Online Linear Optimization with Inventory Management Constraints

LIN YANG*, The Chinese University of Hong Kong, China
MOHAMMAD H. HAJIESMAILI*, University of Massachusetts Amherst, USA
RAMESH SITARAMAN, University of Massachusetts Amherst, USA
ADAM WIERMAN, California Institute of Technology, USA
ENRIQUE MALLADA, Johns Hopkins University, USA
WING S. WONG, The Chinese University of Hong Kong, China

This paper considers the problem of online linear optimization with inventory management constraints. Specifically, we consider an online scenario where a decision maker needs to satisfy her time-varying demand for some units of an asset, either from a market with a time-varying price or from her own inventory. In each time slot, the decision maker is presented a (linear) price and must immediately decide the amount to purchase for covering the demand and/or for storing in the inventory for future use. The inventory has a limited capacity and can be used to buy and store assets at low price and cover the demand when the price is high. The ultimate goal of the decision maker is to cover the demand at each time slot while minimizing the cost of buying assets from the market. We propose `ARP`, an online algorithm for linear programming with inventory constraints, and `ARPRate`, an extended version that handles rate constraints to/from the inventory. Both `ARP` and `ARPRate` achieve optimal competitive ratios, meaning that no other online algorithm can achieve a better theoretical guarantee. To illustrate the results, we use the proposed algorithms in a case study focused on energy procurement and storage management strategies for data centers.

CCS Concepts: • **Theory of computation → Online algorithms**; **Linear programming**; • **Hardware →** *Energy generation and storage*; *Power estimation and optimization.*

Additional Key Words and Phrases: Online linear optimization; inventory management; competitive online algorithms; energy procurement; data center

ACM Reference Format:
Lin Yang, Mohammad H. Hajiesmaili, Ramesh Sitaraman, Adam Wierman, Enrique Mallada, and Wing S. Wong. 2020. Online Linear Optimization with Inventory Management Constraints. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 1, Article 16 (March 2020), 29 pages. https://doi.org/10.1145/3379482

*Both authors contributed equally to this research.

Authors' addresses: Lin Yang, The Chinese University of Hong Kong, China, Hong Kong, linyang@cs.umass.edu; Mohammad H. Hajiesmaili, hajiesmaili@cs.umass.edu, University of Massachusetts Amherst, USA, 140 Governors Dr. Amherst, MA, 01002; Ramesh Sitaraman, University of Massachusetts Amherst, USA, 140 Governors Dr. Amherst, MA, 01002, ramesh@cs.umass.edu; Adam Wierman, California Institute of Technology, USA, 1200 E. California Blvd. Pasadena, CA, 91125, adamw@caltech.edu; Enrique Mallada, Johns Hopkins University, USA, 3400 N Charles St, Baltimore, MD, 21218, mallada@jhu.edu; Wing S. Wong, The Chinese University of Hong Kong, China, Hong Kong, wswong@ie.cuhk.edu.hk.

2476-1249/2020/3-ART16 $15.00
https://doi.org/10.1145/3379482

1 INTRODUCTION

Online optimization and decision making under uncertainty are classical topics that have been studied in a broad set of applications using a wide range of theoretical tools. On the theoretical side, the topic has been approached from the perspective of online algorithms using competitive analysis [17], online learning with the regret as the performance metric [37, 58], and reinforcement learning with different modeling techniques such as Markov Decision Processes (MDPs) [51]. On the application side, recent scenarios where theoretical results have had an impact for real-world design include data center optimization [10, 11, 41, 50, 53], energy systems [12, 34, 35, 40, 61, 65], cloud management [44, 62], and computer and communication networks [28, 36].

This paper studies a variation of the classical online optimization formulation: *online linear optimization with inventory management constraints* (OLIM). In this problem, in each slot, a decision maker should satisfy a demand of $d(t) \geq 0$ units of an asset arriving online. The demand should be satisfied either from a market with time-varying price or from her own inventory with a finite capacity of B. In slot t, the decision maker is presented an online price $p(t) \geq 0$, and must decide the amount to procure from the market, $x(t) \geq 0$, which can be used to cover the demand $d(t)$ or to store in the inventory for future use. The goal of the decision maker is to cover the demand over the time horizon while minimizing the cost of buying from the market through the use of the inventory.

While online linear optimization has been studied for decades [13, 23, 39, 48], progress on online linear optimization with inventory constraints has only begun to occur in recent years, e.g., [21, 22, 42], with most work focusing on special cases. The reason is that inventory management constraints couple the decision of the online agent across rounds in an unavoidable way. So, without knowledge of future prices and demands, it is challenging to design online algorithms with provable worst-case guarantees.

OLIM is of particular interest due to its relationships to classical problems in the online algorithms community. Specifically, OLIM is a generalization of several classic online algorithmic problems in the area of online search or conversion problems [45]. Notable examples include the time series search and one-way trading problems [25], the multiple-choice secretary problem [14], the k-search problem [43], and online linear programs with covering constraints [18]. In contrast to these problems, in OLIM there is uncertainty not just in the arriving costs, but also in the arrival of the demand. In addition, OLIM incorporates inventory management constraints to track the arrival and departures to the inventory over time.

Beyond its algorithmic interest, our interest in OLIM stems from its connection to the problem of energy storage management when procuring energy from the electricity markets. Energy procurement and storage management is a challenging problem for large-scale electricity customers such as data centers, university campuses, or enterprise headquarters. Usually, these customers can satisfy their energy demand from a portfolio of sources, including the electric grid, local renewable sources, and on-site energy storage systems. Notable examples are thermal energy storage in Google data center in Taiwan [4], Tesla batteries to power Amazon data center in California [7], Google data center with on-site renewable sources in Belgium [5], and large-scale batteries in Apple Park's microgrid [3]. However, the electricity pricing for large customers is moving toward real-time pricing where price changes dynamically over time [31, 57, 61]. To counter this volatility, on-site storage systems are necessary, since they enable purchasing energy at low price periods. However, the management of such storage systems is challenging and can be captured via OLIM.

Summary of Contributions. In this paper, we develop online algorithms for OLIM with provably the best possible competitive ratios. More specifically, we propose two new algorithms: ARP[1]

[1] ARP is short for Adaptive Reservation Price.

(§3.1) and ARPRate (§3.3). ARP is the simpler of the two, and works for OLIM problems that have no rate constraints, i.e., no limit on input and output rate to/from the inventory at any slot. ARPRate is more complex, but works in a more general context, where the input and output rates are bounded.

The high-level intuition behind ARP is to store assets when prices are low and use the stored assets when the price rises. However, the fact that prices and demands are dynamic makes this challenging. The main ideas in the design of ARP are: (i) *adaptive reservation based on inventory utilization* that manages the challenges due to price dynamics; and (ii) the construction of *virtual inventory* that is used to tackle the challenges due to demand dynamics. The notion of virtual inventory is an algorithmic advance compared to previous work [21, 22]. In particular, given some back-up assets in the inventory, one can see satisfying the demand in each slot as the buying (minimization) version of an online k-search problem[2] [43] with the current demand as the target amount. However, the dynamic arrival of demands makes these online search problems coupled over time, and exacerbates the competitive analysis of the algorithms. Virtual inventory allows this to be managed. The full details are provided in §3.1 with analysis in §3.2.

ARPRate extends ARP to the case with rate constraints that limit the buying amount in each time slot to respect the limits of the inventory. In the energy storage management example, the rate constraints capture the charge/discharge limits of the batteries. Rate constraints add considerable complexity and have not been previously included in online optimization problems with inventory constraints. Two significant changes when generalizing from ARP to ARPRate are: (i) adaptive determination of the capacity of virtual inventories to reflect the output rate, and (ii) adaptive setting of reservation price to reflect the input constraints. The full details are provided in §3.3.

Our main technical results provide a competitive analysis and demonstrate that both ARP and ARPRate achieve the optimal competitive ratio of α as

$$\alpha = \left(W\left(-\frac{\theta - 1}{\theta \exp(1)} \right) + 1 \right)^{-1}, \tag{1}$$

where θ is the price fluctuation ratio, and $W(.)$ is Lambert-W function, defined as the inverse of $f(z) = z\exp(z)$. Interestingly, the above competitive ratios for ARP and ARPRate are exactly the same as the optimal competitive ratio for k-min search problem (see [43, Theorem 2]), when $k \to \infty$. However, OLIM is different since it comes with additional uncertainty about the demand as compared to [43] and includes an inventory management constraint that is not present in [43]. For illustrative purposes, Figure 1 depicts the value of the optimal competitive ratio α as a function of price fluctuation ratio θ, as compared to a sub-optimal competitive ratio of $\sqrt{\theta}$ achieved in [22] in a slightly different setting, and $\log\theta$ as a baseline, and it shows that its growth is less than the $\sqrt{\theta}$. However, series expansion of Lambert-W function shows that α as indicated in Equation (1) is $\Theta(\sqrt{\theta})$.

To illustrate the performance of ARP and ARPRate outside of the worst-case setting, we also perform a case study centered on our motivating example: energy storage management. More specifically, we evaluate our algorithms using data traces of electricity prices from several electricity markets (CAISO [19], NYISO [47], ERCOT [26], and German Electricity Market [30]), energy demands from multiple data centers of Akamai's CDN [46], and renewable energy production from solar [24] and wind installations [2, 6]. This case study shows that ARP and ARPRate provide significant improvements compared to state-of-the-art solutions. Specifically, in a broad set of representative scenarios that include different seasons and locations, we show that ARP achieves a

[2]In the minimization version of online k-search problem, a player must buy $k \geq 1$ units of an asset with the goal of minimizing the cost. In each slot $t = \{1, \ldots, T\}$, the player is presented a price $p(t)$, and must immediately decide whether or not to buy some units of the asset given $p(t)$. Details in [43].

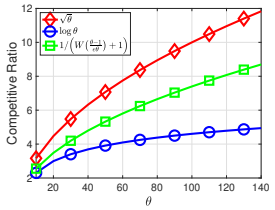


Fig. 1. An illustration of competitive ratio in Equation (11) as a function of price fluctuation ratio θ in comparison with $\sqrt{\theta}$ [22] and $\log\theta$.

cost reduction of 15% in comparison with using no energy storage at all, establishing the value of batteries for energy procurement. Further, ARP achieves an energy cost that is within 21–23% of the theoretically smallest achievable cost in an offline setting. In addition, ARP outperforms state-of-the-art algorithms for energy procurement by 7.5–10%.

2 PRELIMINARIES

In this section, we present the system model and formulate the problem. The interpretation of the model is illustrated using the cases study of storage-assisted energy procurement.

We assume that the time-slotted model in which the time horizon is divided into T slots, indexed by t, each with fixed length, e.g., 5 minutes in California ISO (CAISO) and New York ISO (NYISO) [1]. We consider the following scenario. At each slot, a demand $d(t) \geq 0$ arrives online that must be satisfied from either the market with the real-time price $p(t) \geq 0$ or from the local inventory, e.g., the energy storage system. We note that while the demand and price values are known for the current slot, those values are unknown for the future slots. Thus, the decision making has to be performed in an online fashion. The decision maker can purchase excess from the market in order to store in the inventory for future use. The goal is to design an algorithm to determine the value of $x(t)$, i.e., the procurement amount in each slot, so that the aggregate cost of purchases over the time horizon is minimized and the demand is satisfied.

2.1 Inventory Management Constraints

A key piece of the model is the constraints that define the evolution of the inventory. Let B, ρ_c, and ρ_d be the capacity, the maximum input rate, and the maximum output rate of the inventory. Let $b(t) \in [0, B]$ be the inventory level at the end of slot t that represents the amount of assets that are already in the inventory. The evolution of the inventory level is then given by

$$b(t) = b(t-1) + x(t) - d(t), \tag{2}$$

which states that the amount of assets in the inventory at the end of each slot is equal to the previous value $b(t-1)$ and the current procurement amount $x(t)$ subtracted by the demand $d(t)$. Further, we have two constraints on the value of $x(t)$:

$$x(t) \geq d(t) - \min\left\{\rho_d, b(t-1)\right\}, \tag{3}$$

which captures the maximum output rate from the inventory and ensures covering the demand. Specifically, the procurement amount should be greater than or equal to the current demand $d(t)$ subtracted by the maximum possible amount to use from the inventory, i.e., $\min\left\{\rho_d, b(t-1)\right\}$. Also, we have

$$x(t) \leq d(t) + \min\left\{\rho_c, B - b(t-1)\right\}, \tag{4}$$

which captures the input rate constraint to the inventory. Specifically, the procurement amount should be always less than the sum of current demand $d(t)$ and the maximum amount that could be stored in the storage, i.e., $\min\{\rho_c, B - b(t-1)\}$. Finally, we have the following inventory capacity constraint:

$$0 \le b(t) \le B.$$

In the example of energy storage, the rate constraints ρ_c and ρ_d are the charge and discharge rate of the energy storage systems. Several examples of actual values of ρ_c and ρ_d for different energy storage technologies, e.g., lead-acid and lithium-ion batteries, and compressed air energy storage are provided in §4.4.

2.2 Problem Formulation

We can now summarize the full formulation of online linear optimization with inventory management (OLIM). When the demands and market prices are known for the entire time horizon in advance, the *offline* version of OLIM can be formulated as a linear program as follows:

$$\begin{aligned}
\text{OLIM}: \quad \min \quad & \sum_{t\in\mathcal{T}} p(t)x(t) \\
\text{s.t.}, \quad & \forall t \in \mathcal{T}: \\
& x(t) \ge d(t) - \min\{\rho_d, b(t-1)\}, && (5)\\
& x(t) \le d(t) + \min\{\rho_c, B - b(t-1)\}, && (6)\\
& b(t) = b(t-1) + x(t) - d(t), && (7)\\
& 0 \le b(t) \le B, && (8)\\
\text{vars.}, \quad & x(t) \ge 0.
\end{aligned}$$

In this problem, the objective is to minimize the procurement cost from the market. Constraints (5)-(6) ensure covering the demand and rate limits. Constraint (7) dictates the evolution of the inventory, and (8) enforces the capacity of the inventory.

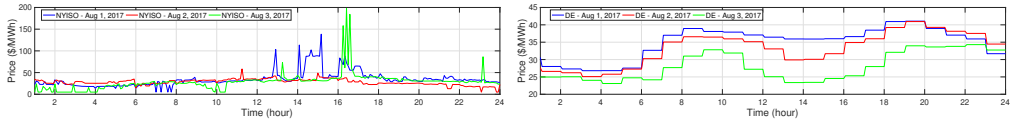
The OLIM problem is a linear program and can be solved efficiently in an offline manner. In this work, however, we are interested in developing *online algorithms* for OLIM that make decisions in each time t, knowing the past and current prices and demands, but not knowing those same inputs for the future. The algorithmic challenge is to procure assets from the market and store them in the inventory, without knowing if such decisions will work out favorably in the future.

Our approach for evaluating online algorithms is competitive analysis [17], where the goal is to design algorithms with the smallest *competitive ratio*, that is, the cost ratio between the online algorithm and an *offline* optimal algorithm that has access to *complete* input sequence. In our work, we devise online algorithms that have provably the best competitive ratio for OLIM.

In the analysis of our algorithms, we assume that the values of $p_{\max}$ and $p_{\min}$, the maximum and minimum prices, are known a priori by the algorithm. This assumption is reasonable because these values can be predicted using historical data. Further, it is known that obtaining positive results for problems like OLIM requires this assumption, e.g., [25, 43, 45]. Define $\theta = p_{\max}/p_{\min}$ as the price fluctuation ratio. This quantity appears in our main results.

2.3 Related Algorithmic Problems

OLIM is a generalization of online linear optimization, a topic that has received considerable attention over decades [13, 23, 39, 48, 59]. Beyond work on online linear optimization, OLIM is also related to several classical algorithmic problems, including one-way trading [25], the secretary problem [14], and the k-search problem [43]. In all of these problems, the goal is to search for the



(a) NYISO (b) German Electricity Market

Fig. 2. The energy price dynamics in New York ISO (NYISO) and German Electricity Market in three consecutive days in August 2017

optimum given the online arrival of cost/price functions over time. However, none of these other formulations include inventory management constraints, which add considerable difficulty.

Perhaps the most closely related problem to OLIM is the minimization variant of k-search, known as the k-min search problem [43]. In this problem, a player wants to buy $k \geq 1$ units of an asset with the goal of minimizing the cost. At any slot $t = \{1, \ldots, T\}$, the player is presented a price $p(t)$, and must immediately decide whether or not to buy some integer units of the asset given $p(t)$. By setting $d(t) = 0, t \in \{1, \ldots, T-1\}$ and $d(T) = B$ and allowing *fractional purchase*, OLIM degenerates to the k-min search problem. Hence, OLIM is an extension of the continuous version of k-min search problem. More insights about the connection between the above problems and OLIM are given in §5.

The other category of problems most related to OLIM is online linear programs with covering constraints [18]. The constraint (5) in OLIM is in form of a covering constraint, and constraints (6) and (8) could be represented as box constraints that limit the feasible values of the optimization variables. All three constraints could be captured in online linear programs with covering constraints. However, as compared to [18], the inventory management constraint (7) in OLIM couples the covering constraints across different time slots, which results in more challenging problem than existing online covering linear programs. Finally, while the existing solutions for online covering problems could be applied to OLIM, this paper develops specific online algorithms that achieve the best possible competitive ratio for OLIM.

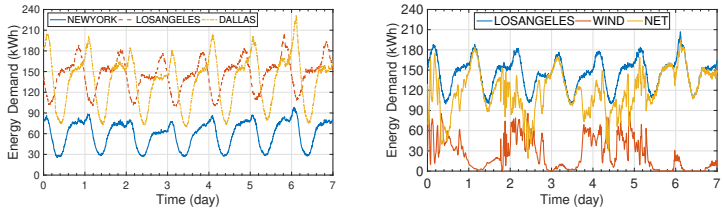
2.4 A Case Study: Energy Storage Management

Our motivation for studying OLIM comes from energy storage management. Energy storage management is a challenging issue faced by large-scale energy customers such as data centers, university campuses, and enterprise headquarters. In such scenarios, the demand and price values are highly uncertain and unpredictable and local energy storage is used to hedge the risk associated with fluctuations.

More specifically, electricity pricing for large customers like data centers is moving toward real-time pricing, and real-time prices can change dynamically over time [22, 33, 38, 54, 55]. Two examples of real-time energy prices in NYISO and German Electricity Market are demonstrated in Figure 2. By comparing the price dynamics in these two markets, we see that, while the prices in NYISO fluctuate dramatically without any regular pattern, in German Market there is a regular daily pattern with low price fluctuations.[3]

Not only prices fluctuate dramatically, the energy demand of large-scale energy consumers is highly unpredictable. For example, in data centers, the user demand for internet services is extremely variable, leading to high variability in the load of data centers. Further, the sophisticated optimization

[3]Note that, in practice there is another cost model for the electricity bill of data centers that includes a hybrid cost of volume and peak pricing [57, 61], however, we focus on volume pricing and leaves the peak pricing as future work.



(a) Different Cities in the United States (b) Net energy demand with wind

Fig. 3. Data center energy demand in different locations

algorithms used to improve the energy efficiency of data center's internal operations [27] can further increase the unpredictable variability of the energy demand.

In addition to price fluctuations and demand variability, many large-scale energy consumers are equipped with on-site renewable sources, e.g., Google's data center in Belgium [5]. The energy production level of renewable sources is uncertain and intermittent. For instance, energy production from solar panels can change in a matter of minutes due to cloud cover moving in to obscure the sun. This leads to an increased uncertainty in net energy demand of data center.

All these factors lead to the importance of energy storage management in large-scale customers such as data centers. However, the management of such systems is challenging.

3 OPTIMAL ONLINE ALGORITHMS

In this section, we present our main technical results. In §3.1, we propose ARP, an online algorithm for OLIM without rate constraints. Then, in §3.2, we prove that ARP achieves the optimal competitive ratio in this context. Based on the insights from the design of ARP, in §3.3, we propose ARPRate, which tackles the general OLIM problem with rate constraints. Finally, we prove that ARPRate achieves the optimal competitive ratio.

3.1 ARP: Adaptive Reservation Price

We begin by considering the case where the inventory has only capacity constraints, but no rate constraints, i.e., $\rho_c = \rho_d = B$. In this case we can rewrite constraint (5) as $x(t) \geq d(t) - b(t-1)$, and replace constraint (6) with $x(t) \leq d(t) + B - b(t-1)$.

3.1.1 The Design of ARP. The high-level goal of algorithms for OLIM is to store the asset when the market price is cheap and use the stored asset when the price is expensive. However, the dynamics of the prices and demand make this challenging. The main ideas that ARP uses to accomplish this are: (i) adaptive reservation price based on inventory utilization, which tackles the challenges due to price dynamics; and (ii) the construction of virtual inventories, which tackle the challenges due to demand dynamics. We discuss each of these in turn in the following.

Adaptive Reservation Price: ARP deals with price dynamics by defining a notion of *reservation price*. Having a properly constructed reservation price, ARP stores some amount of the asset if the current price is cheaper than the reservation price; otherwise, it releases the asset from the inventory. ARP adaptively determines the reservation price based on the available inventory space. Intuitively, once the inventory level is low, it is more eager to store assets; hence, it accepts higher prices. On the other hand, at high inventory utilization, it stores the asset only if the price is low. This is different from the fixed reservation design introduced in [22] that determines the reservation prices without considering the current inventory utilization.

Algorithm 1 The ARP algorithm for each $t \in \mathcal{T}$

1: // Initialization: at $t = 1$
2: $B_1 \leftarrow B$
3: $\upsilon \leftarrow 1$
4: $\xi_1 \leftarrow p_{\max}/\alpha$
// The main algorithm for t
5: **if** $d(t) > 0$ **then**
6: $\upsilon \leftarrow \upsilon + 1$
7: $B_\upsilon \leftarrow d(t)$
8: $\xi_\upsilon \leftarrow p_{\max}/\alpha$
9: **end if**
10: $x_i(t) \leftarrow [G_{B_i}(p(t)) - G_{B_i}(\xi_i)]^+, \quad \forall 1 \le i \le \upsilon$
11: $\xi_i \leftarrow \min\{p(t), \xi_i\}, \quad \forall 1 \le i \le \upsilon$
12: $x(t) \leftarrow \max\left\{\sum_{i=1}^{\upsilon} x_i(t), [d(t) - b(t-1)]^+\right\}$
13: $b(t) \leftarrow b(t-1) + x(t) - d(t)$
// Renew virtual inventories
14: **if** $b(t) = 0$ **then**
15: $\upsilon \leftarrow 1$
16: $\xi_1 \leftarrow p_{\max}/\alpha$
17: **end if**

Virtual Inventory: ARP deals with demand dynamics by defining the notion of virtual inventory. ARP views the demand in each time slot as an asset that must be purchased from the market with some degree of freedom obtained by shifting it using the inventory. To utilize this opportunity, ARP constructs several virtual inventories to record the satisfied amount of the demand from the market. Specifically, in each slot with $d(t) > 0$, ARP initiates a virtual inventory whose capacity is equal to the demand and is initially empty.

The Details of ARP. The full detail of ARP is provided via pseudocode in Algorithm 1. Note that we assume that the initial inventory level is zero, i.e., $b(1) = 0$ and, for notational convenience, we represent the physical inventory as the first virtual inventory, define $B_1 = B$ as its capacity (Line 2), and υ as the current number of virtual inventories given positive demand. (Line 6 indicates creating a new virtual inventory for the current slot and Line 15 renews the virtual inventories). Let B_i be the capacity of i-th virtual inventory and $B_\upsilon = d(t)$, i.e., we set the value of new virtual inventory to the current demand. Let ξ_i be the reservation price associated to the virtual inventory i with initial value of $p_{\max}/\alpha$, where $\alpha > 1$ is a parameter that is carefully chosen based on the competitive analysis and is defined in Equation (11). The insights behind this parameter are further discussed in the competitive analysis of ARP.

The cornerstone of ARP is in Line 10, where the procurement amount for each virtual inventory, i.e., $x_i(t)$, is set. ARP defines $G_{B_i}(\xi_i)$ as the reservation function to determine the amount of asset to be stored in the i-th virtual inventory in each slot. This function represents the target amount of stored assets in the i-th virtual inventory when the reservation price is ξ_i. In slot t, ARP stores an additional amount of $G_{B_i}(p(t)) - G_{B_i}(\xi_i)$ into virtual inventory i if the current price, $p(t)$, is less than ξ_i; otherwise, it stores nothing. Both situations can be stated in the following compact form

$$x_i(t) = \left[G_{B_i}(p(t)) - G_{B_i}(\xi_i)\right]^+. \tag{9}$$

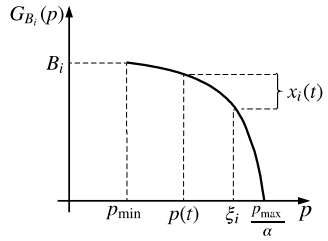


Fig. 4. An illustration of function $G_{B_i}(p)$ and determining $x_i(t)$ as the procurement amount for virtual inventory i

In this way, ARP logically allocates $x_i(t)$ units of asset to inventory i (Line 10), and the reservation price will be updated to $\min\{\xi_i, p(t)\}$ (Line 11). Note that in ARP, ξ_i records the minimum seen price during the lifetime of i-th virtual inventory.

ARP designs function $G_{B_i}(p)$ so that it achieves the optimal competitive ratio. To accomplish this, we choose the following function:

$$G_{B_i}(p) = \alpha B_i \ln\left[\left(1 - \frac{p}{p_{\max}}\right)\frac{\alpha}{\alpha - 1}\right],\ p \in \left[p_{\min}, \frac{p_{\max}}{\alpha}\right], \tag{10}$$

where

$$\alpha = \left(W\left(-\frac{\theta - 1}{\theta \exp(1)}\right) + 1\right)^{-1}, \tag{11}$$

and W denotes *Lembert-W function* defined as inverse of $f(z) = z\exp(z)$, and $\theta = p_{\max}/p_{\min}$ is the price fluctuation ratio. Note that α is achieved as the unique solution for

$$\frac{1 - 1/\theta}{1 - 1/\alpha} = \exp\left(\frac{1}{\alpha}\right). \tag{12}$$

Note that function $G_{B_i}(p)$ is carefully designed to guarantee a worst-case competitive ratio and is based on the threshold function defined in [43] for the k-min search problem. More details and intuition are given in [43, Lemma 3].

Figure 4 depicts function $G_{B_i}(p) \in [0, B_i]$, and $p \in [p_{\min}, p_{\max}/\alpha]$ as a decreasing function. It also demonstrates how to determine the reservation amount for virtual inventory i. Further, when the price p is larger than or equal to $p_{\max}/\alpha$, $G_{B_i}(p) = 0$. When the price is equal to $p_{\min}$, the reservation amount is equal to $\alpha B_i \ln\left[\left(1 - \frac{1}{\theta}\right)\frac{\alpha}{\alpha - 1}\right]$. By substituting the value of α from Equation (12), we have $G_{B_i}(p_{\min}) = B_i$, which means when the price is minimum, ARP stores the full capacity of the inventory.

The last step is to determine the aggregate procurement quantity $x(t)$ (Line 12). To satisfy the demand constraint, i.e., $x(t) \geq d(t) - b(t-1)$, we calculate $x(t)$ as follows

$$x(t) = \max\left\{\sum_{i=1}^{v} x_i(t), d(t) - b(t-1)\right\}. \tag{13}$$

When $\sum_{i=1}^{v} x_i(t) < d(t) - b(t)$, ARP discharges the physical inventory completely to meet the demand, hence, we have $b(t) = 0$. In this case, we renew all virtual inventories to the initial state of having only the physical inventory (Line 15). The intuition is that with fully discharging the physical inventory, we exhausted the capability of shifting the demand using the inventory, hence we renew the process.

3.2 Analysis of ARP

This section proves our main result for ARP as stated in Theorem 1.

Theorem 1. *With the following reservation function*

$$G_{B_i}(p) = \alpha B_i \ln\left[\left(1 - \frac{p}{p_{\max}}\right)\frac{\alpha}{\alpha - 1}\right],\ p \in \left[p_{\min}, \frac{p_{\max}}{\alpha}\right],$$

ARP *achieves the optimal competitive ratio of* α *as*

$$\alpha = \left(W\left(-\frac{\theta - 1}{\theta \exp(1)}\right) + 1\right)^{-1}.$$

Before the main proof on competitiveness of the algorithm, we start our analysis by showing that ARP is a valid algorithm for OLIM, i.e., that ARP produces a feasible solution.

Theorem 2. ARP *generates a feasible solution to* OLIM.

The proof that ARP covers the demand is straightforward because of Equation (13). The range of function $G_{B_i}(p)$ along with the renewal process in Line 15 guarantees that the capacity constraint of physical inventory is respected. The proof is formally given in Appendix A.

In the following, we proceed to prove the competitiveness result in Theorem 1. After some preliminaries (§3.2.1), we characterize an upper bound on the cost of ARP (Lemma 1). Then we construct a lower bound on the offline optimum (Lemma 2). Finally, we prove the theorem by comparing these two values. We end by showing the optimality of the competitive ratio in §3.2.3, which simply uses the fact that OLIM is an extended version of the basic k-min search problem and since it achieves a competitive ratio that is the same as the optimal competitive ratio for k-min search problem, it is optimal for the extended problem as well.

3.2.1 Preliminaries. We begin with a few definitions. Note that for notational brevity, we slightly abuse the notations by dropping index t in the competitiveness analysis in the rest of this section.

Definition 1. *Define* $\omega \in \Omega$ *as an input instance to* OLIM *including the price and demand, i.e.,*

$$\omega \stackrel{def}{=} [\omega(t) = \langle p(t), d(t)\rangle]_{t\in\mathcal{T}}.$$

Moreover, $cost_\omega(\mathsf{ARP})$ *is the cost of* ARP *under instance* ω *and* $cost_\omega(\mathsf{OPT})$ *is the offline optimal cost under* ω*. We drop subscript* ω *from the costs when we are not focusing on a particular* ω*.*

Definition 2. ARP *is* α*-competitive, if for any* $\omega \in \Omega$

$$cost_\omega(\mathsf{ARP}) \le \alpha \cdot cost_\omega(\mathsf{OPT}) + cons, \tag{14}$$

where $cons \ge 0$ *is a constant number.*

Definition 3. *Reservation and idle periods. The time horizon* T *can be divided into two type of periods: the reservation period, which contains the interval between beginning to charge and fully discharging of the inventory; and the idle period, which corresponds to the interval that lies between two adjacent reservation periods.*

Additionally, in the proof we use the following additional notations. Consider an instance in which ARP results in n reservation periods. During the i-th reservation period, $i \le n$, we assume there are $\hat{v}_i$ virtual inventory units created. Let $\hat{\xi}_{i,j}$ be the final reservation price of the j-th virtual inventory during the i-th reservation period. Let $B_{i,j}$ be the capacity of the j-th inventory and $\hat{b}_{i,j}$ be the final inventory level correspondingly. Obviously, we have $\hat{b}_{i,j} = G_{B_{i,j}}(\hat{\xi}_{i,j})$. Let D be the

total demand during the time horizon, i.e., $D = \sum_{t \in \mathcal{T}} d(t)$, and $\hat{b}$ be the final inventory level of the physical inventory, i.e., $\hat{b} = b(T)$.

In addition, let $F_i(\beta)$ be the minimum cost of purchasing β units of asset during i-th reservation period, and $\tilde{p}$ be the minimum price during the idle periods. Finally, for better illustration of several parts in the proof, we need the inverse of function G_{B_i} is defined as

$$G_{B_i}^{-1}(b) = p_{\max}\left[1 - \left(1 - \frac{1}{\alpha}\right)\exp\left(\frac{b}{\alpha B_i}\right)\right],\ b \in [0, B_i]. \tag{15}$$

3.2.2 Proof of Theorem 1. The proofs of all lemmas in this section are given in Appendix B. We begin by characterizing an upper bound on the cost of ARP.

Lemma 1. *cost(*ARP*) is upper bounded by*

$$cost(\mathsf{ARP}) \le \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db + \left(D - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j} + \hat{b}\right)p_{\max}.$$

The key idea of the proof is to explicitly characterize the worst-case cost of the online algorithm using the threshold function in Equation (10). Specifically, in above equation, the first term is an upper bound on the cost over n reservation periods each with $\hat{v}_i, 0 \le i \le n$, virtual inventories, and the second term is an upper bound on the cost over the idle period. The details are given in Appendix B.

The next lemma provides a lower bound on the cost of offline optimal solution.

Lemma 2. *Assume* ARP *has n reservation period, let β_i be the asset purchased by the optimal offline solution during the i-th reservation period, and let $\tilde{p}$ be the minimum price during the idle periods. cost(*OPT*) is lower bounded by*

$$cost(\mathsf{OPT}) \ge \sum_{i=1}^{n} F_i(\beta_i) + \left(D - \sum_{i=1}^{n}\beta_i\right)\tilde{p}.$$

The proof is given in Appendix B.2.

Using the previous lemmas, we now proceed to prove the competitive ratio. First, we consider the simple case, where $D = 0$. In this trivial case, cost(OPT) = 0, that of ARP is at most $Bp_{\max}$. Obviously, ARP is α-competitive since it satisfies the definition of competitive ratio in Equation (14) by setting cons = $Bp_{\max}$.

Second, we focus on the more general case in which $D > 0$. If the minimum price during the time horizon is larger than or equal to $p_{\max}/\alpha$, the cost of ARP is at most $p_{\max}(B + D)$ and that of OPT is lower bounded by $\frac{p_{\max}}{\alpha}D$. It is easy to see that ARP is α-competitive according to the definition. We only consider the case where the minimum price during the time horizon is less than $p_{\max}/\alpha$, and obviously, the minimum price occurs in the reservation period. We have $\sum_{i=1}^{n} F_i(\beta_i) > 0$. Using the

results in Lemmas 1 and 2 we have

$$\begin{aligned}\frac{\text{cost(ARP)} - \hat{b}p_{\max}}{\text{cost(OPT)}} \leq & \frac{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G^{-1}_{B_{i,j}}(b)db + \left(D - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)\cdot p_{\max}}{\sum_{i=1}^{n} F_i(\beta_i) + \left(D - \sum_{i=1}^{n}\beta_i\right)\cdot \tilde{p}} \\ = & \frac{Q + \left(\sum_{i=1}^{n}\beta_i - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max} + \left(D - \sum_{i=1}^{n}\beta_i\right)\cdot p_{\max}}{\sum_{i=1}^{n} F_i(\beta_i) + \left(D - \sum_{i=1}^{n}\beta_i\right)\cdot \tilde{p}} \\ \leq & \max\left\{\frac{Q + \left(\sum_{i=1}^{n}\beta_i - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}}{\sum_{i=1}^{n} F_i(\beta_i)}, \frac{\left(D - \sum_{i=1}^{n}\beta_i\right)\cdot p_{\max}}{\left(D - \sum_{i=1}^{n}\beta_i\right)\cdot \tilde{p}}\right\} \\ \leq & \max\left\{\frac{Q + \left(\sum_{i=1}^{n}\beta_i - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)\cdot p_{\max}}{\sum_{i=1}^{n} F_i(\beta_i)}, \alpha\right\}.\end{aligned} \tag{16}$$

where

$$Q = \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G^{-1}_{B_{i,j}}(b)db.$$

In Equation (16), the first inequality is by Lemma 1 and 2, the second inequality is by $D - \sum_{i=1}^{n}\beta_i \geq 0$ and the third inequality is by $p_{\max}/\tilde{p} \leq \alpha$.

The following two lemmas provide an upper bound for Equation (16), which is the main step in proving the competitive ratio.

Lemma 3. *Given $G^{-1}_{B_{i,j}}(\hat{b}_{i,j}), i \in \{0, 1, \ldots, n\}$ in Equation (15), we have*

$$\frac{\int_0^{\hat{b}_{i,j}} G^{-1}_{B_{i,j}}(b)db + (B_{i,j} - \hat{b}_{i,j})p_{\max}}{\hat{\xi}_{i,j}B_{i,j}} = \alpha, \ \forall\ \hat{b}_{i,j} \in [0, B_{i,j}]. \tag{17}$$

Proof. By substituting Equation (15), we first calculate the second term in numerator of Equation (17) as follows

$$\begin{aligned}\int_{b=0}^{\hat{b}_{i,j}} G^{-1}_{B_{i,j}}(b)db = & p_{\max}\left[b - \left(1 - \frac{1}{\alpha}\right)\exp\left(\frac{b}{\alpha B_{i,j}}\right)\alpha B_{i,j}\right]\Bigg|_{b=0}^{\hat{b}_{i,j}} \\ = & p_{\max}\left[\hat{b}_{i,j} - \left(1 - \frac{1}{\alpha}\right)\exp\left(\frac{\hat{b}_{i,j}}{\alpha B_{i,j}}\right)\alpha B_{i,j}\right] \\ & + p_{\max}\left(1 - \frac{1}{\alpha}\right)\alpha B_{i,j}.\end{aligned}$$

Then, we calculate the numerator

$$\begin{aligned}(B_{i,j}-\hat{b}_{i,j})p_{\max}+\int_{b=0}^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db &= \alpha B_{i,j}\left(p_{\max}\left[1-\left(1-\frac{1}{\alpha}\right)\exp\left(\frac{\hat{b}_{i,j}}{\alpha B_{i,j}}\right)\right]\right)\\ &= \alpha B_{i,j}G_{B_{i,j}}^{-1}(\hat{b}_{i,j}). \qquad (18)\end{aligned}$$

Substituting (18) into (17) completes the proof. □

Using the result in Lemma 3, we have the following result.

Lemma 4.

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db+\left(\sum_{i=1}^{n}\beta_i-\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}}{\sum_{i=1}^{n}F_i(\beta_i)}\le\alpha.$$

Proof. We prove the result in Lemma 4 by contradiction. Assume

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db+\left(\sum_{i=1}^{n}\beta_i-\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}}{\sum_{i=1}^{n}F_i(\beta_i)}>\alpha.$$

Then, we have

$$\begin{aligned}&\frac{Q+\left(\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}B_{i,j}-\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}}{\sum_{i=1}^{n}F_i\left(\sum_{j=1}^{\hat{v}_i}B_{i,j}\right)}\\ =&\frac{Q+\left(\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\beta_{i,j}-\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}+\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}(B_{i,j}-\beta_{i,j})p_{\max}}{\sum_{i=1}^{n}F_i\left(\sum_{j=1}^{\hat{v}_i}\beta_{i,j}\right)+\sum_{i=1}^{n}\left[F_i\left(\sum_{j=1}^{\hat{v}_i}B_{i,j}\right)-F_i\left(\sum_{j=1}^{\hat{v}_i}\beta_{i,j}\right)\right]}\\ \ge&\frac{Q+\left(\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\beta_{i,j}-\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}+\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}(B_{i,j}-\beta_{i,j})p_{\max}}{\sum_{i=1}^{n}F_i\left(\sum_{j=1}^{\hat{v}_i}\beta_{i,j}\right)+\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}(B_{i,j}-\beta_{i,j})\frac{p_{\max}}{\alpha}}\\ >&\alpha.\end{aligned}$$

In the aforementioned result, we use the fact that there is a worst-case input instance such that for all $0<\beta<\beta'<\sum_{j=1}^{\hat{v}_i}B_{i,j}-\hat{b}_i$, we have $F_i(\beta')-F_i(\beta)\le\frac{p_{\max}}{\alpha}(\beta'-\beta)$, where $\hat{b}_i$ is the initial state of the inventory of the offline algorithm in the i-th reservation period . The formal statement of this fact is stated in Lemma 6 in Appendix B.3.

During the lifetime of the j-th virtual inventory of the i-th reservation period, the minimum market price is $\hat{\xi}_{i,j}$. When the procurement amount during the i-th reservation period is $\sum_{j=1}^{\hat{v}_i}B_{i,j}$, the cost of the optimal algorithm satisfies

$$\sum_{i=1}^{n}F_i\left(\sum_{j=1}^{\hat{v}_i}B_{i,j}\right)\ge\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{\xi}_{i,j}B_{i,j}.$$

Thus, we have

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db + \left(\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} B_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}}{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{\xi}_{i,j}B_{i,j}}$$
$$=\frac{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\left[\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db + \left(B_{i,j}-\hat{b}_{i,j}\right)p_{\max}\right]}{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{\xi}_{i,j}B_{i,j}} > \alpha.$$

That means that there is at least a pair of i and j such that

$$\frac{\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db + (B_{i,j}-\hat{b}_{i,j})p_{\max}}{\hat{\xi}_{i,j}B_{i,j}} > \alpha.$$

The above equation contradicts the results in Lemma 3. This completes the proof. □

Using the result in Lemma 4 and Equation (16), we have

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db + \left(D - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j}\right)p_{\max}}{\sum_{i=1}^{n} F_i(\beta_i) + \left(D-\sum_{i=1}^{n}\beta_i\right)\cdot\tilde{p}} \le \alpha.$$

Thus, we have

$$\text{cost}(\texttt{ARP}) \le \alpha\cdot\text{cost}(\texttt{OPT}) + \hat{b}p_{\max} \le \alpha\cdot\text{cost}(\texttt{OPT}) + Bp_{\max},$$

where $Bp_{\max}$ is a constant. And the proof of Theorem 1 is complete.

3.2.3 The Optimality of the Competitive Ratio. Finally, we conclude this section by arguing that the competitive ratio proven in the previous subsection is optimal among online algorithms. To see this, set $d(t) = 0, t \in \{1,\ldots,T-1\}$ and $d(T) = B$. In this case, OLIM degenerates into the k-min search problem, whose optimal competitive ratio [43, Theorem 2] is exactly equal to that of ARP. Thus α is a lower bound for OLIM, since it is a generalization of the k-min search problem.

3.3 ARPRate: Incorporating Rate Constraints

In this section, we build on the design of ARP and develop ARPRate, which can additionally handle input and output rate constraints. Algorithm 2 provides the pseudocode of ARPRate. While the general flow of the algorithms are similar, ARPRate includes two new ideas.

First, ARPRate intelligently sets the capacity of the virtual inventories to respect the output rate constraints. The high-level intuition for doing this is that the output (discharge) rate constraint limits the capability of using the inventory in each slot. Hence, it may not be possible to fully satisfy the demand by discharging the inventory. So, creating a virtual inventory with capacity equal to the demand does not make sense. This is done by calling sub-procedure InitVS in Line 8 of ARPRate, with details explained in §3.3.1.

Second, ARPRate intelligently sets the value of the reservation prices to respect the input (charge) rate constraints. The high-level intuition for how to do this is that the input rate constraint limits the amount of stored asset in each slot. Hence, if the price is very cheap, ARP might propose to store some amount that is beyond the capability of inventory to input. Instead, ARPRate updates

Algorithm 2 The ARPRate algorithm for each $t \in \mathcal{T}$

1: // Initialization: at $t = 1$
2: $B_1 \leftarrow B$
3: $v \leftarrow 1$
4: $\xi_1 \leftarrow p_{\max}/\alpha$
// The main algorithm for t
5: **if** $d(t) > 0$ **then**
6: $v \leftarrow v + 1$
7: $\xi_v \leftarrow p_{\max}/\alpha$
8: $B_v \leftarrow \texttt{InitVS}\,(p(t), d(t), \rho_d, v, \{\xi_i, B_i\}_{i=1:v-1}, \varepsilon_1)$
9: **end if**
10: $\hat{x}(t) \leftarrow \sum_{i=1}^{v} \left[G_{B_i}(p(t)) - G_{B_i}(\xi_i)\right]^+$
11: $x(t) \leftarrow \hat{x}(t)$
12: $p \leftarrow p(t)$
// Check active output rate constraint
13: **if** $\hat{x}(t) < [d(t) - \min\{b(t-1), \rho_d\}]^+$ **then**
14: $x(t) \leftarrow [d(t) - \min\{b(t-1), \rho_d\}]^+$
15: **end if**
// Check active input rate constraint
16: **if** $\hat{x}(t) > \rho_c + d(t)$ **then**
17: $x(t) \leftarrow \rho_c + d(t)$
18: $p \leftarrow \texttt{CalRP}\,(d(t), \rho_c, v, \{\xi_i, B_i\}_{i=1:v}, \varepsilon_2)$
19: **end if**
20: $b(t) \leftarrow b(t-1) + x(t) - d(t)$
21: $\xi_i \leftarrow \min\{p, \xi_i\}, \quad \forall 1 \le i \le v$
// Renew virtual inventories
22: **if** $b(t) = 0$ **then**
23: $v \leftarrow 1$
24: $\xi_1 \leftarrow p_{\max}/\alpha$
25: **end if**

Algorithm 3 $\texttt{InitVS}\,(p(t), d(t), \rho_d, v, \{\xi_i, B_i\}_{i=1:v-1}, \varepsilon_1)$

1: $B_v \leftarrow 0,\ B'_v \leftarrow d(t)$
2: **while** $|B'_v - B_v| > \varepsilon_1$ **do**
3: $B_v \leftarrow B'_v$
4: $\hat{x}(t) \leftarrow \sum_{i=1}^{v} \left[G_{B_i}(p(t)) - G_{B_i}(\xi_i)\right]^+$
5: $B'_v \leftarrow d(t) - [d(t) - \rho_d - \hat{x}(t)]^+$
6: **end while**
7: **Output**: B_v

the reservation price based on the capability to store, i.e., input rate constraint. This is done by calling the sub-procedure CalRP in Line 18 of ARPRate, with details presented in §3.3.2.

Algorithm 4 CalRP $(d(t), \rho_c, v, \{\xi_i, B_i\}_{i=1:v}, \varepsilon_2)$

$p \leftarrow p_{\min}, p' \leftarrow p_{\max}/\alpha$;
while $|p' - p| > \varepsilon_2$ **do**
  $z \leftarrow \sum_{i=1}^{v} \left[G_{B_i}((p+p')/2) - G_{B_i}(\xi_i)\right]^+$
  if $z > \rho_c + d(t)$ **then** $p \leftarrow (p+p')/2$ **else** $p' \leftarrow (p+p')/2$
end while
Output: p

3.3.1 Initializing a New Virtual Inventory. At t, the *preferred procurement amount*, denoted as $\hat{x}(t)$, should be calculated as

$$\hat{x}(t) = \sum_{i=1}^{v} \left[G_{B_i}(p(t)) - G_{B_i}(\xi_i)\right]^+, \tag{19}$$

where B_v is the new capacity of virtual inventory. Instead of the way that ARP sets the capacity to $d(t)$, ARPRate subtracts $[d(t) - \rho_d - \hat{x}(t)]^+$, i.e., the additional amount due to output rate constraint, from the capacity. Hence,

$$B_v = d(t) - [d(t) - \rho_d - \hat{x}(t)]^+. \tag{20}$$

Equations (19) and (20) show that B_v and $\hat{x}(t)$ are dependent on each other. To address this, we devise the sub-procedure InitVS (Algorithm 3) to calculate the capacity of the new virtual inventory. InitVS captures this dependency and updates determining B_v and $x(t)$ using a simple search algorithm in iterative manner with parameter ε_1 as the stopping criteria (ε_1 can be arbitrarily close to 0). In Appendix C, we prove that InitVS converges to a solution of Equations (19) and (20). Lines 11-19 of ARPRate calculates $x(t)$ by taking into account rate constraints in the following cases:

(1) *Inactive output and input rate, Line* 11. In this case, $[d(t) - \min\{b(t-1), \rho_d\}]^+ \le \hat{x}(t) \le \rho_c + d(t)$, hence, $x(t) = \hat{x}(t)$.

(2) *Active output rate constraint, Line* 13. In this case, $\hat{x}(t)$ fails to satisfy the demand, so we set the actual procurement amount $x(t) = [d(t) - \min\{b(t-1), \rho_d\}]^+$.

(3) *Active input rate constraint, Line* 19. In this case, $\hat{x}(t)$ will be beyond the input rate of inventory, hence $x(t) = \rho_c + d(t)$.

3.3.2 Calculating the Reservation Price. The final step is to update the reservation price ξ_i for each virtual inventory. For cases (1) and (2), the reservation price for each virtual inventory is updated similar to that of ARP, i.e., $\min\{\xi_i, p(t)\}$ (Lines 12 and 21). For case (3), ξ_i is updated as follows. Let p be the updated reservation price, whose value is the solution to the following equation

$$\sum_{i=1}^{v_t} \left[G_{B_i}(p) - G_{B_i}(\xi_i)\right]^+ = \rho_c + d(t).$$

ARPRate solves the above equation for p by calling the sub-procedure CalRP (Algorithm 4) in iterative manner, with parameter ε_2 as the stopping criteria (ε_2 can be arbitrarily close to 0).

3.3.3 Competitive Analysis of ARPRate. We end this section by outlining the the following Theorem on the competitiveness of ARP.

Theorem 3. ARPRate *achieves the optimal competitive ratio of* α *as in Equation* (11).

The full details are given in Appendix D, and here we simply give an overview of the main steps in the proof. Much of the argument proceeds similarly to that for ARP. In particular, the lower bounds are the same. The key new piece is to prove an upper bound for ARPRate similar to that

of in Lemma 1. To do this, we first show that, in worst case, the output rate is not active. This simplifies the analysis and means that the upper bound is only impacted by input rate constraints. More specifically, we prove the following upper bound in the appendix. Denote $x(t)$ as the amount of reserved asset by the online algorithm at time slot t. Also, we define $\mathcal{T}_r \subset \mathcal{T}$ be the set of time slots at which the input rate truncates the procurement amount. Last, $p'(t)$ is the actual reservation price computed in Algorithm 4, and obviously, $p'(t) > p(t)$. Let $\mu(t) = p'(t) - p(t)$ denotes the difference between $p'(t)$ and $p(t)$.

Lemma 5. *The cost of* ARPRate *is upper bounded by*

$$cost(\texttt{ARPRate}) \leq \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\int_0^{\hat{b}_{i,j}} G^{-1}_{B_{i,j}}(b)db + \left(D - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i}\hat{b}_{i,j} + \hat{b}\right)\cdot p_{\max} - \sum_{t\in\mathcal{T}_r}\mu(t)x(t).$$

The proof is given in Appendix D.

4 A CASE STUDY

To illustrate the performance improvements that result from the optimal online algorithms developed in this paper, we end with a case study illustrating the use of ARP and ARPRate in the context of energy storage management by a data center. These results highlight the average-case performance in a real-world scenario, as opposed to the worst-case performance results in Theorems 1 and 3. Our results address the following questions:

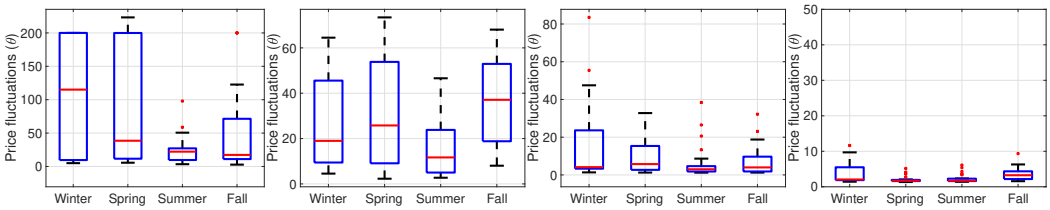
(i) *How does the empirical cost ratio of* ARP *compare to the theoretical competitive ratio?* We find that ARP achieves a significantly smaller average cost ratio than the worst-case competitive ratio guarantee provided by our theoretical analysis (Observation 1).
(ii) *How does* ARP *compare to existing algorithms?* We find that ARP outperforms all the baseline and existing algorithms [22, 33, 54] by 7%-15%, on average (Observation 2).
(iii) *How sensitive is* ARP *to various parameters such as the penetration of renewable energy?* Our experiments demonstrate that ARP is minimally affected by these parameters as compared to substantial performance fluctuations in alternative algorithms (Observations 4).
(iv) *How does* ARPRate *compare to* ARP*?* We find that the empirical cost ratio of ARPRate is slightly better than ARP when rate constraints are tight (Observation 6).

4.1 Experimental Setup

In this subsection, we explain the data traces for data center energy demand, energy prices, and renewable generation used in the experiments.

Data Center Energy Demand: We use a repository of demand traces from Akamai's server clusters in several data centers collected during a 31 day period from multiple locations around the world. The data includes the server load information from 973 data centers in 102 countries, collected every 5 minutes. To calculate energy consumption as a function of load, we use the standard linear model [15]. Let d_{idle} and d_{peak} be the energy consumptions by an idle and a fully utilized server, respectively. Then, the energy (in kWh) consumed by a server serving normalized load $l \in [0, 1]$ is $d(l) = d_{\text{idle}} + (d_{\text{peak}} - d_{\text{idle}}) \times l$. In our experiments, we use $d_{\text{idle}} = 100$kWh, and $d_{\text{peak}} = 250$kWh, representing energy proportionality factor, i.e., defined as $(d_{\text{peak}} - d_{\text{idle}})/d_{\text{peak}}$, of 0.6 [49]. We report the results of different algorithms for a selection of data centers in the four different cities: Los Angeles, New York, Dallas, and Frankfurt. A representative 7-days snapshot of the energy consumption is depicted in Figure 3.

Energy Prices: We use the electricity prices from a local electricity market for each data center location, i.e., CAISO [19] for Los Angeles, NYISO [47] for New York, ERCOT [26] for Dallas, and German Electricity Market (abbreviated as DE in results) for Frankfurt. Note that FERC is forcing the



(a) CAISO, high fluctuation (b) NYISO, high fluctuation (c) ERCOT, medium fluctuation (d) DE, low fluctuation

Fig. 5. Price fluctuations in different seasons/locations

U.S. electricity markets to transition to real-time markets with 5-minutes settlement intervals [1]. Currently CAISO and NYISO adapt this policy, and the rest are in the middle of this transition. To have a common settlement interval for all different markets, we set the length of each slot to 5 minutes, and for those that the current real-time market comes with different length (ERCOT with 15 minutes and DE with 1 hour intervals), we down-sample the market price readings to 5 minutes.

Recall that the performance of our algorithm is a function of parameter θ (see Equation (11)) as the price fluctuation ratio. Note that different markets exhibit different price fluctuations in different seasons. In Figure 5, the box plots for different markets in different seasons are shown. The results show that the fluctuations in spot prices in CAISO (Figure 5(a)) and NYISO (Figure 5(b)) are high, in ERCOT it is medium (Figure 5(c)), and in DE it is low (Figure 5(d)). Consequently, to have a comprehensive experimental study in different fluctuation patterns, we compare the performance of different algorithms in different markets and different seasons.

Renewable Data Traces: We evaluate the results of different algorithms in three different scenarios: (i) without any on-site renewable supply, (ii) with 50% penetration on-site wind generation; and (iii) with 50% penetration on-site solar generation. Note that with local renewable supply, the net energy demand, i.e., the total demand subtracted by the local renewable supply, must be procured from the grid with real-time pricing. Since the renewable supply is uncertain, the net demand in cases (ii) and (iii) is expected to be more uncertain (as depicted in Figure 3(b)).

We use the solar data from PVWatts [24] and obtain the hourly solar radiation in different seasons. We match each data center with solar readings from a location as close to it as possible. The exact distance from data center to the location from where the readings were obtained is shown in Table 1. We scale the values so that 50% of the total demand is satisfied by solar panels. We set the parameters according to the default values [24, Table 2]. While the spot prices and energy demand readings are 5-minutes, the solar data is hourly. Hence, we make an assumption that the solar data is almost constant during each hour and use the hourly values for each 5 minute slots. The wind traces for the U.S. locations are obtained from [2], and for European location are obtained from OPSD [6]. A summary of locations, markets, and distances to renewables is listed in Table 1.

4.2 Comparison Algorithms and Settings

We implemented our algorithms and several other state-of-the-art algorithms for comparison as described below (see Table 2).

▷ (OPT) Optimal offline algorithm with storage by solving OLIM in §2. Since OPT represents the best achievable cost for the given inputs, all other algorithms are evaluated by computing *empirical cost ratio* which is the ratio of the cost of the algorithm with the cost of OPT. The cost ratio is always greater than equal to 1 and lower the cost ratio of an algorithm, the better the performance.

Table 1. Summary of data center locations, markets, and nearby solar and wind power plants

City	Market	Distance from solar	Distance from wind
Los Angeles	CAISO	80 mi.	48 mi.
New York	NYISO	37 mi.	52 mi.
Dallas	ERCOT	63 mi.	145 mi.
Frankfurt	DE	35 mi.	-

Table 2. Summary of algorithms that are evaluated

Our proposed online algorithms	
ARP	Basic online algorithm (§3.1)
ARPRate	Online algorithm with rate constraints (§3.3)
PreDay	A simple data-driven approach that uses the optimal solution for the previous day for the current day
Other algorithms for comparison	
OPT	Optimal offline solution with storage
NoSTR	Optimal offline solution without storage
OnFix [22]	State-of-the-art online algorithm with fixed threshold price
LypOpt [33]	State-of-the-art online Lyapunov-based algorithm

▷ (NoSTR) A baseline scheme that simply satisfies the net energy demand from the grid, assuming no storage is available. The cost ratio of NoSTR quantifies the maximum benefit of having storage.

▷ (PreDay) Our data-driven approach that uses the optimal derived for the previous day for the current day by projecting into a feasible range (satisfying capacity, demand, and rate constraints). PreDay is representative of a statistical approach that uses historical statistics to inform future decisions.

▷ (OnFix) Existing sub-optimal online algorithm [22] is a simple strategy that uses a fixed threshold of $p = \sqrt{p_{\max} \times p_{\min}}$ as the purchasing threshold and fully charges the storage if the current price is lower than p, otherwise, discharges the storage as much as possible and purchases the remaining amount.

▷ (LypOpt) Lyapunov-based approach [33, 54] that uses Lyapunov optimization to solve OLIM. Note that [33] considers load-balancing among multiple data centers as well, and to have a fair comparison, we focus on single data center model in [33], and with this reduction both algorithms in [33, 54] become similar.

Unless otherwise mentioned, we set the length of each slot to 5 minutes, according to FERC rules. The time horizon is 1 day; hence $T = 12 \times 24 = 288$. We set the time horizon to 1 day to potentially see the impact of daily patterns in PreDay. The capacity of energy storage is set to $C = 18 \times \max_{t \in \mathcal{T}} d(t)$, sufficient to power the data center for 1.5 hours at max net demand. In experiments with renewable, the penetration level for solar and wind is 50%. Finally, each data point in figures and tables corresponds to the average results of 30 runs (days) over a month, each with the corresponding demand, renewable generation, and market prices.

4.3 Results for ARP

In Table 3, the empirical cost ratio of 5 algorithms (NoSTR, PreDay, LypOpt, OnFix, and ARP) are reported across a broad set of settings: (i) four different locations; (ii) four different seasons; (iii) and with/without renewable. We report the following notable observations.

Table 3. The empirical cost ratio of different algorithms in different markets and different seasons

	Market	θ	α	Cost ratio for no renewables					Cost ratio for 50% wind					Cost ratio for 50% solar				
				NoSTR	PreDay	LypOpt	OnFix	ARP	NoSTR	PreDay	LypOpt	OnFix	ARP	NoSTR	PreDay	LypOpt	OnFix	ARP
Winter	**CAISO**	110.00	7.74	1.88	1.60	1.79	1.53	1.44	2.06	2.13	1.74	1.68	1.51	1.93	1.67	1.71	1.56	1.44
	NYISO	26.89	3.99	1.52	1.46	1.47	1.45	1.29	1.60	1.54	1.49	1.55	1.33	1.55	1.49	1.42	1.48	1.29
	ERCOT	15.83	3.13	1.23	1.15	1.13	1.13	1.15	1.26	1.16	1.15	1.16	1.13	1.26	1.17	1.16	1.15	1.13
	DE	2.22	1.36	1.11	1.03	1.10	1.09	1.07	1.13	1.04	1.12	1.10	1.08	1.11	1.03	1.10	1.09	1.07
	Average	**38.73**	**4.05**	**1.43**	**1.31**	**1.37**	**1.29**	**1.24**	**1.51**	**1.47**	**1.37**	**1.36**	**1.26**	**1.46**	**1.34**	**1.35**	**1.31**	**1.23**
Spring	**CAISO**	96.95	7.29	1.99	1.87	1.51	1.54	1.34	2.05	1.73	1.68	1.63	1.39	2.01	1.94	1.82	1.56	1.35
	NYISO	28.79	4.11	1.54	1.42	1.47	1.44	1.33	1.58	1.45	1.49	1.49	1.34	1.57	1.45	1.52	1.49	1.34
	ERCOT	10.09	2.56	1.27	1.17	1.22	1.15	1.15	1.33	1.21	1.27	1.19	1.17	1.29	1.16	1.20	1.17	1.14
	DE	2.04	1.31	1.07	1.03	1.07	1.07	1.05	1.08	1.03	1.07	1.08	1.05	1.08	1.03	1.07	1.08	1.05
	Average	**34.47**	**3.81**	**1.47**	**1.37**	**1.32**	**1.30**	**1.22**	**1.51**	**1.35**	**1.38**	**1.34**	**1.24**	**1.49**	**1.39**	**1.40**	**1.32**	**1.22**
Summer	**CAISO**	25.10	3.86	1.56	1.36	1.51	1.41	1.32	1.56	1.37	1.51	1.42	1.33	1.60	1.39	1.54	1.44	1.34
	NYISO	19.96	3.48	1.33	1.26	1.31	1.34	1.24	1.35	1.29	1.28	1.38	1.26	1.35	1.27	1.30	1.38	1.24
	ERCOT	5.91	2.03	1.17	1.07	1.14	1.12	1.09	1.20	1.10	1.14	1.13	1.09	1.18	1.07	1.14	1.13	1.07
	DE	2.31	1.37	1.08	1.03	1.08	1.07	1.06	1.09	1.04	1.09	1.08	1.06	1.09	1.03	1.08	1.08	1.07
	Average	**13.32**	**2.68**	**1.29**	**1.18**	**1.26**	**1.23**	**1.18**	**1.30**	**1.20**	**1.25**	**1.25**	**1.19**	**1.30**	**1.19**	**1.27**	**1.25**	**1.18**
Fall	**CAISO**	51.84	5.42	1.58	1.70	1.39	1.42	1.29	1.64	1.89	1.43	1.47	1.31	1.63	1.80	1.44	1.45	1.30
	NYISO	36.04	4.57	1.71	1.71	1.67	1.61	1.40	1.81	1.77	1.75	1.75	1.43	1.74	1.72	1.64	1.65	1.40
	ERCOT	7.26	2.22	1.22	1.12	1.20	1.13	1.08	1.23	1.15	1.16	1.14	1.09	1.23	1.12	1.16	1.13	1.08
	DE	2.12	1.33	1.12	1.05	1.11	1.09	1.09	1.15	1.08	1.12	1.11	1.10	1.12	1.06	1.12	1.09	1.09
	Average	**24.31**	**3.38**	**1.41**	**1.40**	**1.34**	**1.31**	**1.21**	**1.46**	**1.47**	**1.36**	**1.36**	**1.23**	**1.43**	**1.43**	**1.34**	**1.33**	**1.22**
Year	**CAISO**	70.97	6.28	1.75	1.63	1.55	1.47	1.35	1.83	1.78	1.59	1.55	1.39	1.79	1.70	1.63	1.50	1.36
	NYISO	27.92	4.06	1.52	1.46	1.48	1.46	1.31	1.58	1.51	1.50	1.54	1.34	1.55	1.48	1.47	1.50	1.32
	ERCOT	9.77	2.53	1.22	1.13	1.17	1.13	1.12	1.25	1.15	1.18	1.16	1.12	1.24	1.13	1.17	1.14	1.11
	DE	2.17	1.34	1.09	1.04	1.09	1.08	1.07	1.11	1.05	1.10	1.09	1.07	1.10	1.04	1.09	1.08	1.07
	Average	**27.71**	**3.76**	**1.40**	**1.31**	**1.32**	**1.28**	**1.21**	**1.44**	**1.37**	**1.34**	**1.33**	**1.23**	**1.42**	**1.34**	**1.34**	**1.30**	**1.21**

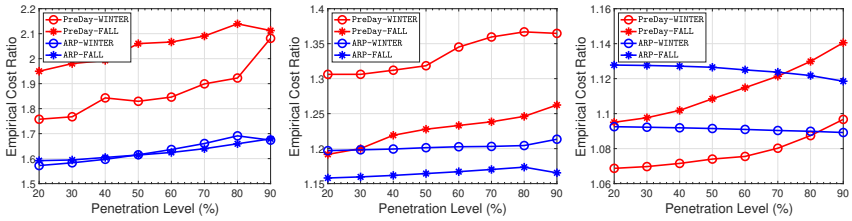
Table 4. Comparison of different algorithms using different energy storage technologies (CAES stands for Compressed Air Energy Storage)

Market	θ	Lithium-Ion ($\rho_c/B = 0.35$)					Lead-Acid ($\rho_c/B = 0.2$)					CAES ($\rho_c/B = 0.05$)				
		NoSTR	PreDay	LypOpt	OnFix	ARPRate	NoSTR	PreDay	LypOpt	OnFix	ARPRate	NoSTR	PreDay	LypOpt	OnFix	ARPRate
CAISO	25.10	1.49	1.34	1.44	1.33	1.31	1.47	1.33	1.44	1.32	1.32	1.43	1.31	1.42	1.29	1.32
NYISO	19.96	1.27	1.21	1.24	1.24	1.21	1.25	1.19	1.22	1.22	1.20	1.23	1.17	1.20	1.19	1.18
ERCOT	7.26	1.14	1.10	1.11	1.13	1.07	1.14	1.10	1.13	1.13	1.07	1.13	1.09	1.13	1.12	1.07
DE	2.12	1.07	1.03	1.05	1.08	1.05	1.06	1.04	1.05	1.08	1.05	1.06	1.03	1.05	1.08	1.05
Average	**13.32**	**1.24**	**1.17**	**1.21**	**1.19**	**1.16**	**1.23**	**1.16**	**1.21**	**1.18**	**1.16**	**1.21**	**1.15**	**1.20**	**1.17**	**1.15**

Observation 1. ARP *achieves a significantly smaller average cost ratio than the worst-case competitive ratio guarantee provided by our theoretical analysis.* The average theoretical competitive ratio (α in Equation (11)) for year-round experiments over four locations is 3.76, while the empirical cost ratios for ARP are much smaller, i.e., 1.21 for no renewable and with solar, and 1.23 for wind.

Observation 2. ARP *outperforms alternative algorithms, when averaged across the entire year and all four locations, and with/without renewables and it is close to offline optimal* OPT. For example, in the case with wind as the renewable source, the last row of Table 3 shows that ARP outperforms NoSTR by 15%, PreDay by 10%, LypOpt by 8%, and OnFix by 7.5%. Further, on average over the whole year ARP achieves a cost ratio of 1.21 to 1.23, i.e., a cost that is within 21–23% of the cost of OPT. However, there are a few cases in which other algorithms outperform ARP, e.g., PreDay in Frankfurt/DE. The reason is investigated in the next observation.

Observation 3. PreDay *is the best algorithm in settings with low price uncertainty and recurring daily price patterns.* To elaborate this observation, we need to further investigate the dynamics of the real-time prices in DE market. Figure 2 shows the real-time prices in three days in August



(a) New York (b) Dallas (c) Frankfurt

Fig. 6. The impact of renewable penetration

2017 for NYISO with high fluctuation ratio and DE Market with low price fluctuations. One can see that in DE Market the prices do not fluctuate a lot and there is almost a regular daily pattern. This regular daily pattern is the key to PreDay's good performance since it uses the previous day values to derive the procurement plan for today. However, the irregular pattern in NYISO and other markets (as shown in Figure 2) motivates our general ARP approach, since relying on the past information or stochastic modeling is less effective in these real-world markets. Predictably, PreDay does not perform as well in CAISO and NYISO markets in Table 2 (the cost ratio of 1.63 and 1.46 for PreDay as compared to 1.35 and 1.31 for ARP).

Observation 4. *With increased uncertainty due to renewable penetration, the performance of* ARP *is robust, however, the performance of* PreDay *degrades substantially.* The performance of ARP slightly degrades with injection of 50% renewable. The performance of PreDay, however, degrades substantially (e.g., from 1.31 to 1.37 for wind). To further elaborate this, in Figure 6, we compare the performance of ARP and PreDay in different seasons and locations as the penetration level varies. The result signifies the robust performance of ARP and degradation of PreDay with increased renewable penetration.

Observation 5. *The seasonal and locational patterns result in different degrees of uncertainty, thereby impact the performance of the algorithms substantially and increased uncertainty increases the cost ratio of algorithms.* For example, in ARP, the year-round cost ratio in Los Angles as the most uncertain location (highest $\theta = 70.97$, on average) is 1.47, while the same value for Frankfurt (lowest $\theta = 2.17$) as the least uncertain one is 1.07. As for seasonal variations, ARP achieves the cost ratio of 1.18 in the summer as the least uncertain season (with average $\theta = 13.32$), while this value is 1.24 in the winter as the most uncertain scenario (with $\theta = 38.73$).

4.4 Results for ARPRate

To evaluate the performance of ARPRate, we consider identical charging and discharge rates, i.e., $\rho_c = \rho_d$, and normalize it against the storage capacity. Hence, we define $\rho = \rho_c/B$ as a measure of the rate at which an energy storage is charged/discharged relative to its capacity. This measure is similar to C-rate and E-rate values in battery specifications [52]. The small values of ρ represent the energy storage is slow in charge/discharge and large values represents fast storages. A broad spectrum of energy storage technologies are integrated in the operations of the data centers, each with a different ρ. To obtain practical values, we use the energy density as the normalized capacity, and power density as the normalized discharge rate from [29]. We choose four common categories of storages in data centers based on their ρ values [20, 56]: (1) Compressed Air Energy Storage (CAES) with $\rho \approx 0.05$; (2) Lead-Acid (LA) with $\rho \approx 0.2$; (3) Lithium-Ion (LI) with $\rho \approx 0.35$, and (4) Flywheels $\rho \approx 1$. For Flywheels $\rho = 1$ which means that there is no rate constraints and it reduces

to ARP. Hence, in this experiment, we investigate the performance of ARPRate for the first three technologies, and the results are reported in Table 4.

Observation 6. *The performance of* ARPRate *improves as ρ increases, i.e., the rate constraints becomes more relax, while the performance of* PreDay *exhibits no regular pattern.* This observation is inferred from the results Table 4 that reports the results for three representative energy storage technologies.

5 RELATED WORK

The problem studied in this paper generalizes a long history of work focused on online linear optimization, e.g., [13, 23, 39, 48], and has intrinsic similarities to several other classic algorithmic problems regarding online search, such as the *time series search problem* [43], the *one-way trading problem* [25], the *multiple-choice secretary problem* [9] and the *online knapsack problem* [16]. The most relevant problem is the k-min search problem [43]. In this problem, a player wants to buy $k \geq 1$ units of an asset with the goal of minimizing the cost. At any slot $t = \{1, \ldots, T\}$, the player is presented a price $p(t)$, and must immediately decide whether or not to buy one unit of the asset given $p(t)$. In online setting, the difficulty of above problems comes from the uncertainty of the information for future time slots, for example, the market pricing in the k-min search problem.

The idea of using an adaptive pricing function that is used in ARP is adapted from the optimal online algorithm design for the k-min search problem in [43] and QoS buffer management problem in [60]. However, different from the k-min search problem, in OLIM, in addition to the uncertainty in the market pricing, we have another source of uncertainty due to online arrival of energy demand. Therefore, direct application of existing algorithms for the k-min search problem can only guarantee sub-optimal competitive ratios. We tackled the additional demand uncertainty by introducing the novel notion of virtual inventories. With this view, one can see that satisfying each net demand as an instance of the k-min search problem, where k is the energy demand and must be purchased within the time horizon. Different from the k-min search problem, where k, as the amount of required commodity is given a priori, ARP deals with online arrival of different commodities, i.e., demands in each slot. Moreover, different from [60], which is a profit maximization problem, this paper studies a cost minimization problem with more complicated set of constraints due to rate limits of storage systems. Consequently, it comes with different algorithms and proof techniques.

The idea of modeling energy storage management using search problems is used in [21, 22]. However, [21, 22] adopt a fixed thresholding policy, leading to suboptimal online algorithms with competitive ratio of $\sqrt{\theta}$. In contrast, this work leverages the adaptive thersholding policy and is the first that effectively tackles the demand uncertainty in OLIM by introducing the notion of virtual inventories, and develops online algorithms that achieve the optimal competitive ratio for basic and rate limit settings. We both theoretically (in Figure 1) and experimentally (in §4) compare our results with the algorithm proposed in [22], and show its better performance in theory and experiments.

Note that recently the one-way trading problem has been extended to include general convex functions [42]. This formulation allows for inventory constraints; however the inventory constraints in [42] are simply a fixed budget constraint. This paper, in contrast, tackles the online linear programs with *inventory management* constraints, that takes into account the evolution of the inventory over the time. This additional complexity makes the design and analysis of algorithms completely different from [42], and more challenging. Also, in another recent study [59], the online linear programs have been extended to the cases with adaptive budget constraints, however, this result also fails to consider inventory management constraints.

Energy procurement problems similar to the one used in our case study have have been studied in literature frequently, typically using empirical evaluations [32, 49, 63, 64]. These approaches require exact modeling of the inputs, which is difficult to obtain due to multidimensional uncertainty in the problem. However, another approach is to use the historical data and train machine learning models to find the energy procurement decisions [8]. Similar to other empirical approaches, this approach fails to achieve a provable performance guarantee in practice.

In contrast, some papers have used stochastic optimization tools, such as Markov decision processes [55], to propose optimal policies given the probabilistic modeling of uncertain inputs. Deviations from the stochastic models may severely degrade the overall performance of such designs however. Alternatively, Lyapunov optimization has also been proposed [33, 38, 54]. It obtains optimal control policies over infinite horizon with i.i.d. assumptions for the inputs. In practice, however, the time horizon is usually finite and the inputs may not be i.i.d. Different from the aforementioned approaches, this paper leverages online competitive design. This approach assumes that neither the exact values nor the stochastic modeling of the uncertain input are known in advance, and tries to achieve a bounded competitive ratio for the worst-case input that can be generated by an adversary.

6 CONCLUDING REMARKS

In this paper, we studied a challenging variation of online linear optimization where inventory management constraints are considered. While some special cases of inventory constraints have been considered previously, e.g., budget constraints [42, 59], no results to this point allow for general inventory constraints. For this problem, we introduced new online algorithms that have competitive ratios that match the optimal achievable by any online algorithm, both with and without rate constraints on the inventory. The algorithms we introduced, `ARP` and `ARPRate`, rely on novel adaptive techniques that have not been used in previous algorithms.

To illustrate the improvements that `ARP` and `ARPRate` provide over existing algorithms, we performed a case study for the application of energy storage management. We evaluated the proposed algorithms using extensive data-traces and showed that the improvements are not just visible in worst-case performance bounds, they are realized in real-world scenarios.

The work here motivates a variety of important research directions. On the applied side, the results in the case study are quite promising and merit further work to implement and test the algorithms in real systems. On the theoretical side, OLIM is highly related to a wide variety of classical algorithms problems and it will be interesting to investigate whether the ideas that underlie `ARP` and `ARPRate` can yield improved algorithms in other settings.

7 ACKNOWLEDGMENTS

This work was funded by the National Science Foundation through the CNS-1908298, CNS-1763617, AitF-1637598, CNS-1518941, CPS-1544771, EPCN-1711188, CAREER-1752362, AMPS-1736448 grants, and ARO: W911NF-17-1-0092, DoE: ENERGISE-DE-EE0008006 grants, and a Google Faculty Research Award. Lin Yang wants to acknowledge the support from Schneider Electric, Lenovo Group (China) Limited and the Hong Kong Innovation and Technology Fund (ITS/066/17FP) under the HKUST-MIT Research Alliance Consortium.

REFERENCES

[1] 2016. Wholesale Electricity Market Design Initiatives in the United States: Survey and Research Needs. *EPRI,Technical Results, available at https:// www.epri.com/pages/product/000000003002009273/* (2016).

[2] 2020. Eastern and Western Data Sets. available at https://www.nrel.gov/grid/eastern-western-wind-data.html.

[3] 2020. Environmental Responsibility Report - Apple. available at https://www.apple.com/environment/pdf/Apple_Environmental_Responsibility_Report_2018.pdf.

[4] 2020. Google Data Center in Changhua County, Taiwan. available at https://www.google.com/about/datacenters/inside/locations/changhua-county/.

[5] 2020. Google Data Center in St. Ghislain, Belgium. available at https://www.google.com/about/datacenters/inside/locations/st-ghislain/.

[6] 2020. Open Power System Data. available at https://data.open-power-system-data.org/.

[7] 2020. Tesla's Powerpack proposes battery power for data centers. available at https://www.datacenterdynamics.com/analysis/teslas-powerpack-proposes-battery-power-for-data-centers/.

[8] Sohaib Ahmad, Arielle Rosenthal, Mohammad H Hajiesmaili, and Ramesh K Sitaraman. 2019. Learning from Optimal: Energy Procurement Strategies for Data Centers. In *Proc. of ACM eEnergy*. 326–330.

[9] M Ajtai, N Megiddo, and O Waarts. 2001. Improved algorithms and analysis for secretary problems and generalizations. *SIAM Journal on Discrete Mathematics* 14, 1 (2001), 1–27.

[10] Susanne Albers. 2017. On Energy Conservation in Data Centers. In *Proc. of ACM SPAA*. 35–44.

[11] Susanne Albers and Jens Quedenfeld. 2018. Optimal Algorithms for Right-Sizing Data Centers. In *Proc. of ACM SPAA*. 363–372.

[12] Bahram Alinia, Mohammad Sadegh Talebi, Mohammad H Hajiesmaili, Ali Yekkehkhany, and Noel Crespi. 2018. Competitive online scheduling algorithms with applications in deadline-constrained EV charging. In *Proc. of IEEE/ACM IWQoS*. 1–10.

[13] Baruch Awerbuch and Robert Kleinberg. 2008. Online linear optimization and adaptive routing. *J. Comput. System Sci.* 74, 1 (2008), 97–114.

[14] Moshe Babaioff, Nicole Immorlica, David Kempe, and Robert Kleinberg. 2008. Online auctions and generalized secretary problems. *ACM SIGecom Exchanges* 7, 2 (2008), 7.

[15] Luiz André Barroso and Urs Hölzle. 2007. The case for energy-proportional computing. *Computer* 12 (2007), 33–37.

[16] H Böckenhauer, D Komm, R Královič, and P Rossmanith. 2014. The online knapsack problem: Advice and randomization. *Theoretical Computer Science* 527 (2014), 61–72.

[17] A. Borodin and R El-Yaniv. 1998. *Online computation and competitive analysis.* Cambridge University Press.

[18] Niv Buchbinder and Joseph Naor. 2009. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research* 34, 2 (2009), 270–286.

[19] CAISO 2020. CAISO electricity market. available at https://www.caiso.com/.

[20] Santosh Chalise, Amir Golshani, Shekhar Raj Awasthi, Shanshan Ma, Bijen Raj Shrestha, Labi Bajracharya, Wei Sun, and Reinaldo Tonkoski. 2015. Data center energy systems: Current technology and future direction. In *Proc. of IEEE PES*.

[21] Chi-Kin Chau and Lin Yang. 2016. Competitive online algorithms for geographical load balancing in data centers with energy storage. In *Proceedings of the 5th International Workshop on Energy Efficient Data Centres*. 1.

[22] Chi-Kin Chau, Guanglin Zhang, and Minghua Chen. 2016. Cost minimizing online algorithms for energy storage management with worst-case guarantee. *IEEE Transactions on Smart Grid* 7, 6 (2016), 2691–2702.

[23] Chao-Kai Chiang, Tianbao Yang, Chia-Jung Lee, Mehrdad Mahdavi, Chi-Jen Lu, Rong Jin, and Shenghuo Zhu. 2012. Online optimization with gradual variations. In *Proc. of COLT*. 6–1.

[24] Aron Dobos. 2014. *PVWatts version 5 manual.* National Renewable Energy Laboratory Golden, CO.

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

[26] ERCOT 2020. ERCOT Electricity Market. available at http://www.ercot.com.

[27] Richard Evans and Jim Gao. 2016. DeepMind AI Reduces Google Data Centre Cooling Bill by 40%. *https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/* (2016).

[28] Guy Even, Moti Medina, and Dror Rawitz. 2018. Online Generalized Caching with Varying Weights and Costs. In *Proc. of ACM SPAA*. 205–212.

[29] FACTs 2020. Comparison of commercial battery types. available at https://en.wikipedia.org/wiki/Comparison_of_commercial_battery_types.

[30] German Electricity Market 2020. German Electricity Market.

[31] Mahdi Ghamkhari, Adam Wierman, and Hamed Mohsenian-Rad. 2016. Energy Portfolio Optimization of Data Centers. *IEEE Transactions on Smart Grid* (2016).

[32] Sriram Govindan, Di Wang, Anand Sivasubramaniam, and Bhuvan Urgaonkar. 2013. Aggressive Datacenter Power Provisioning with Batteries. *ACM Transactions on Computing Systems* 31, 1 (2013), 2:1–2:31.

[33] Yuanxiong Guo and Yuguang Fang. 2013. Electricity cost saving strategy in data centers by using energy storage. *IEEE Transactions Parallel and Distributed Systems* 24, 6 (2013), 1149–1160.

[34] Mohammad H Hajiesmaili, Chi-Kin Chau, Minghua Chen, and Longbu Huang. 2016. Online microgrid energy generation scheduling revisited: The benefits of randomization and interval prediction. In *Proc. of ACM eEnergy*.
[35] Mohammad H Hajiesmaili, Minghua Chen, Enrique Mallada, and Chi-Kin Chau. 2017. Crowd-Sourced Storage-Assisted Demand Response in Microgrids. In *Proc. of ACM eEnergy*. 91–100.
[36] Mohammad H Hajiesmaili, Lei Deng, Minghua Chen, and Zongpeng Li. 2017. Incentivizing device-to-device load balancing for cellular networks: An online auction design. *IEEE Journal on Selected Areas in Communications* 35, 2 (2017), 265–279.
[37] Elad Hazan et al. 2016. Introduction to online convex optimization. *Foundations and Trends® in Optimization* 2, 3-4 (2016), 157–325.
[38] Longbo Huang, Jean Walrand, and Kannan Ramchandran. 2012. Optimal demand response with energy storage management. In *Proc. IEEE SmartGridComm*. 61–66.
[39] Adam Kalai and Santosh Vempala. 2002. Geometric algorithms for online optimization. In *Journal of Computer and System Sciences*.
[40] Kia Khezeli and Eilyan Bitar. 2018. Risk-sensitive learning and pricing for demand response. *IEEE Transactions on Smart Grid* 9, 6 (2018), 6000–6007.
[41] Minghong Lin, Adam Wierman, Lachlan LH Andrew, and Eno Thereska. 2013. Dynamic right-sizing for power-proportional data centers. *IEEE/ACM Transactions on Networking* 21, 5 (2013), 1378–1391.
[42] Qiulin Lin, Hanling Yi, John Pang, Minghua Chen, Adam Wierman, Michael Honig, and Yuanzhang Xiao. 2019. Competitive online optimization under inventory constraints. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 3, 1 (2019), 10.
[43] J. Lorenz, K. Panagiotou, and A Steger. 2009. Optimal algorithms for k-search with application in option pricing. *Algorithmica* 55, 2 (2009), 311–328.
[44] Zhoujia Mao, Can Emre Koksal, and Ness B Shroff. 2016. Optimal online scheduling with arbitrary hard deadlines in multihop communication networks. *IEEE/ACM Transactions on Networking* 24, 1 (2016), 177–189.
[45] Esther Mohr, Iftikhar Ahmad, and Günter Schmidt. 2014. Online algorithms for conversion problems: a survey. *Surveys in Operations Research and Management Science* 19, 2 (2014), 87–104.
[46] Erik Nygren, Ramesh K Sitaraman, and Jennifer Sun. 2010. The akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review* 44, 3 (2010), 2–19.
[47] NYISO 2020. NYISO Electricity Market. available at http://www.nyiso.com.
[48] Francesco Orabona and Dávid Pál. 2015. Scale-free algorithms for online linear optimization. In *Proc. of ALT*. 287–301.
[49] Darshan S Palasamudram, Ramesh K Sitaraman, Bhuvan Urgaonkar, and Rahul Urgaonkar. 2012. Using batteries to reduce the power costs of internet-scale distributed networks. In *Proc. of ACM SoCC*.
[50] Xiaoqi Ren, Palma London, Juba Ziani, and Adam Wierman. 2018. Datum: Managing Data Purchasing and Data Placement in a Geo-Distributed Data Market. *IEEE/ACM Transactions on Networking* 26, 2 (2018), 893–905.
[51] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
[52] MEV Team et al. 2008. A guide to understanding battery specifications. (2008).
[53] Jinlong Tu, Lian Lu, Minghua Chen, and Ramesh K Sitaraman. 2013. Dynamic provisioning in next-generation data centers with on-site power production. In *Proc. of ACM eEnergy*. 137–148.
[54] R. Urgaonkar, B. Urgaonkar, M.J. Neely, and A. Sivasubramaniam. 2011. Optimal power cost management using stored energy in data centers. In *Proc. of ACM SIGMETRICS*.
[55] Peter M van de Ven, Nidhi Hegde, Laurent Massoulié, and Theodoros Salonidis. 2013. Optimal control of end-user energy storage. *IEEE Transactions on Smart Grid* 4, 2 (2013), 789–797.
[56] Di Wang, Chuangang Ren, Anand Sivasubramaniam, Bhuvan Urgaonkar, and Hosam Fathy. 2012. Energy storage in datacenters: what, where, and how much?. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 40. 187–198.
[57] Hong Xu and Baochun Li. 2014. Reducing electricity demand charge for data centers with partial execution. In *Proc. of ACM eEnergy*. 51–61.
[58] Lin Yang, Lei Deng, Mohammad H Hajiesmaili, Cheng Tan, and Wing Shing Wong. 2018. An optimal algorithm for online non-convex learning. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 2 (2018), 25.
[59] Lin Yang, Mohammad H Hajiesmaili, and Wing S Wong. 2019. Online Linear Programming with Uncertain Constraints. In *Proc. of IEEE CISS*. 1–6.
[60] Lin Yang, Wing Shing Wong, and Mohammad H Hajiesmaili. 2017. An optimal randomized online algorithm for QoS buffer management. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 2 (2017), 36.
[61] Ying Zhang, Mohammad H Hajiesmaili, Sinan Cai, Minghua Chen, and Qi Zhu. 2018. Peak-aware online economic dispatching for microgrids. *IEEE Transactions on Smart Grid* 9, 1 (2018), 323–335.
[62] Zijun Zhang, Zongpeng Li, and Chuan Wu. 2017. Optimal posted prices for online cloud resource allocation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 1 (2017), 23.

[63] Wenli Zheng, Kai Ma, and Xiaorui Wang. 2014. Exploiting thermal energy storage to reduce data center capital and operating expenses. In *Proc. IEEE HPCA*. 132–141.
[64] Haihang Zhou, Jianguo Yao, Haibing Guan, and Xue Liu. 2015. Comprehensive understanding of operation cost reduction using energy storage for IDCs. In *Proc. IEEE INFOCOM*. 2623–2631.
[65] Ruiting Zhou, Zongpeng Li, and Chuan Wu. 2015. An online procurement auction for power demand response in storage-assisted smart grids. In *Proc. of IEEE INFOCOM*. 2641–2649.

A PROOF OF THEOREM 3.1

To prove this result we need to show that ARP respects all the constraints in OLIM. First, we can see that it respects the demand covering constraint, i.e., $x(t) \geq d(t) - b(t-1)$, by the projection in Equation (13). Second, we show that ARP always respects the capacity constraints. At time slot t, which lies in a reservation period (see Definition 3), the total amount of purchased asset from the beginning of current reservation period is equal to $\sum_{i=1}^{v} G_{B_i}(\xi_i)$, which is less than or equal to $\sum_{i=1}^{v} B_i$. The demand from the beginning of current reservation period is $\sum_{i=2}^{v} B_i$. The asset stored in the physical inventory is $\sum_{i=1}^{v} G_{B_i}(\xi_i) - \sum_{i=2}^{v} B_i$, which is less than or equal to $\sum_{i=1}^{v} B_i - \sum_{i=2}^{v} B_i = B_1$ according to the definition of $G_{B_i}(\xi_i)$. B_1 is the capacity of the physical inventory. That is, the amount of reserved asset always respects the capacity constraint.

B PROOFS RELATED TO ARP

B.1 Proof of Lemma 1

For each virtual inventory, ARP stores the asset only if the market price is less than the reservation price. Hence, the cost of the stored assets in the j-th inventory is less than $\int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db$. By aggregation over n reservation periods and virtual inventories, we can compute the aggregate cost incurred by ARP as

$$\sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \int_0^{\hat{b}_{i,j}} G_{B_{i,j}}^{-1}(b)db, \tag{21}$$

where there are $\hat{v}_i$ virtual inventory units at reservation period i, $\hat{\xi}_{i,j}$ and $\hat{b}_{i,j}$ is the final inventory level of virtual inventory j at reservation period i. The additional amount of asset needed to satisfy the demand in the idle period is equal to $D - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} + \hat{b}$, where D is the total demand during the time horizon, i.e., $D = \sum_{t \in \mathcal{T}} d(t)$, and $\hat{b}$ be the final inventory level of the physical inventory, i.e., $\hat{b} = b(T)$. Hence, the cost of the online algorithm during the idle period is at most

$$\left(D - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} + \hat{b} \right) p_{\max}, \tag{22}$$

Adding (21) and (22) completes the proof.

B.2 Proof of Lemma 2

The cost of an optimal offline solution, denoted as cost(OPT), can be split into two parts. To characterize a lower bound for the offline optimum let us define $F_i(\beta)$ as the minimum cost of purchasing β units of asset during the i-th reservation period. Let β_i be the asset purchased by the optimal offline solution during the i-th reservation period. Then, the cost of the optimal offline solution during the i-th reservation period is at least $F_i(\beta_i)$. Let $\tilde{p}$ be the minimum price during the idle periods, then we have that the cost of the offline algorithm is lower bounded by $\sum_{i=1}^{n} F_i(\beta_i) + (D - \sum_{i=1}^{n} \beta_i) \times \tilde{p}$. This completes the proof.

B.3 Additional Material Related to Lemma 4

We state the following lemma on the properties of function $G_{B_i}^{-1}(b)$ and $F_i(\beta)$ to facilitate the proof of Lemma 4. Let $B_{i,j}$ be the capacity of the j-th inventory and $\hat{b}_{i,j}$ be the final inventory level at reservation period i.

Lemma 6. *Defining $\hat{b}_i$ as the initial state of the inventory of the offline algorithm in the i-th reservation period, there is a worst-case input instance such that for all $0 < \beta < \beta' < \sum_{j=1}^{\hat{v}_i} B_{i,j} - \hat{b}_i$, we have*

(1) $F_i(0) = 0$;
(2) $F_i(\beta') > F_i(\beta)$;
(3) $F_i(\beta') - F_i(\beta) \leq \frac{p_{\max}}{\alpha}(\beta' - \beta)$.

Proof. Statements (1) and (2) are immediate. For the third statement, assume there is a worst instance $\omega = [\langle p(t), d(t)\rangle]_{t\in\mathcal{T}}$. The adversary can construct a new instance ω' by adding one time slot before each time slot of instance ω as follows

$$\omega' = \left[\left\langle \frac{p_{\max}}{\alpha}, 0\right\rangle, \langle p(1), d(1)\rangle, \left\langle \frac{p_{\max}}{\alpha}, 0\right\rangle, \langle p(2), d(2)\rangle, \ldots, \left\langle \frac{p_{\max}}{\alpha}, 0\right\rangle, \langle p(t), d(t)\rangle, \left\langle \frac{p_{\max}}{\alpha}, 0\right\rangle\right].$$

Note that this is possible since the adversary can set the length of time horizon. The market prices for the newly added time slots is $p_{\max}/\alpha$ and the demand is always equal to zero.

In this way, we construct a new instance under which the cost of the online algorithm does not change, and that of the offline optimal solution will not increase. Thus, ω' is also the worst instance.

Let $F_i(\beta)$ be the minimum cost when buying β units of asset during the i-th reservation period. When the optimal policy buys another $\beta' - \beta$, $\beta' < \sum_{j=1}^{\hat{v}_i} B_{i,j} - \hat{b}_i$, units of asset, the cost will not be larger than $\frac{p_{\max}}{\alpha}(\beta' - \beta)$, since the optimal policy can buy asset at any newly added time slots.

Thus, there is a worst instance, such that $F_i(\beta') - F_i(\beta) \leq p_{\max}/\alpha(\beta' - \beta)$, for $0 < \beta < \beta' < \sum_j \hat{b}_{i,j} - \hat{b}_i$. This completes the proof. □

C CONVERGENCE OF INITVS

Theorem 4. *Given a market price $p(t) \in [p_{\min}, p_{\max}]$, InitVS converges to a feasible solution B_v and $\hat{x}(t)$ which satisfies Equations (19) and (20) simultaneously.*

Proof. It is straightforward to see that B'_v is always larger than or equal to B_v. Further, if the value of $B'_v - B_v$ is larger than ε_1, the value of B_v will increase by at least ε_1. Thus, there must be an iteration such that $B'_v - B_v \leq \varepsilon_1$. □

D COMPETITIVE ANALYSIS OF ARPRATE

If $D = 0$, ARPRate the result follows from the analysis of ARP. hus, we focus our analysis on the case $D > 0$.

Similar to the analysis for ARP, we would like to upper bound the cost of ARPRate. To achieve this, first we give the following two lemmas which characterize properties of the worst instance for ARPRate. Lemma 7 implies that in worst case, the output constraint is not active. Lemma 5 characterizes an upper bound on the cost of ARPRate.

Lemma 7. *Under the worst case, $\tilde{x}(t) = 0$, for $\forall t \in \mathcal{T}$, where $\tilde{x}(t) = [d(t) - \rho_d - \hat{x}(t)]^+$.*

Proof. We prove this lemma by contradiction. Assume there is a worst instance $\omega = [\langle p(t), d(t)\rangle]_{t\in\mathcal{T}}$, where $\tilde{x}(t) > 0$ for time slot t. We can construct a new instance which is the same as ω except at the t-th time slot. For time slot t, the demand is set to $x(t) - \delta$, where $\delta < \tilde{x}(t)$, and the market price is equal to $p(t)$. In this way, the cost of ARP at time slot t will decrease by $p(t)\delta$. The costs on

other time slots are unchanged, because the modification on the demand does not influence the capacity and reservation price of virtual inventory according to the rules of ARPRate. On the other hand, the cost of OPT at time slot t will decrease by at least $p(t)\delta$, since the procurement amount of OPT is larger than δ. In this case, we have a new instance ω' under which the cost ratio is larger than that of the worst instance, contradicting the assumption. This completes the proof. □

Lemma 7 implies that under the worst case the output constraint is not active. Thus, all that is left is to take into account the influence of the input rate constraint. Recall that in the basic version, ARP, the procurement amount is always larger than or equal to $\hat{x}(t)$, which is computed in Equation (19). With an input constraint, $\hat{x}(t)$ may not be satisfied, and the maximum procurement amount is limited by $\rho_c + d(t)$. We define $\mathcal{T}_r \subset \mathcal{T}$ be the set of time slots at which the input rate truncates the procurement amount. That is, the following equation holds for $t \in \mathcal{T}_r$.

$$\sum_{i \le v} \left[G_{B_i}(p(t)) - G_{B_i}(\xi_i) \right]^+ > \rho_c + d(t).$$

For $t \in \mathcal{T}_r$, we define $p'(t)$ as the value which satisfies the following equation.

$$\sum_{i \le v} \left[G_{B_i}(p'(t)) - G_{B_i}(\xi_i) \right]^+ = \rho_c + d(t), \text{ for } \forall t \in \mathcal{T}.$$

$p'(t)$ is the actual reservation price computed in Algorithm 4, and obviously, $p'(t) > p(t)$. Let $\mu(t) = p'(t) - p(t)$ denotes the difference between $p'(t)$ and $p(t)$.

Similarly to Lemma 1, we use Lemma 5 to upper bound the cost of the ARPRate.

Proof. Proof of Lemma 5 By Lemma 7, we have that, under the worst case, $\tilde{x}(t) = 0$ and the amount of reserved asset is always less than or equal to the value computed in Equation (19). Based on the analysis in 1, we have that the cost of ARPRate is upper bounded by $Q + \left(D - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} + \hat{b} \right) \cdot p_{\max}$. Moreover, $\forall t \in \mathcal{T}_r$, the actual price is less than the reservation price by $\mu(t)$, so the above upper bound is further modified to $Q + \left(D - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} + \hat{b} \right) \cdot p_{\max} - \sum_{t \in \mathcal{T}_r} \mu(t) x(t)$. This completes the proof. □

With the above two lemmas, the competitive ratio of ARPRate is upper bounded by

$$\begin{aligned}
\frac{\text{cost(ARPRate)} - \hat{b} p_{\max}}{\text{cost(OPT)}} &\le \frac{Q + \left(D - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} \right) \cdot p_{\max} - \sum_{t \in \mathcal{T}_r} \mu(t) x(t)}{\sum_{i=1}^{n} F_i'(\beta_i) + \left(D - \sum_{i=1}^{n} \beta_i \right) \cdot \tilde{p}} \\
&\le \max \left\{ \frac{Q + \left(\sum_{i=1}^{n} \beta_i - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} \right) \cdot p_{\max} - \sum_{t \in \mathcal{T}_r} \mu(t) x(t)}{\sum_{i=1}^{n} F_i'(\beta_i)}, \alpha \right\} \\
&\le \max \left\{ \frac{Q + \left(\sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} B_{i,j} - \sum_{i=1}^{n} \sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j} \right) p_{\max} - \sum_{t \in \mathcal{T}_r} \mu(t) x(t)}{\sum_{i=1}^{n} F_i' \left(\sum_{j=1}^{\hat{v}_i} B_{i,j} \right)}, \alpha \right\},
\end{aligned}$$

where $F_i'(\beta)$ is defined as the minimum cost of purchasing β units of asset during the i-th reservation period. The definition of $F_i'(\beta)$ is similar to that of $F_i(\beta)$ for for the basic version of the problem and it also respects the properties listed in Lemma 6.

During the lifetime of the j-th virtual inventory of the i-th reservation period, the minimum reservation price is $\hat{\xi}_{i,j}$. The cost of the optimal algorithm satisfies

$$\sum_{i=1}^{n} F_i'\left(\sum_{j=1}^{\hat{v}_i} B_{i,j}\right) \geq \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} \hat{\xi}_{i,j} B_{i,j} - \sum_{t\in\mathcal{T}_r} \mu(t)x^*(t).$$

Then, we have

$$\mathsf{cr}(\mathtt{ARPRate}) \leq \max\left\{\frac{Q + \left(\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} B_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j}\right) p_{\max} - \sum_{t\in\mathcal{T}_r} \mu(t)x(t)}{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} \hat{\xi}_{i,j} B_{i,j} - \sum_{t\in\mathcal{T}_r} \mu_t x^*(t)}, \alpha\right\}$$

The following lemma characterizes a bound on $x(t)/x^*(t)$.

Lemma 8. *Under the worst case, we have that $x(t)/x^*(t)$ is less than or equal to the competitive ratio, for any $t \in \mathcal{T}_r$.*

Proof. Let $\omega = [\langle p(t), d(t)\rangle]_{t\in\mathcal{T}}$ be the worst instance and at time slot t, there is $x(t)/x^*(t) > \mathsf{cr}(\mathtt{ARPRate})$. We can construct a new instance ω' by increasing the market price at time slot t by δ, where $\delta \leq \mu(t)$. That is

$$\omega' = [\langle p(1), d(1)\rangle, \ldots, \langle p(t)+\delta, d(t)\rangle, \ldots, \langle p(T), d(T)\rangle].$$

Under instance ω', the cost of ARP will increase by $x(t)\delta$, and that of OPT increase by less than $\frac{x(t)\delta}{\mathsf{cr}(\mathtt{ARPRate})}$. In this way, we can get a worse instance ω' than ω, contradicting the assumption that ω is the worst instance. This completes the proof. □

By the above lemma, we have that,

$$\frac{\sum_{t\in\mathcal{T}_r} \tilde{p}_t x(t)}{\sum_{t\in\mathcal{T}_r} \tilde{p}_t x^*(t)} \leq \mathsf{cr}(\mathtt{ARPRate}).$$

This yields

$$\mathsf{cr}(\mathtt{ARPRate}) \leq \max\left\{\frac{Q + \left(\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} B_{i,j} - \sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} \hat{b}_{i,j}\right) p_{\max}}{\sum_{i=1}^{n}\sum_{j=1}^{\hat{v}_i} \hat{\xi}_{i,j} B_{i,j}}, \alpha\right\}.$$

Combining the above with Lemma 3, we have $\mathsf{cr}(\mathtt{ARPRate}) \leq \alpha$, as desired.

Received October 2019; revised December 2019; accepted January 2020