($m < k$) items after the arrival of the $(T-1)$-th price, she is compelled to trade all the remaining $k-m$ items at the last price p_T. We make no assumptions on the underlying distribution of the prices and only assume they are bounded within known limits $p_{\min}$ and $p_{\max}$, i.e., $p_t \in [p_{\min}, p_{\max}], \forall t \in [T]$. This is a standard assumption for designing online algorithms with bounded competitive ratios in the literature of online search problems [8, 18]. The price fluctuation ratio is defined as $\theta = p_{\max}/p_{\min}$.

Define the price sequence $\mathcal{I} := \{p_t\}_{t\in[T]}$ as an instance of the k-search problem. Let $\texttt{ALG}(\mathcal{I})$ denote the objective value of the k-search problem from an online algorithm. Under the framework of competitive analysis [5], we aim to design the online algorithm such that its performance is competitive with that of an offline algorithm in hindsight. In particular, if an instance $\mathcal{I}$ is given from the start, the k-max (k-min) search problem can be formulated as

$$\text{max (min)} \quad \sum\nolimits_{t\in[T]} p_t x_t, \tag{1a}$$

$$\text{subject to} \quad \sum\nolimits_{t\in[T]} x_t = k, \tag{1b}$$

$$\text{variable} \quad x_t \in \{0, 1, \ldots, k\}, \forall t \in [T]. \tag{1c}$$

Let $\texttt{OPT}(\mathcal{I})$ denote the optimal objective of above optimization problem. We evaluate the performance of an online algorithm by its competitive ratio, which is defined as

$$\alpha = \max_{\mathcal{I}} \frac{\texttt{OPT}(\mathcal{I})}{\texttt{ALG}(\mathcal{I})} \text{ and } \varphi = \max_{\mathcal{I}} \frac{\texttt{ALG}(\mathcal{I})}{\texttt{OPT}(\mathcal{I})}, \tag{2}$$

for k-max search and k-min search, respectively. Both α and φ are greater than one; the smaller their value, the better the performance of the online algorithm.

2.2 Worst-case optimized algorithms

The k-search problem can be solved by an online threshold-based algorithm (OTA) as shown in Algorithm 1. This algorithm takes k threshold values $\Phi := \{\Phi_i\}_{i\in[k]}$ in k-max search ($\Psi := \{\Psi_i\}_{i\in[k]}$ in k-min search) as its input, and trades the i-th item only if the current price is at least Φ_i (at most Ψ_i). Let $\texttt{OTA}_\Phi$ denote the OTA with threshold Φ. $\texttt{OTA}_\Phi$ can achieve the smallest competitive ratios

Algorithm 1 $\texttt{OTA}_\Phi$ for k-max ($\texttt{OTA}_\Psi$ for k-min)

input: threshold values $\Phi = \{\Phi_i\}_{i\in[k]}$ ($\Psi = \{\Psi_i\}_{i\in[k]}$);
initialization: $m = 1$;
for step $t = 1, \ldots, T-1$ **do**
  $x_t = 0$;
  while $p_t \geq \Phi_m$ ($p_t \leq \Psi_m$) and $m \leq k$ **do**
    $m = m + 1$ and $x_t = x_t + 1$;
  end while
end for
$x_T = k - m + 1$.

for k-search problems when the thresholds are designed optimally. Based on the analysis in [18, 29], we have the following lemmas.

Lemma 2.1 (k-max search). OTA *for k-max search is $\alpha_*^{(k)}$-competitive if the threshold values are*

$$\Phi_i = p_{\min}\left[1 + (\alpha_*^{(k)} - 1)\left(1 + \alpha_*^{(k)}/k\right)^{i-1}\right], i \in [k], \tag{3}$$

where $\alpha_*^{(k)}$ is the solution of $\frac{\theta-1}{\alpha-1} = \left(1 + \frac{\alpha}{k}\right)^k$.

Lemma 2.2 (k-min search). OTA *for k-min search is $\varphi_*^{(k)}$-competitive if the threshold values are*

$$\Psi_i = p_{\max}\left[1 - \left(1 - 1/\varphi_*^{(k)}\right)\left(1 + 1/(k\varphi_*^{(k)})\right)^{i-1}\right], i \in [k], \tag{4}$$

where $\varphi_*^{(k)}$ is the solution of $\frac{1-1/\theta}{1-1/\varphi} = \left(1 + \frac{1}{k\varphi}\right)^k$.

In the following, we sketch the key intuitions in the design and analysis of the threshold values, which are important for designing learning-augmented algorithms in the next section. Consider the threshold values Φ for the k-max search. The threshold values are monotonically non-decreasing because the algorithm must aggressively trade items in the beginning to hedge the risk that values of the future prices will all drop to the lowest $p_{\min}$. As more items are traded, the algorithm can gradually become more selective by choosing higher threshold values and taking the opportunity to trade at potentially high prices. Since the uncertain prices vary within $[p_{\min}, p_{\max}]$, the k threshold values divide this range into $k+1$ disjoint intervals $[\Phi_0, \Phi_1), [\Phi_1, \Phi_2), \ldots, [\Phi_{k-1}, \Phi_k), [\Phi_k, \Phi_{k+1}]$, where $\Phi_0 := p_{\min}$ and $\Phi_{k+1} := p_{\max}$. Suppose the highest price falls in an interval $[\Phi_{i-1}, \Phi_i)$, the total revenue of the offline algorithm is upper bounded by $k\Phi_i$, and the revenue of the online algorithm is lower bounded by $\sum_{j\in[i-1]} \Phi_j + (k-i+1)p_{\min}$, where the first $i-1$ items are traded at prices just equal to the first $i-1$ thresholds and the remaining items are compulsorily traded at the lowest price $p_{\min}$. In this case, the ratio of the offline and online revenues is upper bounded by

$$\alpha_i^{(k)}(\Phi) = \frac{k\Phi_i}{\sum_{j\in[i-1]} \Phi_j + (k-i+1)p_{\min}}. \tag{5}$$

Note that Equation (5) holds for all $i \in [k+1]$ when the maximum price falls in the corresponding $k+1$ intervals. Thus, the worst-case ratio of the k-max search is $\alpha_*^{(k)} = \max_{i\in[k+1]} \alpha_i^{(k)}$ and the minimum is achieved when the ratios from different intervals are balanced, i.e., $\alpha_*^{(k)} = \alpha_i^{(k)}, \forall i \in [k+1]$. This gives $k+1$ equations

with k unknown thresholds and one unknown ratio $\alpha_*^{(k)}$. Solving those equations gives the thresholds and the competitive ratio $\alpha_*^{(k)}$ for k-max search in Lemma 2.1. We can apply similar approaches to design thresholds and $\varphi_*^{(k)}$ for k-min search in Lemma 2.2.

2.3 Learning-augmented algorithms

Prediction model. In the k-max (k-min) search, we consider a prediction $P \in [p_{\min}, p_{\max}]$ of the actual highest (lowest) price that can be obtained from ML tools. Define $\varepsilon = |V - P|$ as the error of the ML prediction, where V represents the actual highest (lowest) price. We aim to augment the worst-case optimized algorithm with the prediction P. Notably, we make no assumptions about the quality of the ML predictions, and thus the prediction error ε is unknown to the algorithm. To evaluate the performance of the algorithm with such "untrusted" predictions, we focus on two metrics: (i) consistency η, which is the competitive ratio when the prediction is accurate, i.e., $\varepsilon = 0$; and (ii) robustness γ, which is the competitive ratio regardless of the prediction error. Therefore, consistency and robustness measure the algorithm's performance when the prediction is of good quality and arbitrarily bad, respectively. Our goal is to design the algorithm that can achieve the Pareto-optimal trade-off between consistency and robustness, i.e., for a given robustness γ, no other online algorithms can achieve a smaller consistency η. This consistency-robustness framework was first introduced by [19] to study the online caching problem and then has been widely adopted for designing online algorithms with untrusted predictions [4, 16, 17, 23].

Baseline algorithm. To design learning-augmented algorithms with bounded consistency and robustness for online resource allocation, one natural approach is to divide the limited resource into two portions, and then simultaneously run the robust algorithm and the prediction-based algorithm using the two portions, respectively (see examples in [13, 14]). Let $\lambda \in [0, 1]$ be a hyper-parameter that indicates the degree of untrust in the prediction. The baseline algorithm divides the k items into two portions, $k_r = \lceil \lambda k \rceil$ and $k_c = k - k_r$. The algorithm, in parallel, runs a worst-case optimized k_r-search algorithm and a prediction-based algorithm that trades all k_c items at the first price no smaller than the predicted price P. The decision of the baseline algorithm is the sum of the decisions from the worst-case algorithm and the prediction-based algorithm.

Lemma 2.3. *Given a hyper-parameter $\lambda \in [0, 1]$, the baseline algorithm is $\frac{k}{k_r/\alpha_*^{(k_r)} + k - k_r}$-consistent and $\frac{k}{k_r/\alpha_*^{(k_r)} + (k-k_r)/\theta}$-robust for the learning-augmented k-max search problem, where $k_r := \lceil \lambda k \rceil$.*

Although bounded consistency and robustness can be guaranteed by the intuitive baseline algorithm, there exists a performance gap between the baseline algorithm and the Pareto-optimal algorithm as shown in Figure 2. This motivates us to design the Pareto-optimal algorithm for k-search to close the theoretical gap. In addition, we will show that the Pareto-optimal algorithm empirically outperforms the baseline algorithm in experiments using real data (see Section 5), which is of significance in real-world applications. The proof of Lemma 2.3 and the consistency-robustness result of the baseline algorithm for k-min search are given in Appendix A.

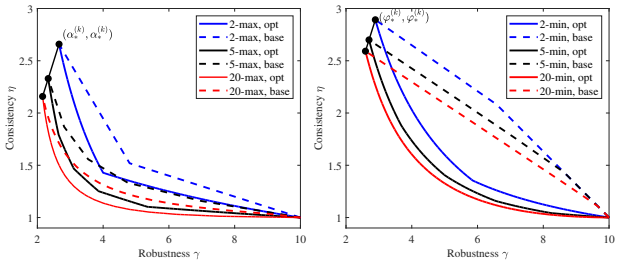


Figure 2: Comparing consistency-robustness trade-offs of baseline algorithms and Pareto-optimal algorithms for k-max and k-min search. $p_{\max} = 50$ and $p_{\min} = 5$.

3 PARETO-OPTIMAL ALGORITHMS WITH PREDICTION FOR k-SEARCH

In this section, we design learning-augmented algorithms for k-max search and k-min search, and prove that they can attain the Pareto-optimal trade-off between consistency and robustness. We first focus on the k-max search in detail and then state the main results for k-min search and defer the details to Appendix B.

In k-max search, we have a prediction of the maximum price $P \in [p_{\min}, p_{\max}]$. Our learning-augmented algorithm is to redesign the threshold values $\phi(k, P) := \phi = \{\phi_i\}_{i \in [k]}$ in OTA based on the prediction P such that $\mathtt{OTA}_\phi$ can attain the Pareto-optimal consistency and robustness. To achieve this goal, we first determine a lower bound for the robustness-consistency trade-off. Then, ϕ is designed such that $\mathtt{OTA}_\phi$ can achieve this lower bound.

3.1 Learning-augmented algorithms for k-max

3.1.1 Lower bound. We first present a lower bound for any learning-augmented algorithm for the k-max problem, which is used as a target trade-off in our algorithm design.

Theorem 3.1. *For k-max search, any γ-robust deterministic learning-augmented algorithm must have a consistency lower bounded by*

$$\Gamma(\gamma) = \frac{\theta}{\left[1 + (\gamma - 1)(1 + \frac{\gamma}{k})^{\xi}\right]/\gamma + (\theta - 1)(1 - \frac{\xi}{k})}, \tag{6}$$

where $\xi = \left\lceil \ln\left(\frac{\theta-1}{\gamma-1}\right) / \ln\left(1 + \frac{\gamma}{k}\right) \right\rceil$.

The lower bound result of the k-max search in Theorem 3.1 generalizes the existing results of 1-max search and one-way trading in [24]. When $k \to \infty$, we have $\xi/k \to \frac{1}{\gamma} \ln \frac{\theta-1}{\gamma-1}$ and $(1 + \frac{\gamma}{k})^{\xi} \to \frac{\theta-1}{\gamma-1}$. Thus, the lower bound of k-max search approaches that of one-way trading. When $k = 1$, we have $\xi = 1$. The lower bound of k-max search approaches that of 1-max search.

Proof of Theorem 3.1. To show the lower bound, we consider a special family of instances, and show that under the special instances, any γ-robust deterministic online algorithm at least has a consistency η that is lower bounded by $\Gamma(\gamma)$. □

Definition 3.2 (p-instance). *A p-instance $\mathcal{I}_p$ consists of a sequence of prices that increase continuously from $p_{\min}$ to p and drop to $p_{\min}$ at the last price.*

Let $g(p) : [p_{\min}, p_{\max}] \to \{0, 1, \ldots, k\}$ denote the cumulative trading decision of an online algorithm when it executes the instance $\mathcal{I}_p$ before the compulsory trading in the last step. Since the online decision is irrevocable, $g(p)$ is non-decreasing as p increases from $p_{\min}$ to $p_{\max}$. In addition, items must be traded if the price is the highest one $p_{\max}$. Thus, we must have $g(p_{\max}) = k$. Given an online algorithm, let $\mathcal{I}_{\hat{p}_i}$ denote the first instance, in which the algorithm trades the i-th item, i.e., $\hat{p}_i = \inf_{\{p \in [p_{\min}, p_{\max}] : g(p) \geq i\}} p$.

For any γ-robust online algorithm, $\{\hat{p}_i\}_{i\in[k]}$ must satisfy

$$\frac{k\hat{p}_i}{\sum_{j\in[i-1]} \hat{p}_j + (k-i+1)p_{\min}} \leq \gamma, i \in [k+1], \tag{7a}$$

$$\hat{p}_i \leq p_{\max}, i \in [k], \tag{7b}$$

where $\hat{p}_0 := p_{\min}$ and $\hat{p}_{k+1} := p_{\max}$. Based on Equation (7), we have

$$\hat{p}_i \leq \min\left\{p_{\min} + p_{\min}(\gamma - 1)\left(1 + \frac{\gamma}{k}\right)^{i-1}, p_{\max}\right\}, \forall i \in [k]. \tag{8}$$

Suppose the prediction is given by $P = p_{\max}$. To ensure η-consistency under $\mathcal{I}_{p_{\max}}$, any γ-robust algorithm must have

$$\eta \geq \frac{\mathtt{OPT}(\mathcal{I}_{p_{\max}})}{\mathtt{ALG}(\mathcal{I}_{p_{\max}})} = \frac{kp_{\max}}{\sum_{i\in[k]} \hat{p}_i} = \frac{kp_{\max}}{\sum_{i\in[\xi]} \hat{p}_i + (k-\xi)p_{\max}} \tag{9a}$$

$$\geq \frac{kp_{\max}}{\sum_{i\in[\xi]} p_{\min}[1 + (\gamma-1)(1+\frac{\gamma}{k})^{i-1}] + (k-\xi)p_{\max}} \tag{9b}$$

$$= \frac{\theta}{\left[1 + (\gamma-1)(1+\frac{\gamma}{k})^{\xi}\right]/\gamma + (\theta-1)(1-\frac{\xi}{k})}, \tag{9c}$$

where the inequality (9b) holds due to Equation (8) enforced by γ-robustness, and ξ satisfies $p_{\min} + p_{\min}(\gamma-1)(1+\frac{\gamma}{k})^{\xi-1} < p_{\max} \leq p_{\min} + p_{\min}(\gamma-1)(1+\frac{\gamma}{k})^{\xi}$, which gives $\xi := \left\lceil \ln\left(\frac{\theta-1}{\gamma-1}\right)/\ln\left(1+\frac{\gamma}{k}\right)\right\rceil$. This completes the lower bound proof.

3.1.2 Pareto-optimal algorithm. Based on the lower bound result, for a given $\lambda \in [0, 1]$, we set our target consistency and robustness of k-max search with predictions as

$$\gamma^{(k)}(\lambda) = \alpha_*^{(k)} + (1-\lambda)(\theta - \alpha_*^{(k)}),\ \eta^{(k)}(\lambda) = \Gamma(\gamma^{(k)}(\lambda)). \tag{10}$$

where λ is the confidence factor that indicates the degree of untrust in the prediction. Our goal is to design threshold value $\phi(k, P) := \{\phi_i\}_{i\in[k]}$ that depends on k and prediction P such that $\mathtt{OTA}_\phi$ can achieve the target in Equation (10). In particular, the threshold value ϕ is designed in the form of

$$\phi_i = \begin{cases} z_i & i = 1, \ldots, j^*, \\ c_i & i = j^*+1, \ldots, i^*, \\ r_i & i = i^*+1, \ldots, k, \end{cases} \tag{11}$$

where the two sequences $\{z_i\}_{i\in[j^*]}$ and $\{r_i\}_{i=i^*+1}^{k}$ are designed to guarantee the robustness and the sequence $\{c_i\}_{i=j^*+1}^{i^*}$ is designed to ensure the consistency. Recall the threshold values divide the uncertainty price range $[p_{\min}, p_{\max}]$ into $k+1$ regions and the worst-case ratio when the actual maximum price falls in each region is $\alpha_i^{(k)}(\phi)$ derived in Equation (5). We design ϕ such that

$$\alpha_i^{(k)}(\phi) \leq \begin{cases} \eta^{(k)}, & i = j^*+1, \ldots, i^*, \\ \gamma^{(k)}, & \text{otherwise}, \end{cases} \tag{12}$$

when prediction $P \in [\phi_{j^*+1}, \phi_{i^*}]$, and thus an accurate prediction will lead to a consistency $\eta^{(k)}$.

Design of threshold values. Given a prediction P of the maximum price and the target consistency η and robustness γ, we design a piecewise threshold ϕ including three cases as follows.

Let $\sigma^* \in [k]$ be the largest index such that $\frac{1+(\theta-1)/(1+\gamma/k)^{k-\sigma^*}}{1+(\eta-1)(1+\eta/k)^{\sigma^*}} \leq \frac{\gamma}{\eta}$. We define two boundary values $\tilde{p}_1 = p_{\min} + p_{\min}(\eta-1)(1+\eta/k)^{\sigma^*-1}$ and $\tilde{p}_2 = \max\{\tilde{p}_1, \gamma p_{\min}\}$.

Case I. When $P \in [p_{\min}, \tilde{p}_1]$, set $j^* = 0$, $i^* = \sigma^*$ and

$$c_i = p_{\min} + p_{\min}(\eta-1)\,(1+\eta/k)^{i-1}, i \in [i^*], \tag{13a}$$

$$r_i = p_{\min} + \frac{p_{\max} - p_{\min}}{(1+\gamma/k)^{k-i+1}}, i = i^*+1, \ldots, k. \tag{13b}$$

Case II. When $P \in (\tilde{p}_1, \tilde{p}_2]$, set $j^* = 0$ and

$$c_i = \begin{cases} P & i = 1, \ldots, m^*, \\ \eta\frac{m^*P + (k-m^*)p_{\min}}{k} & i = m^*+1, \\ p_{\min} + (c_{m^*+1} - p_{\min})(1+\eta/k)^{i-m^*-1} & i = m^*+2, \ldots, i^*, \end{cases} \tag{14a}$$

$$r_i = p_{\min} + \frac{p_{\max} - p_{\min}}{(1+\gamma/k)^{k-i+1}}, \quad i = i^*+1, \ldots, k, \tag{14b}$$

where $m^* = \left\lceil k\frac{P/\eta - p_{\min}}{P - p_{\min}}\right\rceil$ and i^* is the largest index that ensures $\alpha_{i^*+1}^{(k)}(\phi) = \frac{kr_{i^*+1}}{\sum_{i=[i^*]} c_i + (k-i^*)p_{\min}} \leq \gamma$.

Case III. When $P \in (\tilde{p}_2, p_{\max}]$, set

$$z_i = p_{\min} + p_{\min}(\gamma-1)(1+\gamma/k)^{i-1}, \quad i \in [j^*], \tag{15a}$$

$$c_i = \begin{cases} P, & i = j^*+1, \ldots, m^*, \\ \eta\frac{\sum_{i\in[j^*]} z_i + (m^*-j^*)P + (k-m^*)p_{\min}}{k}, & i = m^*+1, \\ p_{\min} + (c_{m^*+1} - p_{\min})(1+\eta/k)^{i-m^*-1}, & i = m^*+2, \ldots, i^*, \end{cases} \tag{15b}$$

$$r_i = p_{\min} + \frac{p_{\max} - p_{\min}}{(1+\gamma/k)^{k-i+1}}, \quad i = i^*+1, \ldots, k, \tag{15c}$$

where we have

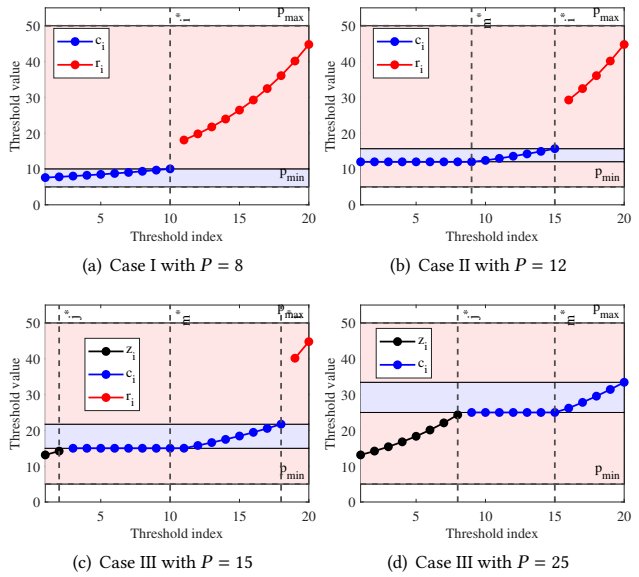
$$j^* = \left\lceil \ln\left(\frac{P/p_{\min} - 1}{\gamma - 1}\right)/\ln\left(1+\frac{\gamma}{k}\right)\right\rceil$$

$$m^* = j^* + \left\lceil \frac{kP/\eta - kp_{\min}[1 + (\gamma-1)(1+\gamma/k)^{j^*}]/\gamma}{P - p_{\min}}\right\rceil$$

and i^* is the largest index such that $\alpha_{i^*+1}^{(k)}(\phi) \leq \gamma$.

In Figure 3, we illustrate the design of the threshold ϕ in above three cases with varying predictions. $\mathtt{OTA}_\phi$ can ensure that the worst-case ratio is upper bounded by η if the actual maximum price falls in the shaded blue region and upper bounded by γ if the actual maximum price belongs to the shaded pink region. Also note that the shaded blue region always contains the prediction P. Therefore, $\mathtt{OTA}_\phi$ is η-consistent and γ-robust.

Theorem 3.3. *Given $\lambda \in [0, 1]$, $\mathtt{OTA}_\phi$ is $\eta^{(k)}(\lambda)$-consistent and $\gamma^{(k)}(\lambda)$-robust for the learning-augmented k-max search when ϕ is given by Equations* (13)-(15)*, where $\eta = \eta^{(k)}(\lambda)$ and $\gamma = \gamma^{(k)}(\lambda)$ are given by Equation* (10)*. Further, $\mathtt{OTA}_\phi$ is Pareto-optimal.*



(a) Case I with $P = 8$ (b) Case II with $P = 12$

(c) Case III with $P = 15$ (d) Case III with $P = 25$

Figure 3: Threshold values for k-max search with $p_{\max} = 50$, $p_{\min} = 5$, $k = 20$, $\eta = 1.52$, and $\gamma = 2.63$.

We first compare the trade-off between consistency and robustness for k-max search with similar results in [24] for 1-max and one-way trading problems in Figure 1. In particular, we illustrate the Pareto boundaries of the consistency and robustness in one-way trading and the k-max search problem for four different values of k. For all problems, there exists a strong consistency-robustness trade-off. As the consistency improves from $\alpha_*^{(k)}$ (i.e., the optimal worst-case competitive ratio) to the best possible ratio 1, the robustness degrades from $\alpha_*^{(k)}$ to the worst possible ratio θ. In addition, as the total number of budget k increases, the Pareto boundary of k-max search improves towards that of the one-way trading. This is because a large value of k gives more flexibility in online decision-making to hedge the worst-case risk while using ML predictions.

We next sketch the key ideas in the design and analysis of $\texttt{OTA}_\phi$ and defer the full proof of Theorem 3.3 to Appendix B.1. Recall that in the worst-case optimized algorithm, the threshold divides the uncertainty range $[p_{\min}, p_{\max}]$ to $k+1$ intervals and the worst-case ratio $\alpha_i^{(k)}, i \in [k+1]$ is captured by Equation (5). The algorithm adopts a balancing rule by letting $\alpha_i^{(k)} = \alpha_*^{(k)}, \forall i \in [k+1]$ to achieve the optimal competitive ratio. In the learning-augmented algorithm, we aim to leverage the prediction P to achieve unbalanced ratios as shown in Equation (12).

Our high-level idea is to prioritize the trading decisions for the intervals neighbouring the prediction P by designing the thresholds (i.e., $\{c_i\}_{i=j^*+1}^{i^*}$) to attain good consistency, and just guarantee robustness in above-prediction (i.e., $\{r_i\}_{i=i^*+1}^{k}$) and below-prediction (i.e., $\{z_i\}_{i\in[j^*]}$) intervals. In particular, we design the thresholds $\{c_i\}_{i=j^*+1}^{i^*}$ by letting $\alpha_i^{(k)} \le \eta$ to ensure consistency. The key challenge is to simultaneously ensure the robustness of below-prediction and above-prediction intervals (that are illustrated by shaded pink areas in Figure 3). In the design, we have two key observations that are summarized in Propositions 3.4 and 3.5.

Proposition 3.4. *Given $j \in [k-1]$, if the thresholds after j are*

$$\phi_i = p_{\min} + \frac{p_{\max} - p_{\min}}{(1+\gamma/k)^{k-i+1}}, i = j+1, \ldots, k, \tag{16}$$

and $\alpha_{j+1}^{(k)}(\phi) \le \gamma$, then $\alpha_i^{(k)}(\phi) \le \gamma, i = j+2, \ldots, k+1$.

Proposition 3.5. *For a given $j \in [\xi]$, if the first j thresholds are*

$$\phi_i = p_{\min} + p_{\min}(\gamma - 1)\,(1+\gamma/k)^{i-1}, i = 1, \ldots, j, \tag{17}$$

then $\alpha_i^{(k)}(\phi) \le \gamma, i = 1, \ldots, j$.

First, conditioned on $\alpha_{j+1}^{(k)} \le \gamma$, if we design the remaining thresholds for $i = j+1, \ldots, k$ in Equation (16), the worst-case ratios of the above-prediction intervals are upper bounded by the robustness γ, and this leads to the design of the sequence $\{r_i\}_{i=i^*+1}^{k}$ and the determination of i^*. Second, based on Proposition 3.5, we can design the threshold $\{z_i\}_{i\in[j^*]}$ in Equation (17) to ensure the robustness of below-prediction intervals. Since the robustness of the below-prediction intervals can be trivially satisfied when prediction $P < \gamma p_{\min}$, the sequence $\{z_i\}_{i\in[j^*]}$ is only needed in Case III.

3.2 Learning-augmented algorithms for k-min

In the k-min search, we consider a prediction $P \in [p_{\min}, p_{\max}]$ of the minimum price of an instance. Our algorithm is still an OTA and we aim to leverage P to design the threshold $\psi(k, P) := \{\psi_i\}_{i\in[k]}$ such that $\texttt{OTA}_\psi$ can achieve the Pareto-optimal consistency-robustness trade-off. We start by deriving the lower bound of the trade-off.

Theorem 3.6. *For k-min search, any γ-robust deterministic learning-augmented algorithm must have a consistency lower bounded by*

$$\Lambda(\gamma) = \theta\gamma - \theta(\gamma-1)\,(1+1/(\gamma k))^{\zeta} - (\theta-1)\,(1-\zeta/k)\,, \tag{18}$$

where $\zeta = \left\lceil \ln\left(\frac{\theta-1}{\theta-\theta/\gamma}\right) / \ln\left(1+\frac{1}{\gamma k}\right) \right\rceil$.

Based on this lower bound, for a given $\lambda \in [0,1]$, we set the target consistency and robustness as

$$\gamma^{(k)}(\lambda) = \varphi_*^{(k)} + (1-\lambda)(\theta - \varphi_*^{(k)}), \quad \eta^{(k)}(\lambda) = \Lambda(\gamma(\lambda)), \tag{19}$$

where $\varphi_*^{(k)}$ is the optimal competitive ratio of k-min search.

We next derive the threshold ψ based on the prediction P. Different from the threshold ϕ in the k-max search, ψ is a non-increasing sequence of values bounded within $[p_{\min}, p_{\max}]$. ψ segments the price range into $k+1$ intervals $[\psi_0, \psi_1), \ldots, [\psi_{k-1}, \psi_k), [\psi_k, \psi_{k+1}]$, where $\psi_0 := p_{\max}$ and $\psi_{k+1} := p_{\min}$. We design ψ in the same form as ϕ in Equation (11), where we reuse the notations k-max search. Similar to k-max search, the threshold ψ is designed such that

$$\varphi_i^{(k)}(\psi) \le \begin{cases} \eta^{(k)}, & i = j^*+1, \ldots, i^*, \\ \gamma^{(k)}, & \text{otherwise}, \end{cases} \tag{20}$$

where $\varphi_i^{(k)}(\psi) := \frac{\sum_{j\in[i-1]} \psi_j + (k-i+1)p_{\max}}{k\psi_i}$ when the prediction of the minimum price $P \in [\psi_{j^*+1}, \psi_{i^*}]$. Thus, an accurate prediction will lead to consistency η, and regardless of the prediction, the robustness γ can be guaranteed. ψ is also designed in three cases, and the Pareto-optimality of $\texttt{OTA}_\psi$ is given in the follow-up theorem.

Let $\sigma^* \in [k]$ be the largest index that $\frac{1-(1-1/\eta)(1+1/(\eta k))^{\sigma^*}}{1-(1-1/\theta)/(1+1/(\gamma k))^{k-\sigma^*}} \le \frac{\gamma}{\eta}$. We define two boundary values $\tilde{p}_1 = p_{\max} - p_{\max}(1-1/\eta)(1+1/(\eta k))^{\sigma^*-1}$ and $\tilde{p}_2 = \min\{\tilde{p}_1, p_{\max}/\gamma\}$.

Case IV. When $P \in (\tilde{p}_1, p_{\max}]$, set $j^* = 0$, $i^* = \sigma^*$, and

$$c_i = p_{\max} - p_{\max}(1-1/\eta)(1+1/(\eta k))^{i-1}, \quad i \in [i^*], \tag{21a}$$

$$r_i = p_{\max} - \frac{p_{\max} - p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, \quad i = i^*+1, \dots, k. \tag{21b}$$

Case V. When $P \in (\tilde{p}_2, \tilde{p}_1]$, set $j^* = 0$ and

$$c_i = \begin{cases} P, & i = 1, \dots, m^*, \\ \frac{m^*P + (k-m^*)p_{\max}}{\eta k}, & i = m^*+1, \\ p_{\max} - (p_{\max} - c_{m^*+1})(1+1/(\eta k))^{i-m^*-1}, & i = m^*+2, \dots, i^*, \end{cases} \tag{22a}$$

$$r_i = p_{\max} - \frac{p_{\max} - p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, \quad i = i^*+1, \dots, k, \tag{22b}$$

where $m^* = \left\lceil k\frac{p_{\max}-\eta P}{p_{\max}-P} \right\rceil$ and i^* is the largest index that ensures $\varphi^{(k)}_{i^*+1}(\psi) \le \gamma$.

Case VI. When $P \in [p_{\min}, \tilde{p}_2]$, set

$$z_i = p_{\max} - p_{\max}(1-1/\gamma)(1+1/(\gamma k))^{i-1}, \quad i \in [j^*], \tag{23a}$$

$$c_i = \begin{cases} P, & i = j^*+1, \dots, m^*, \\ \frac{\sum_{i\in[j^*]} z_i + (m^*-j^*)P + (k-m^*)p_{\max}}{\eta k}, & i = m^*+1, \\ p_{\max} - (p_{\max} - c_{m^*+1})(1+1/(\eta k))^{i-1}, & i = m^*+2, \dots, i^*, \end{cases} \tag{23b}$$

$$r_i = p_{\max} - \frac{p_{\max} - p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, \quad i = i^*+1, \dots, k, \tag{23c}$$

in which we have $j^* = \left\lceil \ln\left(\frac{1-P/p_{\max}}{1-1/\gamma}\right) / \ln(1+1/(\gamma k)) \right\rceil$, and $m^* = j^* + \lceil \frac{k p_{\max}[1-(1-1/\gamma)(1+1/(\gamma k))^{j^*}] - kP\eta}{p_{\max}-P} \rceil$, and i^* is the largest index such that $\varphi^{(k)}_{i^*+1}(\psi) \le \gamma$.

Theorem 3.7. *Given $\lambda \in [0,1]$, $\mathtt{OTA}_\psi$ is $\eta^{(k)}(\lambda)$-consistent and $\gamma^{(k)}(\lambda)$-robust for the learning-augmented k-min search when ψ is given by Equations* (21)-(23)*, where $\eta := \eta^{(k)}(\lambda)$ and $\gamma := \gamma^{(k)}(\lambda)$ are given in Equation* (19)*. Further, $\mathtt{OTA}_\psi$ is Pareto-optimal.*

4 ONLINE SEARCH WITH INVENTORY

In this section, we consider an online search problem with inventory dynamics, extend the learning-augmented algorithm for online search to this more general problem, and prove the Pareto-optimality of this algorithm.

4.1 Problem statement

Consider an inventory management problem in a time-slotted system with horizon length T. In a volatile market with time-varying prices, an inventory with capacity B is used to store items when the price is low and use inventory to satisfy demand when the price is high. In each slot t, we first observe a demand $d_t \in \{0, 1, \dots\}$ that must be fulfilled immediately and a price p_t for purchasing the items in the market. The demand d_t can be satisfied by dual sources, either by the inventory that was purchased in previous slots, or the items purchased from the market. Then we decide $x_t \in \{0, 1, \dots\}$, the number of items to purchase from the market, in each slot t. If $x_t > d_t$, the additional items $x_t - d_t$ are stored in the inventory for future use. The goal is to minimize the total cost of purchasing items while fulfilling all real-time demands.

Let $\mathcal{I} := \{p_t, d_t\}_{t\in[T]}$ denote an instance of the online search with inventory dynamics (OSID). Given $\mathcal{I}$, the problem can be formulated as the following integer linear program.

$$\begin{aligned} (\texttt{min-OSID}) \quad \min \quad & \sum\nolimits_{t\in[T]} p_t x_t, && (24a)\\ \text{subject to} \quad & s_t = s_{t-1} + x_t - d_t, \quad \forall t \in [T], && (24b)\\ & x_t \ge d_t - s_{t-1}, \quad \forall t \in [T], && (24c)\\ & s_t \le B, \quad \forall t \in [T], && (24d)\\ \text{variables} \quad & x_t, s_t \in \{0, 1, \dots\}, \quad \forall t \in [T]. && (24e) \end{aligned}$$

where s_t is the inventory level at the end of slot t with initial state $s_0 = 0$. The objective (24a) of OSID is to minimize the total cost of purchasing the items over T slots. The constraint (24b) enforces the inventory dynamics, i.e., the inventory level s_t is equal to the previous inventory level s_{t-1} plus the net amount of purchased items $x_t - d_t$. The constraints (24c) and (24d) ensure that the inventory level is within capacity, i.e., $s_t \ge 0$ and $s_t \le B$ for $t \in [T]$.

OSID is a general framework that can capture various resource allocation problems with inventory dynamics. For instance, we can apply OSID to model and solve the storage-assisted energy trading problem in this paper. Particularly, consider a large energy consumer (e.g., data centers) that aims to purchase energy from the real-time energy market. It is equipped with an energy storage that can be used to store energy when the energy price is low and discharge to satisfy energy demand when the price is high. In each slot t, the consumer observes its energy demand d_t and the real-time electricity price p_t, and then decides how much energy to purchase, x_t, such that the demand can be satisfied by the combined amount from purchasing and storage discharging. In energy trading, demand d_t and decision x_t may be relaxed to continuous variables. Our algorithms and results can be extended for the continuous version of OSID (See Appendix D).

From the theoretical perspective, the offline OSID problem (24) is an extension of the k-min search formulated in (1). Let $\bar{k} = \min_{t\in[T]:d_t>0} d_t$ denote the minimum non-zero demand, and without loss of generality, consider $B \ge \bar{k}$ for an inventory management problem. When $d_t = 0$ for $t = 1, \dots, T-1$, and $d_T = \bar{k}$, OSID is to minimize the cost of buying a total of $\bar{k}$ items and storing them in the inventory over the first $T-1$ slots, and to use the inventory to satisfy the demand $\bar{k}$ in the last slot. This reduces to a $\bar{k}$-min search problem. When facing the general time-varying demand, OSID is much challenging than the classical online search problem.

Correspondingly, the k-max version of OSID can be used to model the problem of selling renewable energy (from solar panels or wind turbines) in electricity markets. The seller is equipped with an energy storage and can use it to temporally store the renewable energy when the market price is not attractive, waiting for a better selling opportunity. In each slot t, the seller obtains $a_t \in \{0, 1, \dots\}$ amount of renewable energy with no production costs, and observes the real-time electricity price p_t. It then determines to sell x_t

amount of energy in the market and earns a profit $x_t p_t$. If $x_t > a_t$, the additional energy $x_t - a_t$ is discharged from the energy storage. Otherwise, the unsold energy $a_t - x_t$ is stored in the inventory for trading in the future. The overall problem can be cast as follows:

$$\begin{aligned}
(\texttt{max-OSID}) \quad \max \quad & \sum\nolimits_{t\in[T]} p_t x_t, && (25a)\\
\text{subject to} \quad & s_t = s_{t-1} - x_t + a_t, \quad \forall t \in [T], && (25b)\\
& x_t \le a_t + s_{t-1}, \quad \forall t \in [T], && (25c)\\
& s_t \le B, \quad \forall t \in [T], && (25d)\\
\text{variables} \quad & x_t, s_t \in \{0, 1, \dots\}, \quad \forall t \in [T]. && (25e)
\end{aligned}$$

Note that if the renewable energy has a marginal production cost c_t ($c_t < p_{\min}, \forall t \in [T]$), we can use a refined electricity price $\hat{p}_t := p_t - c_t$ with refined price uncertainty range $[\hat{p}_{\min}, \hat{p}_{\max}]$, where $\hat{p}_{\min} := p_{\min} - \max_{t\in[T]} c_t$ and $\hat{p}_{\max} := p_{\max} - \min_{t\in[T]} c_t$ in the max-OSID. For simplicity of presentation, we focus on min-OSID in the rest of the paper; but the algorithms and results can be straightforwardly extended for max-OSID.

4.2 Learning-augmented algorithm for OSID

4.2.1 OSID with predictions. The key idea for designing online algorithms for OSID is to note the following fact: in each slot t, if the inventory level is sufficiently large to cover the demand d_t, we can use the inventory to satisfy the demand first, and then run a virtual online search algorithm to buy d_t items from the volatile market in the following slots and restore them back to the inventory. Thus, by running multiple virtual search problems in parallel, an online algorithm for OSID can achieve a similar performance to that of the online search if the inventory is non-empty.

Based on the above idea, we categorize time slots into busy periods with $s_t > 0$ and idle periods with $s_t = 0$, and make purchasing decisions differently in busy and idle periods. In particular, given an online algorithm for OSID, we divide the time horizon into a total of N intervals, where each interval n starts at a slot with non-empty inventory and ends before the starting of the next interval or T. Each interval n can be further divided into a busy period followed by an idle period. Let $\mathcal{T}_n^+$ and $\mathcal{T}_n^0$ denote sets of slots in the busy period and idle period of the interval n, respectively.

In the beginning of each slot t, we are given a prediction P_t. When $t \in \mathcal{T}_n^+$, P_t is the predicted minimum price in the busy period of interval n. When $t \in \mathcal{T}_n^0$, P_t is a prediction of the minimum price in the idle period of interval n and the busy period of $n+1$. Overall the prediction P_t is the minimum price that a virtual search problem created at slot t can observe in its lifecycle.

4.2.2 Learning-augmented algorithm. We propose an online algorithm for OSID in Algorithm 2. This algorithm maintains multiple virtual search problems in parallel in the busy period of each interval and uses the algorithm for k-min search with predictions in Section 3.2 as a subroutine to solve each virtual search problem. In the following, we call the k-min search with prediction P as a (k, P)-search problem. Let $\psi(k, P) := \{\psi_i(k, P)\}_{i\in[k]}$ denote the threshold values for (k, P)-search that can attain Pareto-optimal consistency-robustness trade-off in Equation (19). In the algorithm, we set a target robustness guarantee γ, and choose the optimal consistency $\eta^{(k)} := \eta^{(k)}(\gamma)$ for given γ and k.

Algorithm 2 divides the time horizon into N intervals. For each $t \in \mathcal{T}_n^+$ in the busy period, the algorithm observes the demand d_t and price p_t, and receives the prediction P_t. It then creates a virtual (d_t, P_t)-search problem if demand is strictly positive $d_t > 0$. Then the algorithm runs the learning-augmented online threshold-based algorithm (OTA) for all the i created online search problems in parallel and obtains $\{x_t^{(n,j)}\}_{j\in[i]}$, where $x_t^{(n,j)}$ is the online decision of the j-th search problem. The online decision of OSID is then set to the maximum of $\sum_{j\in[i]} x_t^{(n,j)}$ (i.e., the sum of trading decisions of all virtual search problem) and $d_t - s_{t-1}$ (i.e., the minimum trading amount to satisfy the demand). If the first term is larger, we have $s_t > 0$ and the busy period continues. Otherwise, the inventory becomes empty and the algorithm enters the idle period. Then the algorithm clears all virtual search problems created in the busy period, and initiates a (B, P_t)-search problem based on the physical capacity of the inventory, waiting for a low price that can start the busy period of the next interval.

Algorithm 2 Learning-augmented algorithm for OSID

input: threshold $\psi(k, P)$, inventory B, target robustness γ;
initiation: $n = 1$, $i = 1$; $m^{(1,1)} = 1$, $k^{(1,1)} = B$, $P^{(1,1)} = P_0$, $s_0 = 0$; $\psi^{(1,1)} := \psi(k^{(1,1)}, P^{(1,1)})$;
for $t = 1, \dots, T$ **do**
  observe demand d_t and price p_t; obtain prediction P_t;
  if $d_t > 0$ **then**
    $i = i + 1$, $k^{(n,i)} = d_t$, $P^{(n,i)} = P_t$, $\psi^{(n,i)} := \psi(k^{(n,i)}, P^{(n,i)}) = \{\psi_j^{(n,i)}\}_{j\in[k^{(n,i)}]}$, $m^{(n,i)} = 1$;
  end if
  for $j = 1, \dots, i$ **do**
    $x_t^{(n,j)} = 0$;
    while $p_t \le \psi_{m^{(n,j)}}^{(n,j)}$ and $m^{(n,j)} \le k^{(n,j)}$ **do**
      $m^{(n,j)} = m^{(n,j)} + 1$ and $x_t^{(n,j)} = x_t^{(n,j)} + 1$;
    end while
  end for
  $x_t = \max\left\{\sum_{j\in[i]} x_t^{(n,j)}, d_t - s_{t-1}\right\}$;
  $s_t = s_{t-1} + x_t - d_t$;
  if $s_t = 0$ and $s_{t-1} > 0$ **then**
    $n = n + 1$, $i = 1$, $m^{(n,1)} = 1$, $k^{(n,1)} = B$, $P^{(n,1)} = P_t$, $\psi^{(n,1)} := \psi(k^{(n,1)}, P^{(n,1)})$;
  end if
end for

4.2.3 Consistency-robustness analysis. We next show that Algorithm 2 provides the best consistency-robustness guarantees. We still assume all prices are bounded $p_t \in [p_{\min}, p_{\max}]$, and in OSID, we additionally make the following assumption for the demand.

Assumption 4.1. *The average demand in the idle period of one interval is smaller than capacity B, i.e., $\frac{1}{N}\sum_{n\in[N]}\sum_{t\in\mathcal{T}_n^0} d_t \le B$.*

This assumption basically means OSID is equipped with a sufficiently large inventory such that on average we can buy and store cheap items during busy periods and use the inventory to satisfy the demand in idle periods with high prices.

Theorem 4.2. *Under Assumption 4.1, given $\lambda \in [0, 1]$, Algorithm 2 is $\eta^{(\bar{k})}(\lambda)$-consistent and $\gamma^{(\bar{k})}(\lambda)$-robust for the learning-augmented* OSID *when $\psi(k, P)$ is given by Equations* (21)-(23)*, where $\eta^{(\bar{k})}(\lambda)$*

and $\gamma^{(\bar{k})}(\lambda)$ are defined in Equation (19)*, and $\bar{k} = \min_{t\in[T]:d_t>0} d_t$ is the minimum non-zero demand. Algorithm 2 is Pareto-optimal for* OSID *with minimum non-zero demand $\bar{k}$.*

Our main result in this section is to show that our proposed learning-augmented algorithms for OSID can achieve the best possible consistency-robustness trade-off, and this trade-off is the same as that of $\bar{k}$-min search with predictions, where $\bar{k}$ is the minimum non-zero demand of OSID and unknown to algorithm a prior. The key step of Algorithm 2 is to, in parallel, run multiple learning-augmented OTA for online search problems, and thus can be considered as an extension of the OTA for online search with predictions. The consistency-robustness guarantee is restricted by the performance of the online search problem with capacity $\bar{k}$ since $\bar{k}$-min search is a special case of OSID with minimum non-zero demand $\bar{k}$.

4.2.4 Proof of Theorem 4.2. We sketch the proof for Theorem 4.2, and defer all details to Appendix C. Let $\texttt{ALG}(\mathcal{I})$ and $\texttt{OPT}(\mathcal{I})$ denote the costs from the online algorithm and the offline optimal algorithm under an instance $\mathcal{I}$. We first show that Algorithm 2 generates a feasible solution for OSID.

Lemma 4.3. *The online solution of Algorithm 2 is feasible for* OSID.

Let i_n denote the number of virtual search problems created in the busy period of interval n, and let $m^{(n,i)}-1$ denote the total number of items purchased by the i-th virtual search problem in interval n. The next lemma shows that we can upper bound the total cost of the algorithm by threshold values $\{\psi_j^{(n,i)}\}_{n\in[N],i\in[i_n],j\in[m^{(n,i)}-1]}$ from the virtual search problems.

Lemma 4.4. *The cost of Algorithm 2 for* OSID *is upper bounded by*

$$\texttt{ALG}(\mathcal{I}) \le [s_T + R]\, p_{\max} + \sum_{n\in[N]}\sum_{i\in[i_n]}\left[\sum_{j\in[m^{(n,i)}-1]}\psi_j^{(n,i)} + (k^{(n,i)} - m^{(n,i)} + 1)p_{\max}\right],$$

where $R = \sum_{n\in[N]}\sum_{t\in\mathcal{T}_n^0} d_t - NB$.

Next we show that the total cost of the offline optimum is lower bounded by threshold values $\{\psi_{m^{(n,i)}}^{(n,i)}\}_{n\in[N],i\in[i_n]}$ from the virtual search problems.

Lemma 4.5. *The cost of offline optimum for* OSID *is lower bounded*

$$\texttt{OPT}(\mathcal{I}) \ge \sum_{n\in[N]}\sum_{i\in[i_n]}\psi_{m^{(n,i)}}^{(n,i)}\cdot k^{(n,i)} + \frac{R\cdot p_{\max}}{\gamma}. \tag{26}$$

Based on the property of threshold values given in (20), regardless of the accuracy of the predictions, we have $\sum_{j\in[m^{(n,i)}-1]}\psi_j^{(n,i)} + (k^{(n,i)} - m^{(n,i)} + 1)p_{\max} \le \gamma\psi_{m^{(n,i)}}^{(n,i)} k^{(n,i)}, \forall n\in[N], i\in[i_n]$. Combining Lemma 4.4 and Lemma 4.5 gives

$$\frac{\texttt{ALG}(\mathcal{I}) - s_T p_{\max}}{\texttt{OPT}(\mathcal{I})} \le \frac{\gamma\sum_{n\in[N]}\sum_{i\in[i_n]}\psi_{m^{(n,i)}}^{(n,i)}k^{(n,i)} + Rp_{\max}}{\sum_{n\in[N]}\sum_{i\in[i_n]}\psi_{m^{(n,i)}}^{(n,i)}k^{(n,i)} + \frac{R\cdot p_{\max}}{\gamma}} = \gamma.$$

Thus, the robustness of Algorithm 2 is given by $\texttt{ALG}(\mathcal{I}) \le \gamma\texttt{OPT}(\mathcal{I}) + s_T p_{\max}$, where $s_T p_{\max}$ is an additive loss due to the remaining inventory at the end of time horizon. Note that this additive loss is inevitable compared to the offline optimum since the offline algorithm knows the length of time horizon, and thus can always use up all inventory at the end of horizon to minimize the cost. Further, as the time horizon T increases, both $\texttt{ALG}(\mathcal{I})$ and $\texttt{OPT}(\mathcal{I})$ increase and the performance loss is dominated by the multiplier term γ while the additive term $s_T p_{\max}$ is negligible.

When all predictions are accurate, the threshold values given by Equations (21)-(23) ensure that $\sum_{j\in[m^{(n,i)}-1]}\psi_j^{(n,i)} + (k^{(n,i)} - m^{(n,i)} + 1)p_{\max} \le \eta^{(k^{(n,i)})}\psi_{m^{(n,i)}}^{(n,i)}k^{(n,i)}, \forall n\in[N], i\in[i_n]$. Based on Lemma 4.4 and Lemma 4.5, we have

$$\frac{\texttt{ALG}(\mathcal{I}) - s_T p_{\max}}{\texttt{OPT}(\mathcal{I})} \le \frac{\eta^{(\bar{k})}\sum_{n\in[N]}\sum_{i\in[i_n]}\psi_{m^{(n,i)}}^{(n,i)}k^{(n,i)} + Rp_{\max}}{\sum_{n\in[N]}\sum_{i\in[i_n]}\psi_{m^{(n,i)}}^{(n,i)}k^{(n,i)} + \frac{R\cdot p_{\max}}{\gamma}} \le \eta^{(\bar{k})},$$

where the last inequality holds since $R = \sum_{n\in[N]}\sum_{t\in\mathcal{T}_n^0} d_t - NB \le 0$ under the Assumption 4.1. Thus, Algorithm 2 can guarantee $\texttt{ALG}(\mathcal{I}) \le \eta^{(\bar{k})}\texttt{OPT}(\mathcal{I}) + s_T p_{\max}$.

Therefore, Algorithm 2 is $\gamma^{(\bar{k})}$-robust and $\eta^{(\bar{k})}$-consistent.

Also, note that $\bar{k}$-min search with predictions is a special case of OSID with minimum non-zero demand $\bar{k}$. $\gamma^{(\bar{k})}$ and $\eta^{(\bar{k})}$ are also the lower bound of the consistency-robustness trade-off. Thus, the algorithm is Pareto-optimal.

4.3 Inventory-cost-aware algorithms

In numerous real-world applications of OSID, the utilization of inventory introduces additional expenses, such as the battery cost and energy conversion loss in the context of the storage-assisted energy trading problem. To address these associated costs, we extend Algorithm 2 to an inventory-cost-aware algorithm. Specifically, our extension accounts for two types of inventory-related costs in the energy trading problem: (i) *usage cost*, entailing a cost of δ for each unit of energy charged into or discharged from inventory; and (ii) *conversion loss*, wherein only ρ_c and ρ_d percentages of energy can be efficiently charged into and discharged from the inventory, respectively. The inventory-cost-aware problem can be cast as:

$$\min_{x_t\ge 0, s_t\ge 0} \quad \sum\nolimits_{t\in[T]} p_t x_t + |d_t - x_t|\delta \tag{27a}$$

$$\text{subject to} \quad s_t = s_{t-1} + \rho_c x_t^c - \frac{1}{\rho_d}x_t^d, \forall t\in[T] \tag{27b}$$

$$x_t^c = (x_t - d_t)^+, \forall t\in[T], \tag{27c}$$

$$x_t^d = (d_t - x_t)^+, \forall t\in[T], \tag{27d}$$

$$d_t - x_t \le \rho_d s_{t-1}, \forall t\in[T], \tag{27e}$$

$$s_t \le B, \forall t\in[T], \tag{27f}$$

where the objective (27a) additionally considers a cost $|d_t - x_t|\delta$ from inventory usage and the inventory dynamics (27b) takes into account the energy conversion loss. We further relax the integral constraints for this energy trading problem.

We make two modifications in Algorithm 2 to handle inventory costs. To deal with the conversion loss, we increase the capacity of the *first* virtual storage to $k^{(1,1)} = B/(\rho_c\rho_d)$ in Line 2. The intuition for adjusting only the first storage is that its extended capacity of B can be considered largely responsible for moving units in and out of the physical storage. Modifying just this storage will adjust the

charging amount and leaving the virtual storages with capacity d_t untouched will maintain trading to meet demand.

To deal with the inventory usage cost, we modify the threshold prices depending on the current demand. The primary objective is to encourage the algorithm to refrain from utilizing the inventory unless it has good potential to reduce the purchasing cost substantially. To achieve this, we set thresholds to avoid purchasing a quantity too close to d_t. Specifically, we modify the thresholds $\psi^{(n,j)}$ in Line 10 to use $\psi'^{(n,j)}$ defined as follows:

$$\psi_i'^{(n,j)} = \begin{cases} \psi_\tau^{(n,j)} - 2\delta, & \psi_\tau^{(n,j)} - 2\delta \le \psi_i^{(n,j)} < \psi_\tau^{(n,j)} \\ \psi_\tau^{(n,j)}, & \psi_\tau^{(n,j)} \le \psi_i^{(n,j)} \le \psi_\tau^{(n,j)} + 2\delta \\ \psi_i^{(n,j)}, & \text{otherwise;} \end{cases}$$

where $\tau = m^{(n,j)} + d_t \cdot \left\lfloor \frac{k^{(n,j)}}{\sum_v k^{(n,v)}} \right\rfloor$. First, we set a target utilization τ such that the aggregate target utilization over j inventories is d_t more than the current level. Then, prices within a range 2δ of the corresponding target price $\psi_\tau^{(n,j)}$ are modified such that the algorithm only charges to the physical storage if the price is at least 2δ above/below the threshold price for meeting demand d_t. The range 2δ is chosen because it inevitably costs 2δ to charge and discharge one unit. Adding usage costs and conversion loss fundamentally alters the problem formulation of OSID, making it much more challenging to analyze the theoretical performance of the inventory-cost-aware algorithm. Therefore, we leave the theoretical analysis of the algorithm for our future work but verify the performance of the modified algorithms numerically in Section 5.

5 EXPERIMENTAL RESULTS

5.1 Experimental setup

We apply the learning-augmented algorithms for the continuous OSID (see Appendix D for more details). For supplementary experiments on the basic k-search problem, see Appendix E. We set the length of each slot to 5 minutes and set the time horizon of each instance to 24 hours, i.e., $T = 288$. The energy storage capacity is set to $18 \times \max_{t\in[T]} d_t$ such that a full charge of the storage can power the data center for 90 minutes of maximum energy demand. We report the empirical ratios over 31 days.

Energy price. We use the day-ahead electricity prices of local electricity markets over several different independent system operators in the United States, i.e., CAISO, NYISO, ERCOT, and ISO-NE [9]. To have a consistent 5-minute settlement interval for our markets, we up-sample some market price readings.

Data center demand. We use a repository of demand traces from Akamai's server clusters collected during a 31-day period from multiple locations. We run our algorithms on data centers located in relevant electricity markets: New York for NYISO, Dallas for ERCOT, Los Angeles for CAISO, and Boston for ISO-NE. The energy demand is partially supplied by renewable generation, which has uncertainty that causes the net demand (the data center demand minus renewable supply) to be more volatile.

Algorithm predictions. The learning-augmented algorithm for OSID utilizes a prediction P_t that predicts the minimum price in the lifecyle of the virtual online search created at slot t. In our experiments, we consider utilizing the minimum price of half-hour, 1-hour, and 2-hour windows. To adjust the prediction quality, we add prediction errors drawn from a normal distribution with mean 0 and variance σ scaled by a factor of θ.

5.2 Comparison algorithms

We compare the following seven algorithms. (NoSTR): An algorithm that satisfies the demand by purchasing energy from the market without using storage. The cost ratio of NoSTR demonstrates the maximum benefit of storage. (OS-on): The worst-case optimized online algorithm [27] that solves OSID without predictions. (OS-b): The baseline algorithm introduced in Section 2.3 adapted for OSID. (OS-λ): Our proposed learning-augmented algorithm for OSID. The value of λ is selected according to an online learning algorithm (i.e., adversarial Lipschitz algorithm [20]) to adaptively select λ. (OS-λ^*): Our proposed algorithm for OSID with the best choice of λ. (OS-λ-IA): Our extended learning-augmented algorithm that is aware of inventory usage costs and conversion loss. (OS-on-IA): The inventory-cost-aware algorithm without predictions.

5.3 Experimental results

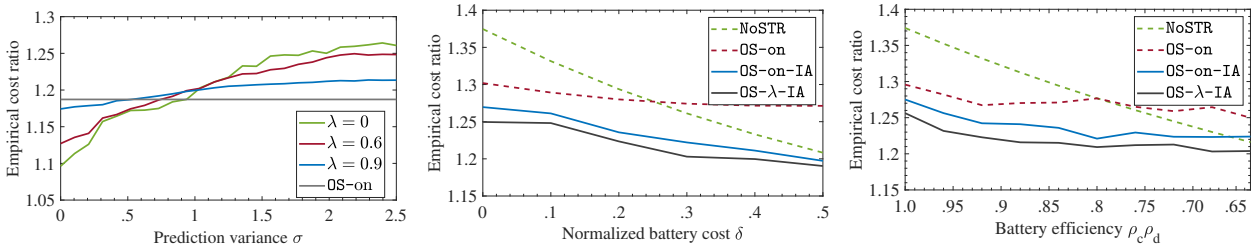
We report the average empirical ratios of five algorithms in Table 1 across data center locations, seasons, and prediction window sizes. The magnitude of the price fluctuation ratio θ clearly indicates the problem difficulty, especially for OS-on. For example, in the seasonal breakdown of ISO-NE, the cost ratio of OS-on increases by 13% in winter as compared to spring. The same effect can be observed for learning-augmented algorithms to a lesser degree: the same comparison for OS-λ increases by 3% from spring to winter.

We observe that OS-λ outperforms other baseline algorithms averaged over the course of a year. The degree of performance gain depends on the prediction quality, with smaller prediction windows yielding a greater gain. In the half-hour prediction window, OS-λ outperforms OS-on averaged over a year by between 5% to 8%, depending on the dataset. In comparison, the 2-hour window yields a year-average performance gain between 1.5% to 3%. The performance of OS-λ relative to OS-b highlights the importance of the optimality of our algorithm over all other prediction-based algorithms, with a performance improvement of up to 8.4% in the year-average CAISO half-hour trial. Both OS-λ and OS-λ^* have relatively smaller improvements on OS-b in the 2-hour window. Despite this, all prediction-based algorithms perform better than OS-on, indicating that even with poor predictions, good usage of the trust parameter still improves on worst-case optimized algorithms.

To demonstrate the inherent tradeoff between the robustness and consistency of our learning-augmented algorithms, we show the performance of the algorithms with varying prediction errors in Figure 4(a). The prediction error is generated by Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma)$, where $\sigma \in [0, 2.5]$ and is normalized by θ. Recall that algorithms with different λs achieve different consistency-robustness tradeoffs, a smaller λ rendering a better consistency and worse robustness. The algorithm with $\lambda = 0$ attains the best consistency, i.e., it achieves the lowest cost ratio when the prediction error is small; however, its robustness is the worst, i.e., it has the highest cost ratio as the prediction error becomes large. As we increase the value of λ, the algorithm can experience lower cost ratios in scenarios with large prediction errors, at the cost of higher cost ratios when the prediction error is small. This corresponds with

Table 1: The reported numbers are empirical cost ratios (the lower, the better, and one is the best) of algorithms in varying markets and seasons; θ is the price fluctuation ratio; $\alpha_*^{(\infty)}$ is the theoretical competitive ratio.

	Season	θ	$\alpha_*^{(\infty)}$	NoSTR	OS-on	Half-hour window			1-hour window			2-hour window		
						OS-b	OS-λ	OS-λ^*	OS-b	OS-λ	OS-λ^*	OS-b	OS-λ	OS-λ^*
CAISO	**Spring**	33.25	4.40	1.34	1.29	1.31	1.19	1.11	1.26	1.22	1.17	1.24	1.24	1.22
	Summer	44.59	5.05	1.47	1.37	1.36	1.21	1.12	1.31	1.23	1.15	1.40	1.41	1.38
	Fall	4.62	1.83	1.36	1.23	1.26	1.18	1.10	1.22	1.15	1.10	1.22	1.17	1.16
	Winter	18.78	3.39	1.43	1.26	1.27	1.18	1.09	1.26	1.17	1.14	1.26	1.22	1.20
	Year	**25.31**	**3.67**	**1.40**	**1.28**	**1.30**	**1.19**	**1.11**	**1.26**	**1.19**	**1.14**	**1.28**	**1.25**	**1.24**
NYISO	**Spring**	3.74	1.68	1.36	1.20	1.22	1.15	1.11	1.19	1.16	1.13	1.18	1.20	1.19
	Summer	5.93	2.03	1.36	1.28	1.28	1.22	1.12	1.23	1.17	1.11	1.23	1.22	1.20
	Fall	4.66	1.84	1.38	1.28	1.28	1.21	1.13	1.23	1.18	1.13	1.21	1.20	1.19
	Winter	2.83	1.50	1.21	1.17	1.19	1.13	1.07	1.21	1.14	1.11	1.20	1.17	1.14
	Year	**4.29**	**1.76**	**1.31**	**1.22**	**1.23**	**1.17**	**1.10**	**1.21**	**1.16**	**1.12**	**1.20**	**1.20**	**1.17**
ERCOT	**Spring**	7.66	2.27	1.38	1.28	1.29	1.20	1.10	1.28	1.22	1.12	1.30	1.31	1.30
	Summer	4.98	1.89	1.52	1.26	1.26	1.19	1.11	1.25	1.20	1.11	1.22	1.21	1.20
	Fall	50.37	5.34	1.57	1.41	1.40	1.21	1.11	1.39	1.28	1.16	1.42	1.40	1.39
	Winter	9.13	2.45	1.33	1.26	1.27	1.22	1.12	1.24	1.18	1.13	1.21	1.20	1.19
	Year	**18.03**	**2.99**	**1.45**	**1.30**	**1.30**	**1.20**	**1.11**	**1.29**	**1.22**	**1.13**	**1.28**	**1.28**	**1.27**
ISO-NE	**Spring**	10.57	2.62	1.32	1.26	1.27	1.25	1.10	1.25	1.24	1.13	1.24	1.23	1.19
	Summer	9.37	2.48	1.47	1.34	1.33	1.22	1.11	1.30	1.24	1.17	1.34	1.32	1.28
	Fall	6.70	2.15	1.36	1.27	1.29	1.18	1.10	1.27	1.22	1.16	1.28	1.27	1.22
	Winter	46.44	5.14	1.69	1.43	1.39	1.29	1.13	1.36	1.34	1.25	1.32	1.32	1.31
	Year	**18.27**	**3.10**	**1.46**	**1.33**	**1.32**	**1.23**	**1.11**	**1.29**	**1.26**	**1.18**	**1.30**	**1.29**	**1.25**



(a) Effect of prediction quality on settings of λ (b) Impact of inventory usage cost (c) Impact of conversion loss

Figure 4: Cost ratio vs. different problem parameters. Dallas Jan 2023 with half-hour prediction window.

the theoretical robustness-consistency tradeoffs. Moreover, the performances of all learning-augmented algorithms smoothly improve with increasing prediction quality. This is desirable for algorithms with untrusted predictions, and it underscores that, despite the algorithms' obliviousness to the prediction quality, their performances consistently improve as prediction accuracy increases.

In Figure 4(b) and Figure 4(c), we showcase the impacts of inventory usage cost and conversion loss on proposed algorithms. We compare the average cost ratios of our inventory-cost-aware algorithms against NoSTR and OS-on, which are ignorant of the additional inventory dynamics. The usage cost δ in Figure 4(b) is normalized by $p_{\min}$ and the charging and discharging losses are represented by the overall battery efficiency $\rho_c\rho_d$. Storage efficiency in datacenters depends on the energy storage device used, ranging from as high as 95% for ultra-capacitors or as low as 68% for compressed air energy storage [25]. We observe that the increase in the usage cost or the conversion loss tends to lower the cost ratios of algorithms since the cost of the offline algorithm also rises. As usage costs or energy loss grow, offline algorithm charges the battery less frequently, and NoSTR becomes increasingly viable. OS-λ-IA offers consistent performance improvements over baseline algorithms with different battery costs until $\delta = 0.5$, where it matches the cost ratio of NoSTR. As δ increases, the performance gain from using OS-λ-IA over OS-on grows from 5% to almost 10%. For battery efficiency below 65%, utilizing predictions is key to maintaining improvement over baselines, as OS-on-IA matches NoSTR.

6 CONCLUSION

We have designed online algorithms augmented by machine-learned predictions for the online search problem and its extension to online search with inventory dynamics. Theoretically, the performance of our proposed algorithms is consistent with that of offline algorithms in hindsight when the prediction is accurate and robust on prediction errors. Further, the trade-off between consistency and robustness has been proven Pareto-optimal. Practically, we have applied the learning-augmented algorithms to the storage-assisted energy trading problem in energy markets. Through extensive experiments using real traces, our proposed learning-augmented algorithm has been shown to achieve the best of both worlds in the sense that it improves the average empirical performance compared to existing benchmark algorithms, while also improving the worst-case performance even when the predictions are inaccurate.

ACKNOWLEDGMENTS

The work of Russell Lee and Mohammad Hajiesmaili was supported by the U.S. National Science Foundation (NSF) under grant numbers CAREER-2045641, NGSDI-2105494, CPS-2136199, CNS-2106299, and CNS-2102963. The work of John C.S. Lui was supported in part by the RGC GRF 14202923.

REFERENCES

[1] Spyros Angelopoulos, Shahin Kamali, and Kimia Shadkami. 2022. Online Bin Packing with Predictions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, Lud De Raedt (Ed.). International Joint Conferences on Artificial Intelligence Organization, 4574–4580.

[2] Antonios Antoniadis, Christian Coester, Marek Elias, Adam Polak, and Bertrand Simon. 2020. Online metric algorithms with untrusted predictions. In *International Conference on Machine Learning*. PMLR, 345–355.

[3] Antonios Antoniadis, Themis Gouleakis, Pieter Kleer, and Pavel Kolev. 2020. Secretary and online matching problems with machine learned advice. *Advances in Neural Information Processing Systems* 33 (2020), 7933–7944.

[4] Santiago Balseiro, Christian Kroer, and Rachitesh Kumar. 2023. Single-Leg Revenue Management with Advice. In *Proceedings of the 24th ACM Conference on Economics and Computation* (London, United Kingdom) *(EC '23)*. Association for Computing Machinery, New York, NY, USA, 207.

[5] Allan Borodin and Ran El-Yaniv. 2005. *Online computation and competitive analysis*. cambridge university press.

[6] Roozbeh Bostandoost, Bo Sun, Carlee Joe-Wong, and Mohammad Hajiesmaili. 2023. Near-optimal Online Algorithms for Joint Pricing and Scheduling in EV Charging Networks. In *Proceedings of the 14th ACM International Conference on Future Energy Systems*. 72–83.

[7] Paul Dütting, Silvio Lattanzi, Renato Paes Leme, and Sergei Vassilvitskii. 2021. Secretaries with advice. In *Proceedings of the 22nd ACM Conference on Economics and Computation*. 409–429.

[8] R. El-Yaniv, A. Fiat, R. M. Karp, and G. Turpin. 2001. Optimal Search and One-Way Trading Online Algorithms. *Algorithmica* 30, 1 (May 2001), 101–139.

[9] Energy Online 2023. LCG Consulting Energy Online. available at httsp://www.energyonline.com/Data.

[10] Dimitris Fotakis, Evangelia Gergatsouli, Themis Gouleakis, and Nikolas Patris. 2021. Learning augmented online facility location. *arXiv preprint arXiv:2107.08277* (2021).

[11] Gemini Exchange 2023. Gemini Exchange Dataset. available at https://www.cryptodatadownload.com/data/gemini.

[12] Sreenivas Gollapudi and Debmalya Panigrahi. 2019. Online algorithms for rent-or-buy with expert advice. In *International Conference on Machine Learning*. PMLR, 2319–2327.

[13] Mark Heinsbroek. 2022. *Online One-Way Trading with Machine Learned Advice*. Master's thesis. Utrecht University.

[14] Sungjin Im, Ravi Kumar, Mahshid Montazer Qaem, and Manish Purohit. 2021. Online knapsack with frequency predictions. *Advances in Neural Information Processing Systems* 34 (2021), 2733–2743.

[15] Shaofeng H-C Jiang, Erzhi Liu, You Lyu, Zhihao Gavin Tang, and Yubo Zhang. 2021. Online Facility Location with Predictions. In *International Conference on Learning Representations*.

[16] Billy Jin and Will Ma. 2022. Online Bipartite Matching with Advice: Tight Robustness-Consistency Tradeoffs for the Two-Stage Model. *arXiv preprint arXiv:2206.11397* (2022).

[17] Russell Lee, Jessica Maghakian, Mohammad Hajiesmaili, Jian Li, Ramesh Sitaraman, and Zhenhua Liu. 2021. Online peak-aware energy scheduling with untrusted advice. *ACM SIGENERGY Energy Informatics Review* 1, 1 (2021), 59–77.

[18] Julian Lorenz, Konstantinos Panagiotou, and Angelika Steger. 2009. Optimal algorithms for k-search with application in option pricing. *Algorithmica* 55, 2 (2009), 311–328.

[19] Thodoris Lykouris and Sergei Vassilvtiskii. 2018. Competitive Caching with Machine Learned Advice. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 3296–3305.

[20] Odalric-Ambrym Maillard and Rémi Munos. 2010. Online learning in adversarial lipschitz environments. In *Joint european conference on machine learning and knowledge discovery in databases*. Springer, 305–320.

[21] Ali Menati, Sid Chi-Kin Chau, and Minghua Chen. 2022. Competitive prediction-aware online algorithms for energy generation scheduling in microgrids. In *Proceedings of the Thirteenth ACM International Conference on Future Energy Systems*. 383–394.

[22] Yanfang Mo, Qiulin Lin, Minghua Chen, and Si-Zhao Joe Qin. 2021. Optimal online algorithms for peak-demand reduction maximization with energy storage. In *Proceedings of the twelfth ACM international conference on future energy systems*. 73–83.

[23] Manish Purohit, Zoya Svitkina, and Ravi Kumar. 2018. Improving Online Algorithms via ML Predictions. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc.

[24] Bo Sun, Russell Lee, Mohammad Hajiesmaili, Adam Wierman, and Danny Tsang. 2021. Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. In *Advances in Neural Information Processing Systems*, Vol. 34.

[25] Di Wang, Chuangang Ren, Anand Sivasubramaniam, Bhuvan Urgaonkar, and Hosam Fathy. 2012. Energy Storage in Datacenters: What, Where, and How Much? 40, 1 (jun 2012), 187–198. https://doi.org/10.1145/2318857.2254780

[26] Alexander Wei and Fred Zhang. 2020. Optimal robustness-consistency tradeoffs for learning-augmented online algorithms. *Advances in Neural Information Processing Systems* 33 (2020), 8042–8053.

[27] Lin Yang, Mohammad H Hajiesmaili, Ramesh Sitaraman, Adam Wierman, Enrique Mallada, and Wing S Wong. 2020. Online linear optimization with inventory management constraints. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 4, 1 (2020), 1–29.

[28] Ali Zeynali, Bo Sun, Mohammad Hajiesmaili, and Adam Wierman. 2021. Data-driven competitive algorithms for online knapsack and set cover. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 10833–10841.

[29] Wenming Zhang, Yinfeng Xu, Feifeng Zheng, and Ming Liu. 2011. Online algorithms for the general k-search problem. *Information processing letters* 111, 14 (2011), 678–682.

A PROOFS ON BASELINE ALGORITHMS

We first prove Lemma 2.3 that provides the consistency and robustness of the baseline algorithm for k-max search. For completeness of the presentation, we then give the corresponding result for k-min search that can be derived similarly.

A.1 Proof of Lemma 2.3

Given a hyper-parameter $\lambda \in [0,1]$, the baseline algorithm reserves $k_r = \lceil \lambda k \rceil$ capacity and $k_c = k - k_r$ capacity for running the robust algorithm and the prediction-based algorithm, respectively. In particular, the robust algorithm is the worst-case optimized algorithm (i.e., Algorithm 1 with threshold values given in Equation (3)) and the prediction-based algorithm waits to trade all k_c items until the first price that is no smaller than the predicted maximum price P.

Let $\bar{x}_t$ and $\hat{x}_t$ denote the trading decisions from the robust algorithm and the prediction-based algorithm, respectively. Then the baseline algorithm trades $x_t = \bar{x}_t + \hat{x}_t$ in step t. Given an instance $\mathcal{I} = \{p_t\}_{t\in[T]}$, the total revenue of the baseline algorithm is $\texttt{ALG}(\mathcal{I}) = \sum_{t\in[T]} p_t x_t = \sum_{t\in[T]} p_t \bar{x}_t + \sum_{t\in[T]} p_t \hat{x}_t$ and the offline optimum is $\texttt{OPT}(\mathcal{I}) = kV$, where $V = \max_{t\in[T]} p_t$ is the actual maximum price.

We first show the robustness of the baseline algorithm. Since $\bar{x}_t$ is given by the optimal k_r-max search algorithm, we have $\sum_{t\in[T]} p_t \bar{x}_t \geq k_r V/\alpha_*^{(k_r)}$, in which $\alpha_*^{(k_r)}$ is the optimal competitive ratio of k_r-max search. In addition, the revenue of the prediction-based algorithm is lower bounded by $\sum_{t\in[T]} p_t \hat{x}_t \geq L k_c \geq k_c V/\theta$. Therefore, we have

$$\texttt{ALG}(\mathcal{I}) \geq k_r V/\alpha_*^{(k_r)} + k_c V/\theta = \left[k_r/(k\alpha_*^{(k_r)}) + (k-k_r)/(k\theta)\right] \texttt{OPT}(\mathcal{I}). \tag{28}$$

Thus, the robustness of the baseline algorithm is $k/[k_r/\alpha_*^{(k_r)} + (k-k_r)/\theta]$.

Next, we prove the consistency. When the prediction P is accurate, i.e., $P = V$, the revenue of the prediction-based algorithm is $\sum_{t\in[T]} p_t \hat{x}_t = V k_c$. We then have

$$\texttt{ALG}(\mathcal{I}) \geq k_r V/\alpha_*^{(k_r)} + k_c V = \left[k_r/(k\alpha_*^{(k_r)}) + (k-k_r)/k\right] \texttt{OPT}(\mathcal{I}). \tag{29}$$

Thus, the consistency is $k/[k_r/\alpha_*^{(k_r)} + k - k_r]$. This completes the proof.

A.2 Results of baseline algorithms for k-min search.

The baseline algorithm can also solve the k-min search problem. We just need to modify the robust algorithm to the optimal k_r-min search algorithm and the prediction-based algorithm to wait to trade all k_c items until the first price that is no larger than the predicted minimum price P. Then we can have the following result.

Lemma A.1. *Given $\lambda \in [0,1]$, the baseline algorithm is $\frac{k_r\varphi_*^{(k_r)}+k-k_r}{k}$-consistent and $\frac{k_r\varphi_*^{(k_r)}+(k-k_r)\theta}{k}$-robust for the learning-augmented k-min search problem, where $k_r := \lceil \lambda k \rceil$.*

Proof of Lemma A.1. Given an instance $\mathcal{I} = \{p_t\}_{t\in[T]}$, the total cost of the baseline algorithm is $\texttt{ALG}(\mathcal{I}) = \sum_{t\in[T]} p_t x_t = \sum_{t\in[T]} p_t \bar{x}_t + \sum_{t\in[T]} p_t \hat{x}_t$ and the offline optimum is $\texttt{OPT}(\mathcal{I}) = kV$, where $V = \min_{t\in[T]} p_t$ is the actual minimum price.

We first prove the robustness. Since $\bar{x}_t$ is from the optimal k_r-min search algorithm, we have $\sum_{t\in[T]} p_t \bar{x}_t \leq k_r V \varphi_*^{(k_r)}$, in which $\varphi_*^{(k_r)}$ is the optimal competitive ratio of k_r-min search. In addition, the cost of the prediction-based algorithm is upper bounded by $\sum_{t\in[T]} p_t \hat{x}_t \leq p_{\max} k_c \leq k_c V\theta$. Therefore, we have

$$\texttt{ALG}(\mathcal{I}) \leq k_r V \varphi_*^{(k_r)} + k_c V\theta = \texttt{OPT}(\mathcal{I})[k_r\varphi_*^{(k_r)} + (k-k_r]\theta)/k. \tag{30}$$

Thus, the robustness of the baseline algorithm is $[k_r\varphi_*^{(k_r)} + (k-k_r)\theta]/k$.

Next, we prove the consistency. When the prediction P is accurate, i.e., $P = V$, the cost of the prediction-based algorithm is $\sum_{t\in[T]} p_t \hat{x}_t = V k_c$. We then have

$$\texttt{ALG}(\mathcal{I}) \leq k_r V \varphi_*^{(k_r)} + k_c V = \texttt{OPT}(\mathcal{I})[k_r\varphi_*^{(k_r)}/k + (k-k_r)/k]. \tag{31}$$

Thus, the consistency is $(k_r\varphi_*^{(k_r)} + k - k_r)/k$.

B PROOFS ON PARETO-OPTIMAL LEARNING-AUGMENTED ALGORITHMS

B.1 Proof of Theorem 3.3

The threshold values $\phi = \{\phi_i\}_{i\in[k]}$ for k-max is a non-decreasing sequence $\phi_1 \leq \phi_2 \leq \cdots \leq \phi_k$. For notations convenience, we define $\phi_0 = p_{\min}$ and $\phi_{k+1} = p_{\max}$. The threshold ϕ divides the uncertainty range $[p_{\min}, p_{\max}]$ into $k+1$ regions. If $\phi_i = \phi_{i+1} = \cdots = \phi_j < \phi_{j+1}$, we use $j+1$ to index the interval $[\phi_i, \phi_{j+1})$. If $\texttt{OTA}$ totally trades i items (excluding the compulsory trading at the last step), then the worst-case

price sequence is

$$\phi_1, \phi_2, \ldots, \phi_i, \underbrace{\phi_{i+1} - \epsilon, \ldots, \phi_{i+1} - \epsilon}_{k}, \underbrace{p_{\min}, \ldots, p_{\min}}_{k},$$

where $\epsilon > 0$ and $\epsilon \to 0$. The worst-case ratio under this price sequence can be characterized by

$$\alpha_{i+1}^{(k)}(\phi) = \frac{k\phi_{i+1}}{\sum_{j\in[i]}\phi_j + (k-i)p_{\min}}. \tag{32}$$

Before proceeding to the proof of Theorem 3.3, we first prove Proposition 3.4 and Proposition 3.5.

Proof of Proposition 3.4. Given $j \in [k-1]$, if $\alpha_{j+1}^{(k)} \le \gamma$, we have

$$\sum\nolimits_{s\in[j]}\phi_s + (k-j)p_{\min} \ge \frac{k\phi_{j+1}}{\gamma}. \tag{33}$$

We then have

$$\alpha_{j+2}^{(k)} = \frac{k\phi_{j+2}}{\sum_{s\in[j+1]}\phi_s + (k-j-1)p_{\min}} = \frac{k\phi_{j+2}}{\sum_{s\in[j]}\phi_s + (k-j)p_{\min} + \phi_{j+1} - p_{\min}} \tag{34a}$$

$$\le \frac{k\phi_{j+2}}{k\phi_{j+1}/\gamma + \phi_{j+1} - p_{\min}} \tag{34b}$$

$$= \frac{\gamma\phi_{j+2}}{p_{\min} + (1+\gamma/k)[\phi_{j+1} - p_{\min}]} = \gamma, \tag{34c}$$

where (34b) holds due to Inequality (33) and the last equality (34c) is based on the design of the threshold values in Equation (16). By following the same step, the worst-case ratios in the following intervals can be shown that $\alpha_i^{(k)} \le \gamma, i = j+3, \ldots, k+1$.

Proof of Proposition 3.5. When the first j threshold values are designed based on Equation (17), the worst-case ratio of the i-th interval of OTA can be shown as

$$\alpha_i^{(k)} = \frac{k\phi_i}{\sum_{s\in[i-1]}\phi_s + (k-i+1)p_{\min}} = \frac{k\phi_i}{\frac{k}{\gamma}[p_{\min} + p_{\min}(\gamma-1)(1+\gamma/k)^{i-1}]} = \gamma, \forall i \in [j]. \tag{35}$$

In the following, we show OTA with the designed threshold ϕ can guarantee the target consistency and robustness in all three cases.

Case I. The prediction is $P \in [p_{\min}, \tilde{p}_1]$, where $\tilde{p}_1 = p_{\min} + p_{\min}(\eta-1)(1+\eta/k)^{\sigma^*-1}$ and $\sigma^* \in [k]$ is the largest index such that

$$\frac{1 + (\theta-1)/(1+\gamma/k)^{k-\sigma^*}}{1 + (\eta-1)(1+\eta/k)^{\sigma^*}} \le \frac{\gamma}{\eta}. \tag{36}$$

We first prove that there exists an index $\sigma^* \in [k]$ when the target consistency and robustness are set based on Equation (10). It is sufficient to show that Equation (36) holds when $\sigma^* = 1$, i.e.,

$$\frac{1 + (\theta-1)/(1+\gamma/k)^{k-1}}{1 + (\eta-1)(1+\eta/k)} \le \frac{\gamma}{\eta}. \tag{37}$$

When $1 + (\gamma-1)(1+\gamma/k)^{k-1} \ge \theta$, Equation (37) holds since we can have

$$\frac{1 + (\theta-1)/(1+\gamma/k)^{k-1}}{1 + (\eta-1)(1+\eta/k)} \le \frac{1 + (\theta-1)/(1+\gamma/k)^{k-1}}{\eta} \le \frac{\gamma}{\eta}.$$

When $1+(\gamma-1)(1+\gamma/k)^{k-1} < \theta$, we have $\xi = \left\lceil \ln\left(\frac{\theta-1}{\gamma-1}\right) / \ln\left(1+\frac{\gamma}{k}\right) \right\rceil = k$ and the corresponding target consistency is $\eta = \frac{\theta\gamma}{1+(\gamma-1)(1+\gamma/k)^k}$. Then we can construct a function $f(\gamma) = \frac{\gamma}{\eta}[1 + (\eta-1)(1+\eta/k)] - [1 + (\theta-1)/(1+\gamma/k)^{k-1}]$. By noting that $f(\gamma)$ is increasing in $[\alpha_*^{(k)}, \infty)$ and $f(\alpha_*^{(k)}) = 0$, we have $f(\alpha_*^{(k)}) \ge 0$, which gives Equation (37).

In the following, we start to prove the consistency and robustness by showing that $\alpha_i^{(k)}(\phi)$ satisfies Equation (12). We can derive the worst-case ratios $\alpha_i^{(k)}(\phi)$ given the design of ϕ as

$$c_i = p_{\min} + p_{\min}(\eta-1)\,(1+\eta/k)^{i-1}, i \in [i^*],$$
$$r_i = p_{\min} + \frac{p_{\max} - p_{\min}}{(1+\gamma/k)^{k-i+1}}, i = i^*+1, \ldots, k.$$

For $i \in [i^*]$, we have

$$\alpha_i^{(k)} = \frac{kc_i}{\sum_{j\in[i-1]} c_j + (k-i+1)p_{\min}} = \frac{kc_i}{kc_i/\eta} = \eta.$$

For $i = i^* + 1$, we have

$$\begin{aligned}\alpha_{i^*+1}^{(k)} = \frac{kr_{i^*+1}}{\sum_{j\in[i^*]} c_j + (k-i^*)p_{\min}} &= \frac{k\left[p_{\min} + (p_{\max}-p_{\min})/(1+\gamma/k)^{k-i^*}\right]}{\frac{k}{\eta}\left[p_{\min} + p_{\min}(\eta-1)\,(1+\eta/k)^{i^*}\right]} \\ &= \frac{\eta\left[1+(\theta-1)/(1+\gamma/k)^{k-\sigma^*}\right]}{1+(\eta-1)\,(1+\eta/k)^{\sigma^*}} \le \gamma,\end{aligned}$$

where the last inequality is due to Equation (36).

For $i = i^* + 2, \ldots, k+1$, based on Proposition 3.4 and $\alpha_{i^*+1}^{(k)} \le \gamma$, we have $\alpha_i^{(k)} \le \gamma$, for all $i = i^* + 2, \ldots, k+1$.

Summarizing above results, if $P \in [p_{\min}, \tilde{p}_1]$, we have $\alpha_i^{(k)} \le \eta, i \in [i^*]$ and thus $\texttt{OTA}_\phi$ is η-consistent. The worst-case competitive ratio is $\max_{i\in[k+1]} \alpha_i^{(k)} = \gamma$, and thus $\texttt{OTA}_\phi$ is γ-robust.

Case II. When the prediction $P \in (\tilde{p}_1, \tilde{p}_2]$, where $\tilde{p}_2 = \max\{\tilde{p}_1, \gamma p_{\min}\}$. This case exists only when $\tilde{p}_1 < \gamma p_{\min}$. In the following, we derive the worst-case ratios $\alpha_i^{(k)}$ given the threshold design

$$\begin{aligned} c_i &= \begin{cases} P & i = 1, \ldots, m^*, \\ \eta\frac{m^*P + (k-m^*)p_{\min}}{k} & i = m^*+1, \\ p_{\min} + (c_{m^*+1} - p_{\min})(1+\eta/k)^{i-m^*-1} & i = m^*+2, \ldots, i^*, \end{cases} \\ r_i &= p_{\min} + \frac{p_{\max}-p_{\min}}{(1+\gamma/k)^{k-i+1}}, \quad i = i^*+1, \ldots, k, \end{aligned}$$

where $m^* = \left\lceil k\frac{P/\eta - p_{\min}}{P - p_{\min}} \right\rceil$ and i^* is the largest index that ensures $\alpha_{i^*+1}^{(k)}(\phi) = \frac{kr_{i^*+1}}{\sum_{i=[i^*]} c_i + (k-i^*)p_{\min}} \le \gamma$.

For $i = 1$, $\alpha_1^{(k)} = kP/(kp_{\min}) \le \tilde{p}_2/p_{\min} \le \gamma$.

For $i = m^* + 1$ (noting that $c_i = P, \forall i \in [m^*]$),

$$\alpha_{m^*+1}^{(k)} = \frac{kc_{m^*+1}}{m^*P + (k-m^*)p_{\min}} = \eta.$$

For $i = m^* + 2, \ldots, i^*$, we have

$$\alpha_i^{(k)} = \frac{kc_i}{m^*P + \sum_{j=m^*+1}^{i-1} c_j + (k-i+1)p_{\min}} = \frac{kc_i}{\frac{k}{\eta}[p_{\min} + (c_{m^*+1} - p_{\min})(1+\eta/k)^{i-m^*-1}]} = \eta.$$

For $i = i^* + 1, \ldots, k+1$, based on $\alpha_{i^*+1}^{(k)} \le \gamma$ and Proposition 3.4, we have $\alpha_i^{(k)} \le \gamma$.

Based on above results, when prediction $P \in (\tilde{p}_1, \tilde{p}_2]$, we have $\alpha_i^{(k)} \le \eta, i = m^*+1, \ldots, i^*$, and thus the consistency of $\texttt{OTA}_\phi$ is η. Since $\max_{i\in[k+1]} \alpha_i^{(k)} \le \gamma$, the robustness is γ.

Case III. When the prediction $P \in (\tilde{p}_2, p_{\max}]$, the worst-case ratios $\alpha_i^{(k)}$ can be shown as follows when the threshold values are given by

$$\begin{aligned} z_i &= p_{\min} + p_{\min}(\gamma-1)(1+\gamma/k)^{i-1}, \quad i \in [j^*], \\ c_i &= \begin{cases} P, & i = j^*+1, \ldots, m^*, \\ \eta\frac{\sum_{i\in[j^*]} z_i + (m^*-j^*)P + (k-m^*)p_{\min}}{k}, & i = m^*+1, \\ p_{\min} + (c_{m^*+1} - p_{\min})(1+\eta/k)^{i-m^*-1}, & i = m^*+2, \ldots, i^*, \end{cases} \\ r_i &= p_{\min} + \frac{p_{\max}-p_{\min}}{(1+\gamma/k)^{k-i+1}}, \quad i = i^*+1, \ldots, k, \end{aligned}$$

where $j^* = \left\lceil \ln\left(\frac{P/p_{\min}-1}{\gamma-1}\right)/\ln\left(1+\frac{\gamma}{k}\right) \right\rceil$, $m^* = j^* + \left\lceil \frac{kP/\eta - kp_{\min}[1+(\gamma-1)(1+\gamma/k)^{j^*}]/\gamma}{P-p_{\min}} \right\rceil$, and i^* is the largest index such that $\alpha_{i^*+1}^{(k)}(\phi) = \frac{kr_{i^*+1}}{\sum_{i\in[j^*]} z_i + \sum_{i=j^*+1}^{i^*} c_i + (k-i^*)p_{\min}} \le \gamma$.

For interval $i \in [j^*]$, we have $\alpha_i^{(k)} \le \gamma$ since z_i is designed based on Equation (17) in Proposition 3.5. j^* is determined as the largest index such that $\phi_{j^*} < P \le p_{\max}$ and thus $j^* \le \xi$. Note that as the prediction P increases, m^* also increases to ensure the consistency. When $P = p_{\max}$, we have $j^* = \xi$ and m^* is given by

$$m^* = \xi + \left\lfloor \frac{kp_{\max}/\eta - kp_{\min}[1 + (\gamma-1)(1+\gamma/k)^{\xi}]/\gamma}{p_{\max} - p_{\min}} \right\rfloor,$$

which cannot exceed the total number of available units k. Since the target consistency η and robustness γ are given in Equation (10), we have

$$\frac{kp_{\max}/\eta - kp_{\min}[1 + (\gamma-1)(1+\gamma/k)^{\xi}]/\gamma}{p_{\max} - p_{\min}} = k \cdot \frac{\left[1 + (\gamma-1)(1+\frac{\gamma}{k})^{\xi}\right]/\gamma + (\theta-1)(1-\frac{\xi}{k}) - [1 + (\gamma-1)(1+\gamma/k)^{\xi}]/\gamma}{\theta - 1} = k - \xi,$$

and thus the target η and γ can ensure $m^* \le k$.

For $i = m^* + 1$ (that corresponds to the interval $[\phi_{j^*+1}, \phi_{m^*+1})$), we have

$$\alpha_{m^*+1}^{(k)} = \frac{kc_{m^*+1}}{\sum_{j\in[j^*]} z_j + (m^* - j^*)P + (k - m^*)p_{\min}} = \eta.$$

Finally, we have $\alpha_i^{(k)} \le \eta$ for $i = m^* + 2, \dots, i^*$ and $\alpha_i^{(k)} \le \gamma$ for $i = i^* + 1, \dots, k+1$ by following the same proof as Case II.

In this case, the worst-case ratio $\alpha_i^{(k)} \le \eta, i = j^* + 1, \dots, i^*$ and $\max_{i\in[k+1]} \alpha_i^{(k)} = \gamma$. Thus, $\texttt{OTA}_\phi$ is η-consistent and γ-robust.

B.2 Proof of Theorem 3.6

To show the lower bound, we consider a special family of instances, and show that under the special instances, any γ-robust online algorithm at least has a consistency η lower bounded by $\Lambda(\gamma)$.

Definition B.1 (reverse p-instance). *A reverse p-instance $\mathcal{I}_p$ consists of a sequence of prices that decrease continuously from $p_{\max}$ to p and spike to $p_{\max}$ at the end.*

Let $g(p) : [p_{\min}, p_{\max}] \to \{0, 1, \dots, k\}$ denote the cumulative trading decision of an online algorithm when it executes the instance $\mathcal{I}_p$ before the compulsory trading in the last step. Since online decision is irrevocable, $g(p)$ is non-decreasing as p decreases from $p_{\max}$ to $p_{\min}$. In addition, items must be traded at the lowest price $p_{\min}$. Thus, we must have $g(p_{\min}) = k$. Given an online algorithm, let $\mathcal{I}_{\hat{p}_i}$ denote the first instance, in which the algorithm trades the i-th item, i.e., $\hat{p}_i = \sup_{\{p\in[p_{\min},p_{\max}]:g(p)\ge i\}} p$.

For any γ-robust online algorithm, $\{\hat{p}_i\}_{i\in[k]}$ must satisfy

$$\frac{\sum_{j\in[i-1]} \hat{p}_j + (k-i+1)p_{\max}}{k\hat{p}_i} \le \gamma, i \in [k+1], \qquad \hat{p}_i \ge p_{\min}, i \in [k], \tag{41}$$

where $\hat{p}_0 := p_{\max}$ and $\hat{p}_{k+1} := p_{\min}$. Based on Equation (41), we have for $i \in [k]$

$$\hat{p}_i \ge \max\left\{ p_{\max} - p_{\max}\left(1 - \frac{1}{\gamma}\right)\left(1 + \frac{1}{k\gamma}\right)^{i-1}, p_{\min} \right\}. \tag{42}$$

Suppose the prediction is $P = p_{\min}$. To ensure consistency under the instance $\mathcal{I}_{p_{\min}}$, any γ-robust online algorithm must have

$$\eta \ge \frac{\texttt{ALG}(\mathcal{I}_{p_{\min}})}{\texttt{OPT}(\mathcal{I}_{p_{\min}})} = \frac{\sum_{i\in[k]} \hat{p}_i}{kp_{\min}} \ge \frac{\sum_{i\in[\zeta]} \hat{p}_i + (k-\zeta)p_{\min}}{kp_{\min}} \tag{43a}$$

$$\ge \frac{\sum_{i\in[\zeta]} \left[p_{\max} - p_{\max}\left(1 - \frac{1}{\gamma}\right)\left(1 + \frac{1}{k\gamma}\right)^{i-1}\right] + (k-\zeta)p_{\min}}{kp_{\min}} \tag{43b}$$

$$= \theta\left[\gamma - (\gamma-1)\left(1 + \frac{1}{\gamma k}\right)^{\zeta}\right] - (\theta - 1)\left(1 - \frac{\zeta}{k}\right), \tag{43c}$$

where ζ satisfies $p_{\max} - p_{\max}\left(1 - \frac{1}{\gamma}\right)\left(1 + \frac{1}{k\gamma}\right)^{\zeta-1} > p_{\min} \ge p_{\max} - p_{\max}\left(1 - \frac{1}{\gamma}\right)\left(1 + \frac{1}{k\gamma}\right)^{\zeta}$, which gives $\zeta = \left\lceil \ln\left(\frac{\theta-1}{\theta-\theta/\gamma}\right) / \ln\left(1 + \frac{1}{\gamma k}\right) \right\rceil$.

B.3 Proof of Theorem 3.7

The threshold values $\psi = \{\psi_i\}_{i\in[k]}$ for k-min is a non-increasing sequence $\psi_1 \geq \psi_2 \geq \cdots \geq \psi_k$. We also define $\psi_0 = p_{\max}$ and $\psi_{k+1} = p_{\min}$. The threshold ψ divides the uncertainty range $[p_{\min}, p_{\max}]$ into $k+1$ regions. If $\psi_i = \psi_{i+1} = \cdots = \psi_j < \psi_{j+1}$, we use $j+1$ to index the interval $[\psi_i, \psi_{j+1})$. Excluding the compulsory trading at the end, if OTA trades i items, the worst-case price sequence is

$$\psi_1, \psi_2, \ldots, \psi_i, \underbrace{\psi_{i+1}+\epsilon, \ldots, \psi_{i+1}+\epsilon}_{k}, \underbrace{p_{\max}, \ldots, p_{\max}}_{k},$$

where $\epsilon > 0$ and $\epsilon \to 0$. The worst-case ratio under this price sequence is

$$\varphi_i^{(k)} := \varphi_i^{(k)}(\psi) = \frac{\sum_{j\in[i-1]} \psi_j + (k-i+1)p_{\max}}{k\psi_i}. \tag{44}$$

We start by proving the two following two propositions.

Proposition B.2. *For a given $j \in [k-1]$, if the threshold values after j are given by*

$$\psi_i = p_{\max} - \frac{p_{\max} - p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, i = j+1, \ldots, k, \tag{45}$$

and $\varphi_{j+1}^{(k)} \leq \gamma$, then $\varphi_i^{(k)}(\psi) \leq \gamma, i = j+2, \ldots, k+1$.

Proof of Proposition B.2. Given that $\varphi_{j+1}^{(k)} \leq \gamma$, we have

$$\sum\nolimits_{s\in[j]} \psi_s + (k-j)p_{\max} \leq k\gamma\psi_{j+1}. \tag{46}$$

We then have

$$\varphi_{j+2}^{(k)} = \frac{\sum_{s\in[j+1]} \psi_s + (k-j-1)p_{\max}}{k\psi_{j+2}} = \frac{\sum_{s\in[j]} \psi_s + (k-j)p_{\max} + \psi_{j+1} - p_{\max}}{k\psi_{j+2}} \tag{47a}$$

$$\leq \frac{k\gamma\psi_{j+1} + \psi_{j+1} - p_{\max}}{k\psi_{j+2}} \tag{47b}$$

$$= \gamma \frac{p_{\max} + (1+\frac{1}{\gamma k})(\psi_{j+1} - p_{\max})}{\psi_{j+2}} = \gamma, \tag{47c}$$

where the inequality (47b) holds due to inequality (46) and the last equality (47c) is based on the design of the threshold values in equation (45). By following the same step, the worst-case ratios in the following intervals can be shown that $\varphi_i^{(k)} \leq \gamma, \forall i = j+3, \ldots, k+1$.

Proposition B.3. *For a given $j \in [\zeta]$, if the first j threshold values are given by*

$$\psi_i = p_{\max} - p_{\max}\left(1-\frac{1}{\gamma}\right)\left(1+\frac{1}{\gamma k}\right)^{i-1}, i \in 1, \ldots, j, \tag{48}$$

then $\varphi_i^{(k)}(\psi) \leq \gamma, i = 1, \ldots, j$.

Proof of Proposition B.3. When the first j threshold values are designed based on Equation (48), $\forall i \in [j]$, the worst-case ratio of the i-th interval of OTA can be shown as

$$\varphi_i^{(k)} = \frac{\sum_{s\in[i-1]} \psi_s + (k-i+1)p_{\max}}{k\psi_i} = \frac{k\gamma[p_{\max} - p_{\max}(1-1/\gamma)(1+1/(k\gamma))^{i-1}]}{k\psi_i} = \gamma. \tag{49}$$

In the following, we show OTA with the designed threshold ψ can guarantee the target consistency and robustness in the k-min search in Cases IV-VI.

Case IV. When $P \in [\tilde{p}_1, p_{\max}]$, where $\tilde{p}_1 = p_{\max} - p_{\max}(1-1/\eta)(1+1/(\eta k))^{\sigma^*-1}$ and $\sigma^* \in [k]$ is the largest index such that

$$\frac{1-(1-1/\eta)(1+\frac{1}{\eta k})^{\sigma^*}}{1-(1-1/\theta)/(1+\frac{1}{\gamma k})^{k-\sigma^*}} \leq \frac{\gamma}{\eta}. \tag{50}$$

In addition, $i^* = \sigma^*$ and $\tilde{p}_1 = c_{i^*}$. We derive the worst-case ratios $\varphi_i^{(k)}(\psi)$ given the design

$$c_i = p_{\max} - p_{\max}\,(1-1/\eta)\,(1+1/(\eta k))^{i-1}, \quad i \in [i^*],$$

$$r_i = p_{\max} - \frac{p_{\max} - p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, \quad i = i^*+1, \ldots, k.$$

For $i \in [i^*]$, we have

$$\varphi_i^{(k)} = \frac{\sum_{j\in[i-1]} c_j + (k-i+1)p_{\max}}{kc_i} = \frac{k\eta c_i}{kc_i} = \eta.$$

For $i = i^* + 1$, we have

$$\varphi_{i^*+1}^{(k)} = \frac{\sum_{j\in[i^*]} c_j + (k-i^*)p_{\max}}{kr_{i^*+1}} = \frac{k\eta\left[p_{\max} - p_{\max}\left(1-\frac{1}{\eta}\right)\left(1+\frac{1}{\eta k}\right)^{i^*}\right]}{k\left[p_{\max} - \frac{p_{\max}-p_{\min}}{(1+1/(\gamma k))^{k-i^*}}\right]}$$
$$= \frac{\eta\left[1-(1-1/\eta)(1+\frac{1}{\eta k})^{i^*}\right]}{1-(1-1/\theta)/(1+\frac{1}{\gamma k})^{k-i^*}} \le \gamma,$$

where the last inequality is due to Equation (50).

For $i = i^* + 2, \ldots, k+1$, based on Proposition B.2 and $\varphi_{i^*+1}^{(k)} \le \gamma$, we have $\varphi_i^{(k)} \le \gamma$, for all $i = i^*+2, \ldots, k+1$.

Summarizing above results, in this case, we have $\varphi_i^{(k)} \le \eta, i \in [i^*]$ and thus $\texttt{OTA}_\psi$ is η-consistent. The worst-case competitive ratio is $\max_{i\in[k+1]} \varphi_i^{(k)} = \gamma$ and thus $\texttt{OTA}_\psi$ is γ-robust.

Case V. When the prediction $P \in [\tilde{p}_2, \tilde{p}_1)$, where $\tilde{p}_2 = \min\{\tilde{p}_1, p_{\max}/\gamma\}$. This case exists only when $\tilde{p}_1 > p_{\max}/\gamma$. In the following, we derive the worst-case ratios $\varphi_i^{(k)}$ given the threshold design

$$c_i = \begin{cases} P, & i = 1, \ldots, m^*, \\ \frac{m^*P + (k-m^*)p_{\max}}{\eta k}, & i = m^*+1, \\ p_{\max} - (p_{\max} - c_{m^*+1})\,(1+1/(\eta k))^{i-m^*-1}, & i = m^*+2, \ldots, i^*, \end{cases}$$
$$r_i = p_{\max} - \frac{p_{\max}-p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, \quad i = i^*+1, \ldots, k,$$

where $m^* = \left\lceil k\frac{p_{\max}-\eta P}{p_{\max}-P}\right\rceil$ and i^* is the largest index that ensures $\varphi_{i^*+1}^{(k)}(\psi) \le \gamma$.

For $i = 1$, $\varphi_1^{(k)} = (kp_{\max})/kP \le p_{\max}/\tilde{p}_1 \le \gamma$.

For $i = m^*+1$,

$$\varphi_{m^*+1}^{(k)} = \frac{m^*P + (k-m^*)p_{\max}}{kc_{m^*+1}} = \eta.$$

For $i = m^*+2, \ldots, i^*$, we have

$$\varphi_i = \frac{m^*P + \sum_{j=m^*+1}^{i-1} c_j + (k-i+1)p_{\max}}{kc_i} = \frac{k\eta\left[p_{\max} - (p_{\max}-c_{m^*+1})\left(1+\frac{1}{\eta k}\right)^{i-m^*-1}\right]}{kc_i} = \eta.$$

For $i = i^*+1, \ldots, k+1$, based on $\varphi_{i^*+1}^{(k)} \le \gamma$ and Proposition B.2, we have $\varphi_i^{(k)} \le \gamma$.

Based on above results, when prediction $P \in [\tilde{p}_2, \tilde{p}_1)$, the consistency of $\texttt{OTA}_\psi$ is η. Since $\max_{i\in[k+1]} \varphi_i^{(k)} \le \gamma$, the robustness is γ.

Case VI. When $P \in [p_{\min}, \tilde{p}_2)$. The worst-case ratios $\varphi_i^{(k)}$ can be shown as follows when thresholds are

$$z_i = p_{\max} - p_{\max}\,(1-1/\gamma)\,(1+1/(\gamma k))^{i-1}, \quad i \in [j^*],$$
$$c_i = \begin{cases} P, & i = j^*+1, \ldots, m^*, \\ \frac{\sum_{i\in[j^*]} z_i + (m^*-j^*)P + (k-m^*)p_{\max}}{\eta k}, & i = m^*+1, \\ p_{\max} - (p_{\max}-c_{m^*+1})\,(1+1/(\eta k))^{i-1}, & i = m^*+2, \ldots, i^*, \end{cases}$$
$$r_i = p_{\max} - \frac{p_{\max}-p_{\min}}{(1+1/(\gamma k))^{k-i+1}}, \quad i = i^*+1, \ldots, k,$$

where $j^* = \left\lceil \ln\left(\frac{1-P/p_{\max}}{1-1/\gamma}\right)/\ln(1+1/(\gamma k))\right\rceil$, and $m^* = j^* + \left\lceil \frac{kp_{\max}[1-(1-1/\gamma)(1+1/(\gamma k))^{j^*}]-kP\eta}{p_{\max}-P}\right\rceil$, and i^* is the largest index such that $\varphi_{i^*+1}^{(k)}(\psi) \le \gamma$.

For interval $i \in [j^*]$, we have $\varphi_i^{(k)} \le \gamma$ since z_i is designed based on Equation (48) in Proposition B.3. j^* is determined as the largest index such that $\psi_{j^*} > P \ge p_{\min}$.

For interval $i = m^* + 1$ (i.e., $[\psi_{j^*+1}, \psi_{m^*+1}]$), we have

$$\varphi_{m^*+1}^{(k)} = \frac{\sum_{j\in[j^*]} z_j + (m^* - j^*)P + (k - m^*)p_{\max}}{kc_{m^*+1}} = \eta.$$

Finally, we have $\varphi_i^{(k)} \le \eta$ for interval $i = m^* + 2, \ldots, i^*$ and $\varphi_i^{(k)} \le \gamma$ for interval $i = i^* + 1, \ldots, k + 1$ by following the same proof as Case V.

In this case, the worst-case ratio $\varphi_i^{(k)} \le \eta, i = j^* + 1, \ldots, i^*$ and $\max_{i\in[k+1]} \varphi_i^{(k)} = \gamma$. Thus, $\texttt{OTA}_\psi$ is η-consistent and γ-robust.

C PROOFS ON ONLINE SEARCH WITH INVENTORY DYNAMICS

C.1 Proof of Lemma 4.3

First, we can observe the inventory dynamics is enforced in Line 15 of Algorithm 2. Since the online decision is $x_t = \max\left\{\sum_{j\in[i]} x_t^{(n,j)}, d_t - s_{t-1}\right\}$ (Line 14), we have $x_t \ge d_t - s_{t-1}$ and thus $s_t \ge 0, \forall t \in [T]$. Therefore, we only need to check the capacity constraint of the inventory. For any t in a busy period of the n-th interval with i parallel online search problems, observe

$$\begin{aligned} s_t &= \sum_{j=1}^{i} (m_t^{(n,j)} - 1) - \sum_{j=2}^{i} k^{(n,j)} \\ &\le \sum_{j=1}^{i} k^{(n,j)} - \sum_{j=2}^{i} k^{(n,j)} = k^{(n,1)} = B, \end{aligned}$$

where $m_t^{(n,j)} - 1$ denotes the number of purchased items of the j-th online search problem up to time t, and $m_t^{(n,j)} - 1 \le k^{(n,j)}$ in the online search problem. Thus, $\sum_{j=1}^{i}(m^{(n,j)} - 1)$ and $\sum_{j=2}^{i} k^{(n,j)}$ represent the total number of purchased items and the total demand from the start of the busy period to slot t, respectively. Thus, the inventory level never violates the capacity constraint.

C.2 Proof of Lemma 4.4

The total cost of the online algorithm can be upper bounded by the costs of purchasing items during the busy period and idle period. In the busy period of interval n, there are a total of i_n virtual search problems and the i-th problem buys $m^{(n,i)} - 1$ items. Since each virtual search problem buys items based on the OTA, the price of j-th purchased item of problem i in interval n is at most $\psi_j^{(n,i)}$. Thus, the cost of busy period is upper bounded by $\sum_{n\in[N]} \sum_{i\in[i_n]} \sum_{j\in[m^{(n,i)}-1]} \psi_j^{(n,i)}$. In addition, the price of purchasing items in the busy period is upper bounded by $p_{\max}$. Thus,

$$\texttt{ALG}(\mathcal{I}) \le \sum_{n\in[N]} \sum_{i\in[i_n]} \sum_{j\in[m^{(n,i)}-1]} \psi_j^{(n,i)} + \left[\sum_{t\in[T]} d_t + s_T - \sum_{n\in[N]} \sum_{i\in[i_n]} (m^{(n,i)} - 1)\right] p_{\max} \tag{54a}$$

$$= \sum_{n\in[N]} \sum_{i\in[i_n]} \left[\sum_{j\in[m^{(n,i)}-1]} \psi_j^{(n,i)} + (k^{(n,i)} - m^{(n,i)} + 1)p_{\max}\right] + [R + s_T]\, p_{\max}, \tag{54b}$$

where $R := \sum_{t\in[T]} d_t - \sum_{n\in[N]} \sum_{i\in[i_n]} k^{(n,i)} = \sum_{n\in[N]} \sum_{t\in\mathcal{T}_n^0} d_t - NB$.

C.3 Proof of Lemma 4.5

Let $F(\hat{s}_n, y_n)$ denote the cost of the offline optimum in the busy period of the interval n, where $\hat{s}_n$ is the initial inventory in the beginning of the interval n and y_n is the number of purchased items in this busy period. Let $F^*(\hat{s}_n, y_n)$ denote the minimum cost of purchasing y_n items with initial inventory $\hat{s}_n$ in the busy period of interval n without considering the price and demand of other intervals. Then we can have

$$F_n(\hat{s}_n, y_n) \ge F_n^*(\hat{s}_n, y_n) \ge F_n^*(0, y_n), \forall n \in [N], \tag{55}$$

where the first inequality is by definition and the second inequality is because a lower initial inventory gives more flexibility to minimize the cost of purchasing the same amount of items.

Further, we can have, for all $n \in [N]$,

$$F_n^*\left(0, \sum_{i\in[i_n]} k^{(n,i)}\right) - F_n^*(0, y_n) \le \left(\sum_{i\in[i_n]} k^{(n,i)} - y_n\right) \frac{p_{\max}}{\gamma}. \tag{56}$$

Note that $\sum_{i\in[i_n]} k^{(n,i)}$ is the sum of inventory capacity and the total demand in the busy period. Thus, $y_n \le \sum_{i\in[i_n]} k^{(n,i)}$. In addition, there must exist prices no larger than $\frac{p_{\max}}{\gamma}$ in the busy period. Therefore, above inequality holds since the price for purchasing any additional items in the busy period by the cost minimum algorithm will be no larger than $\frac{p_{\max}}{\gamma}$. In addition, we have

$$F_n^*\left(0, \sum_{i\in[i_n]} k^{(n,i)}\right) \ge \sum_{i\in[i_n]} \psi_{m^{(n,i)}}^{(n,i)} k^{(n,i)}, \tag{57}$$

because $\psi^{(n,i)}_{m^{(n,i)}}$ is the minimum price of the i-th virtual search problem of interval n in its lifecycle. Combining Equations (55)- (57) gives

$$F_n(\hat{s}_n, y_n) \geq \sum\nolimits_{i\in[i_n]} \psi^{(n,i)}_{m^{(n,i)}} k^{(n,i)} - \left[\sum\nolimits_{i\in[i_n]} k^{(n,i)} - y_n\right] \frac{p_{\max}}{\gamma}. \tag{58}$$

The offline optimal cost can be lower bounded by

$$\texttt{OPT}(\mathcal{I}) \geq \sum\nolimits_{n\in[N]} F_n(\hat{s}_n, y_n) + \left[\sum\nolimits_{t\in[T]} d_t - \sum\nolimits_{n\in[N]} y_n\right] \frac{p_{\max}}{\gamma} \tag{59a}$$

$$\geq \sum\nolimits_{n\in[N]} \sum\nolimits_{i\in[i_n]} \psi^{(n,i)}_{m^{(n,i)}} k^{(n,i)} + \frac{R\cdot p_{\max}}{\gamma}, \tag{59b}$$

where the first inequality holds since the minimum price during idle period is $\frac{p_{\max}}{\gamma}$, and the second inequality is obtained by substituting inequality (58).

D CONTINUOUS VERSION OF ONLINE SEARCH WITH INVENTORY DYNAMICS

We present algorithms and results for the continuous version of the online search with inventory dynamics (OSID). In this problem, integral constraints on the purchasing decisions and inventory levels are relaxed, i.e., the constraint (24e) is relaxed to $x_t \geq 0, s_t \geq 0, \forall t \in [T]$. We show that the learning-augmented algorithm for OSID (i.e., Algorithm 2) can be extended to solve the continuous OSID and achieve the Pareto-optimal trade-off between consistency and robustness. In Section 5, we further apply the extended algorithm to solve the storage-assisted energy procurement problem for data centers in the electricity markets.

Learning-augmented algorithm for continuous OSID. We use a modified Algorithm 2 to solve the continuous OSID. Specifically, we choose a large integer K as an additional input. In this algorithm, for each created virtual search problem (n, i) with prediction $P^{(n,i)}$, we treat it as a $(K, P^{(n,i)})$-search problem, and use the threshold values $\psi(K, P^{(n,i)})$ designed in Equations (21)-(23) to solve each virtual problem in line 8 to line 13 of Algorithm 2. Then we change the online decisions in line 14 by scaling them back from the virtual problems, i.e.,

$$x_t = \max\left\{\sum_{j\in[i]} \frac{x_t^{(n,j)}\cdot k^{(n,j)}}{K}, d_t - s_{t-1}\right\}, \tag{60}$$

where $k^{(n,i)}$ is the capacity of the virtual search problem.

Consistency-robustness results. We first note that K-min search ($K \to \infty$) is a special case of the continuous OSID. Based on Theorem 3.6, we can have the following lower bound result.

Corollary D.1. *For continuous* OSID *with predictions, any γ-robust deterministic learning-augmented algorithm must have a consistency lower bounded by* $\Lambda(\gamma) = \gamma - (\theta-1)\left[1 - \gamma\ln\left(\frac{\theta-1}{\theta-\theta/\gamma}\right)\right]$.

Based on this lower bound, for a given $\lambda \in [0,1]$, we set the target consistency and robustness as

$$\gamma^{(\infty)}(\lambda) = \varphi_*^{(\infty)} + (1-\lambda)(\theta - \varphi_*^{(\infty)}), \quad \eta^{(\infty)}(\lambda) = \Lambda(\gamma^{(\infty)}(\lambda)), \tag{61}$$

where $\varphi_*^{(\infty)}$ is the optimal competitive ratio of K-min search (as $K \to \infty$), and is the solution of $\frac{1-1/\theta}{1-1/\varphi} = \exp\left(\frac{1}{\varphi}\right)$.

Based on upper bound results in Theorem 3.7, we further have the consistency-robustness result for the modified Algorithm 2.

Corollary D.2. *Given $\lambda \in [0,1]$, the modified Algorithm 2 (with $K \to \infty$) is $\eta^{(\infty)}(\lambda)$-consistent and $\gamma^{(\infty)}(\lambda)$-robust for the learning-augmented continuous* OSID *when $\psi_{K,P}(\cdot)$ are given by Equations* (21)-(23). *The modified Algorithm 2 is Pareto-optimal.*

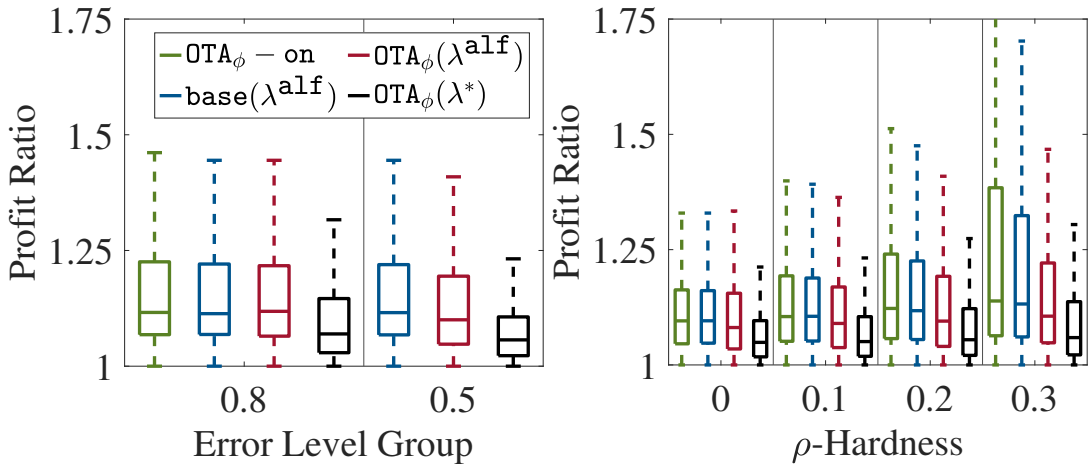
As a remark, the existing work [27] studies this continuous OSID and designs an online algorithm that achieves the optimal competitive ratio. Algorithm 2 can be considered as the learning-augmented extension for the worst-case optimized algorithm. In addition, Algorithm 2 can also achieve the optimal competitive ratio for OSID when we set $\lambda = 1$ to ignore the predictions.

E k-SEARCH EXPERIMENTAL RESULTS IN GENERAL FINANCIAL MARKETS

We present our main results for the application of trading in electricity markets, but our algorithms are applicable to general online search problems. In the additional experiments in this section, we evaluate our algorithms for learning-augmented k-search in a general financial market. We use a 5-year history of Bitcoin (BTC) prices collected from the Gemini Exchange [11] in this application.

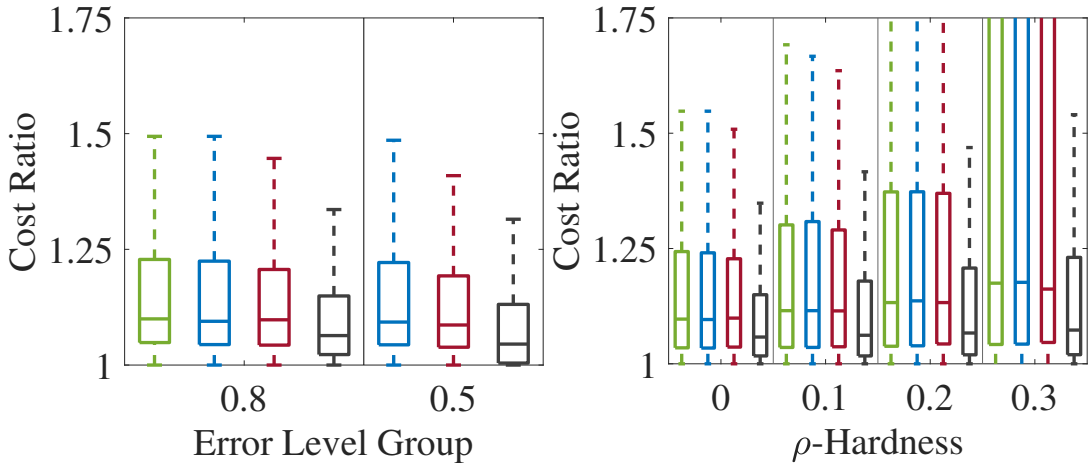
E.1 Experimental setup

We consider the problem of buying or selling BTC over 3-week trading periods. Prices are observed every 10 minutes such that $T = 3024$. We set the units available to be traded to $k = 100$. The price limits $p_{\max}$ and $p_{\min}$ are conservatively set according to the maximum and minimum over the entire 5-year period. To generate a prediction P, we use the observed maximum (or minimum) exchange rate of the previous instance. To evaluate the impact of prediction quality, we adjust the prediction error level with a scalar value between 0 and 1,



(a) Prediction quality (b) Instance hardness

Figure 5: Comparing algorithms for k-max search.



(a) Prediction quality (b) Instance hardness

Figure 6: Comparing algorithms for k-min search.

where 0 indicates perfect predictions and 1 indicates non-adjusted predictions. Although BTC has experienced drastic price fluctuations, these occurrences are still rare. Therefore, to evaluate the performance in worst-case settings, we define an instance ρ–hard by giving it a fixed probability ρ of observing the worst-case price in the last time slot.

E.2 Comparison algorithms

We compare the following four algorithms.

- ▹ ($\texttt{OTA-on}$) The worst-case optimized online algorithm that does not take into account predictions, but guarantees optimal competitive ratios.
- ▹ ($\texttt{OTA}(\lambda^*)$) Our proposed algorithm with the best possible hyper-parameter λ^* chosen offline. This algorithm is not practical since it is fed with the optimal parameter, but it represents the best improvement from available predictions.
- ▹ ($\texttt{OTA}(\lambda^{\texttt{alf}})$) Our proposed algorithm that uses an online learning algorithm (i.e., adversarial Lipschitz algorithm [20]) to adaptively select λ.
- ▹ ($\texttt{base}(\lambda^{\texttt{alf}})$) The baseline algorithm that is introduced in Section 2.3 and uses the same online learning approach to select λ.

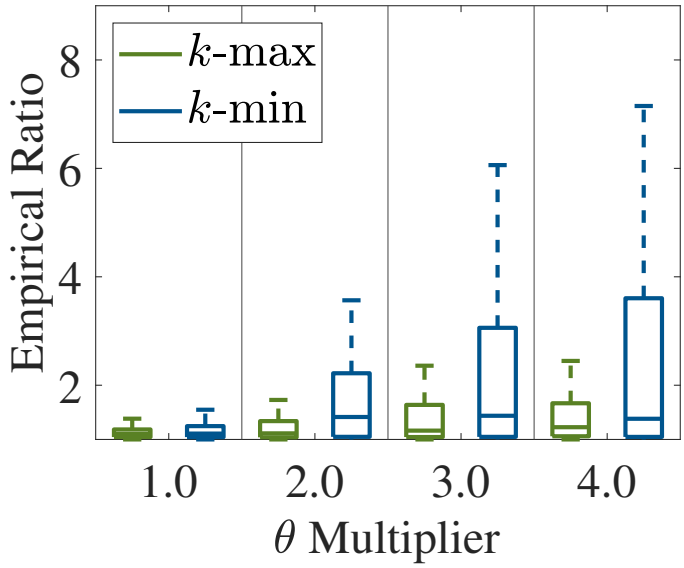


Figure 7: Impact of price uncertainty

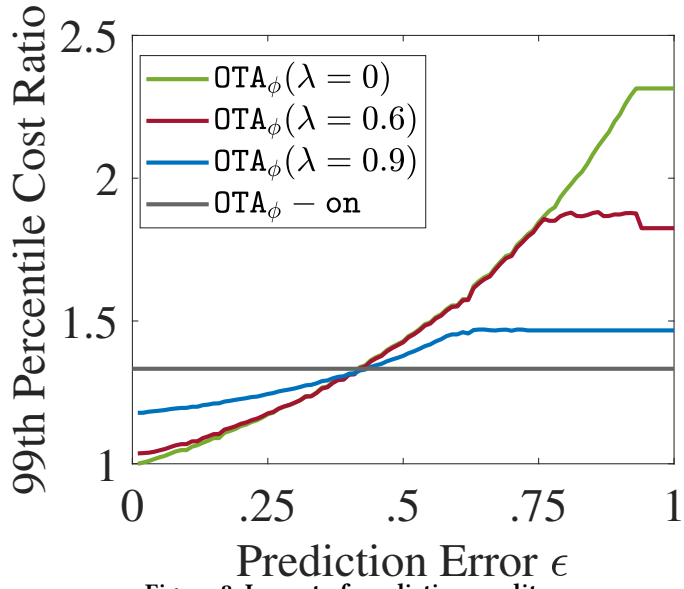


Figure 8: Impact of prediction quality

E.3 Experimental results

We compare the empirical competitive ratios of four algorithms for k-max and k-min search problems in Figures 5 and 6, respectively. Figures 5(a) and 6(a) show $\texttt{OTA}(\lambda^{\texttt{alf}})$ can make the best use of prediction among all algorithms and benefit most as the prediction quality improves (from 0.8 to 0.5). In addition to incorporating good predictions, $\texttt{OTA}(\lambda^{\texttt{alf}})$ also maintains the best robustness, which degrades slowly when the instances become harder, as shown in Figures 5(b) and 6(b). Thus, the experiments show the good potential of $\texttt{OTA}(\lambda^{\texttt{alf}})$ to achieve the best of both worlds. Since k-max and k-min are known to have different worst-case performances [18], we further compare $\texttt{OTA}(\lambda^{\texttt{alf}})$ for k-max and k-min. The empirical performance of k-min is generally worse than k-max, which is most apparent for larger values of uncertainty parameter θ in Figure 7. This is consistent with the worse consistency-robustness trade-off of k-min as shown in Figure 1. In Figure 8, we further investigate the impact of prediction quality on the worst-case performance of our algorithms. Specifically, we consider 100 instances of k-max search while adding prediction errors ϵ (normalized by $p_{\max} - p_{\min}$), and report the 99-th percentile of the empirical ratios. We can observe that even when the prediction is almost incorrect by $(p_{\max} - p_{\min})/2$, our algorithms can still outperform the worst-case optimized algorithm $\texttt{OTA-on}$.