Competitive Online Convex Optimization with Switching Costs and Ramp Constraints

Ming Shi[†], Xiaojun Lin[†], Sonia Fahmy[‡]
[†]School of Electrical and Computer Engineering
[‡]Department of Computer Science
Purdue University, West Lafayette, IN, USA
Email: {sming,linx,fahmy}@purdue.edu

Dong-Hoon Shin
AT&T Labs, San Ramon, CA, USA
Email: misaengma@gmail.com

***Abstract*—We investigate competitive online algorithms for online convex optimization (OCO) problems with linear in-stage costs, switching costs and ramp constraints. While OCO problems have been extensively studied in the literature, there are limited results on the corresponding online solutions that can attain small competitive ratios. We first develop a powerful computational framework that can compute an optimized competitive ratio based on the class of affine policies. Our computational framework can handle a fairly general class of costs and constraints. Compared to other competitive results in the literature, a key feature of our proposed approach is that it can handle scenarios where infeasibility may arise due to hard feasibility constraints. Second, we design a robustification procedure to produce an online algorithm that can attain good performance for both average-case and worst-case inputs. We conduct a case study on Network Functions Virtualization (NFV) orchestration and scaling to demonstrate the effectiveness of our proposed methods.**

I. Introduction

We study online convex optimization (OCO) with switching costs and ramp constraints, which has become an important tool for modeling many classes of practical decision problems with uncertainty, including machine learning [1], networking [2], cloud computing [3] and cyber-physical systems [4]. In the type of OCO problem that we are interested in, at each time t, the environment (or adversary) reveals the input $A(t)$. The decision maker then must choose the decision $X(t)$ from a convex set and incurs a linear cost $C_t(X(t), A(t))$. Additionally, there is a switching cost that penalizes the change $|X(t) - X(t-1)|$ for each time t and/or a ramp constraint on the magnitude of the change $X(t) - X(t-1)$. The goal is to minimize the overall cost, which is non-linear (and convex) due to the switching cost. Further, since future inputs, i.e., $A(t+1), A(t+2), ..., A(T)$, are not revealed at time t, this problem becomes an online decision problem. Clearly, this formulation is general and can model many important online decision problems. For example, in the Network Functions Virtualization (NFV) orchestration and scaling problem [5], a data center operator must decide where to instantiate Virtualized Network Functions (VNFs) on virtual machines (VMs) or containers such as Docker [6] running on servers in order to process incoming traffic. Here, $A(t)$ represents the traffic load, which can be uncertain before time t; $X(t)$ represents the mapping from VNFs to VMs or containers; the linear cost $C_t(X(t), A(t))$ represents VM/container cost and/or distance cost (e.g., latency) [7]. Finally, the switching cost captures the overhead for migrating demand/state among different VNF instances and the cost of instantiating and tearing down VNF instances. As another example, in the real-time dispatch problem in power systems [8], the system operator needs to decide how to adjust the power level of the generators to balance the electricity demand. Here, $A(t)$ represents the uncertain demand and renewable supply revealed on different buses at time t; $X(t)$ represents the dispatch decisions of the generators; the linear cost $C_t(X(t), A(t))$ represents the generation cost of the dispatch decisions. Finally, generators have ramp constraints so that their power output level can at most change by a given value each time.

In this paper, we aim to develop online algorithms with low competitive ratios for this type of OCO problem. Here, the competitive ratio is the maximum ratio of the cost of an online algorithm to that of the offline optimal solution (the latter assuming that all inputs are known in advance), taken over all possible input sequences. Despite the importance of OCO, there are limited results in the literature on the corresponding competitive online algorithms. Most studies of OCO focus on the regret [1], [9], [10], which is the difference between the cost of the online algorithm and the cost of the optimal *static* decision chosen in advance. Such a static decision, however, does not respond to inputs dynamically, and thus is not a useful reference for comparison in application scenarios such as NFV and power systems. Among those that study competitive performance against the *dynamic* offline optimal solution, the references [11] and [12] show that the competitive difference of their proposed algorithms are upper bounded, which implies that the competitive ratio may also be upper bounded under certain conditions. However, they do not provide a way to optimize the competitive ratio, and thus the resulting competitive ratio may still be fairly large. To the best of our knowledge, there is no systematic framework to optimize the competitive ratio of online algorithms for the type of OCO problems that this paper studies.

To address this open question, our first contribution (in Sec. III) is to develop a general and tractable framework that

allow us to find online algorithms with optimized competitive ratios for this type of OCO problem. Capitalizing on the ideas from robust optimization [13], we consider the case where the future uncertain inputs, $A(1), A(2), ..., A(T)$, are from an uncertainty set $\mathcal{U}$. In practice, such an uncertainty set $\mathcal{U}$ can be obtained from imprecise forecasts and historical data [8], [14]. Yet, searching among all possible online decisions appears to be intractable. Instead, in order to obtain simpler policies with reasonably good performance, we focus on affine policies, where the decision $X(t)$ at time t is an affine function of the input $A(t)$, i.e., $X(t) = \eta(t) + H(t)A(t)$. Thus, designing an online algorithm boils down to designing the parameters $\eta(t)$ and $H(t)$ (which depends only on the uncertainty set $\mathcal{U}$ but not the actual inputs). Through this restriction to affine policies, we can formulate the problem of optimizing the competitive ratio as a minmax optimization problem. We call the resulting online algorithm the Robust Affine Policy (RAP). Since this optimization problem is still non-convex, we propose approximations that effectively convexify the problem and make it tractable. In this way, our proposed computational framework can be used to design online algorithms with optimized competitive ratios for OCO problems with fairly complex structures and constraints. We note that the idea of affine policies has been used in adjustable robust optimization [15] to minimize the worst-case cost. In contrast, our approach applies affine policies to minimize the competitive ratio. This approach has not been studied before and gives rise to new technical difficulties as we discuss in Sec. III.

A key feature of our proposed approach is that it can gracefully handle situations where infeasibility may arise due to hard feasibility constraints in the OCO problem. By "hard feasibility constraints," we refer to situations where, after the input $A(t)$ is revealed at time t, no feasible decision $X(t)$ can be found. This situation usually arises when there are both ramp constraints and demand-supply balance constraints. Specifically, due to the ramp constraint, the decision $X(t)$ at time t cannot differ too much from $X(t-1)$. Thus, when the demand is too high at time t, but $X(t-1)$ was not properly chosen, there may not exist any $X(t)$ that can meet the demand. Such a problem can occur in both power systems and NFV. In contrast, the studies in [11] and [12] do not consider such hard feasibility constraints because they do not simultaneously enforce ramp constraints and demand-supply balance constraints. As a result, their competitive guarantees would not hold when there were hard feasibility constraints. To the best of our knowledge, our proposed approach is the first to give online algorithms with optimized competitive ratios with or without such constraints.

Our second key contribution (in Sec. IV) is to resolve a dilemma between the worst-case and average-case performance. Note that while our proposed Robust Affine Policy (RAP) is optimized for the worst-case competitive ratio, it may be too conservative and thus incur high costs for average-case inputs. Other heuristic algorithms (such as RHC, i.e., Receding Horizon Control [12], discussed in Sec. V) may perform well for the average case, but produce inferior competitive ratios for worst-case scenarios. Thus, an open question is whether one can get the best of both worlds. We address this dilemma by providing a "robustification" procedure. Given any online algorithm π_0 that is perceived to have good average-case performance, we intelligently combine π_0 with RAP to produce a new online algorithm with the same worst-case competitive ratio as RAP while still attaining comparable average-case performance to π_0. We note that this "robustification" idea was first introduced in our earlier work [14]. However, our OCO problem formulation is much more general, requiring a new robustification procedure to be developed. We use Network Functions Virtualization (NFV) [5] as a case study and simulate the robustified version of RHC. Our simulation results in Sec. V show that the robustified-RHC algorithm performs close to RHC when the uncertainty is low. When the uncertainty is high, the robustified-RHC algorithm performs significantly better, especially for worst-case inputs.

As discussed above, our work is related to robust optimization [13], [15], but differs in that we focus on a different objective of competitive ratios rather than worst-case costs. Our NFV case study is also related to the literature of NFV orchestration and scaling. However, most existing studies either assume a static model [7], [16]–[18], or provide heuristic online algorithms without any performance guarantees [19], [20]. The references [21] and [22] study online NFV orchestration and scaling, although they do not consider the distance cost (e.g., latency), which is an important cost component, especially when optimizing over multiple data centers. More recently, the reference [23] uses the regularization method to develop online algorithms for NFV orchestration and scaling over multiple data-centers. One of the key differences of our work is that we utilize partial future knowledge, i.e., in the form of an uncertainty set $\mathcal{U}$, to obtain potentially smaller competitive ratios. In contrast, it is unclear how to generalize the approaches in [21]–[23] to utilize such partial future knowledge. Further, in deriving their competitive ratios, the studies in [21] and [23] do not consider constraints on the number of servers available or ramp constraints on the rerouting decisions. NFV orchestration and scaling is also related to the FL (facility location) [24] and GAP (generalized assignment problem) [25], for which competitive online algorithms have been developed. However, in NFV, the demand fluctuates (both increases and decreases) over time in both the online and offline settings. In contrast, online FL and GAP problems usually assume that new demand is sequentially added over time towards a final offline setting where all demand is present. Further, the cost constraints of OCO problems are usually more general, e.g., involving switching costs and ramp constraints. Thus, it is unclear how to apply the competitive results from this literature to OCO and online NFV orchestration and scaling problems.

II. Problem Formulation

We now present our model for online convex optimization (OCO) problems with linear in-stage costs, switching costs and ramp constraints.

A. OCO with Linear In-Stage Costs

In the OCO problem that we consider, there are T rounds of decisions, $t = 1, 2, ..., T$. There is a cost-function $C_t(X(t), A(t))$ for each time t, which is a function of the input $A(t) \in \mathbb{R}^M$ (e.g., traffic load) revealed by the environment at time t, and the action $X(t) \in \mathbb{R}^N$ taken by the decision maker (e.g., system administrator) at time t. Throughout this paper, we assume that $C_t(\cdot, \cdot)$ is a linear function of $(X(t), A(t))$. Further, there is a switching cost $\beta|X(t) - X(t-1)|$ that penalizes the change of decision at time t (hence the transpose of β is a given vector in $\mathbb{R}^N$). The action $X(t)$ must be chosen to satisfy certain constraints. We assume that one set of constraints $\mathbb{X}_t(A(t))$, which may depend on the input $A(t)$, can be written as a linear inequality in $(X(t), A(t))$, i.e.,

$$D_1 X(t) + D_2 A(t) \leq 0, \ \forall \, t, \tag{1}$$

where the transposes of D_1 and D_2 are in $\mathbb{R}^N$ and $\mathbb{R}^M$, respectively. (As in Real Analysis, we use "$\forall$" for "for all", and use "$\exists$" for "there exists".) Further, there may be ramp constraints

$$|X_n(t) - X_n(t-1)| \leq \Delta_X^n, \ \forall \, n, t, \tag{2}$$

where X_n is the n-th element of X. As we will illustrate with a case study, this construction can model several types of costs and constraints. Let $A(t_1{:}t_2)$ denote the input sequence $A(t)$ from $t = t_1$ to t_2. Define $X(t_1{:}t_2)$ similarly.

At each time t, the environment reveals $A(t)$ first. Then the decision maker picks the action $X(t)$ and incurs the in-stage cost $C_t(X(t), A(t))$ and the switching cost $\beta|X(t)-X(t-1)|$. Note that although $C_t(\cdot, \cdot)$ is linear, the switching cost still makes the whole problem convex. Further, this problem is an online problem because the decision maker does not know the future values of $A(t+1), A(t+2), ..., A(T)$ when she makes the decision $X(t)$.

As we discussed in the introduction, the combination of the linear constraint (1) and the ramp constraint (2) may lead to infeasibility. If $X(t-1)$ is not properly chosen, the ramp constraint limits how far $X(t)$ can deviate from $X(t-1)$. Then, there may not exist a feasible point that simultaneously satisfies (1) and (2). For example, this infeasibility can occur when the demand increases suddenly and the traffic cannot be rerouted as quickly to serve the demand. Thus, a key contribution of our work is to be able to deal with cases with or without such "hard infeasibility constraints."

B. Uncertainty Set

Recall that the input $A(t)$ is unknown to the online algorithm until time t. Intuitively, if $A(t)$ can vary in arbitrary ways, one may have to take the most conservative decisions to avoid future infeasibility. Thus, in order to make the online decision problem practically more interesting, we introduce an uncertainty set to model the set of uncertain inputs that we care about. Specifically, we assume that the trajectory $A(1), A(2), ..., A(T)$ chosen by the environment must be from an uncertainty set $\mathcal{U}$. We expect that this uncertainty set $\mathcal{U}$ can be constructed from prediction and historical data [8], [14]. Next, we describe three ways (that can be used in combination) to formulate the uncertainty set $\mathcal{U}$.

(i) Day-ahead prediction: Let $A_{\text{DAP}}(1{:}T)$ denote a predicted trajectory of $A(1{:}T)$. We may assume that the real trajectory $A(1{:}T)$ must be within a neighborhood around $A_{\text{DAP}}(1{:}T)$,

$$A^m_{\text{lower}}(t) \leq A^m(t) \leq A^m_{\text{upper}}(t), \ \forall \, m, t, \tag{3}$$

where the upper/lower bounds are given by

$$A^m_{\text{upper}}(t) = (1 + \epsilon^m(t)) \times A^m_{\text{DAP}}(t), \ \forall \, m, t, \tag{4}$$

$$A^m_{\text{lower}}(t) = \max\{0, (1 - \epsilon^m(t)) \times A^m_{\text{DAP}}(t)\}, \ \forall \, m, t, \tag{5}$$

and $\epsilon^m(t)$ is the uncertainty level for time t.

(ii) Demand changing speed: Often, demand (e.g., traffic or renewable energy) may not change arbitrarily fast. We can model such knowledge by imposing

$$|A^m(t) - A^m(t-1)| \leq \Delta^m_{\text{A}}, \ \forall \, m, t. \tag{6}$$

(iii) The different elements of $A(t)$ may not hit the upper or lower bounds in (3) simultaneously. Thus, we can impose the following constraint (known as the "budget" constraint in the robust optimization literature [13, p. 47]),

$$\sum_{m=1}^{M} \frac{|A^m(t) - A^m_{\text{DAP}}(t)|}{\epsilon^m(t) \times A^m_{\text{DAP}}(t)} \leq \Gamma, \ \forall \, t. \tag{7}$$

Clearly, if $\Gamma = 0$, the uncertainty set only contains the day-ahead prediction $A_{\text{DAP}}(1{:}T)$. Thus, the model becomes deterministic. As Γ increases, more uncertainty will be considered.

The uncertainty set $\mathcal{U}$ that we use in this paper is specified by a combination of the above constraints. We note that the constraint (6) introduces temporal coupling of the inputs, which can be used to refine the near-term future uncertainty. Specifically, at any time t, $A(1{:}t)$ has already been revealed to the online algorithm. Thus, the future uncertainty remaining in the interval $[t+1, T]$ can be written as

$$\begin{aligned} \mathcal{U}_{A(1:t)} = \{A(t+1:T) \mid \exists A'(1{:}T) \in \mathcal{U}, \text{ such that,} \\ A'(1{:}t) = A(1{:}t), \ A'(t+1:T) = A(t+1:T)\}. \end{aligned} \tag{8}$$

C. The Performance Metric

As we discussed earlier, the total cost incurred by the decision maker is given by

$$\begin{aligned} C(X(0), X(1{:}T), A(1{:}T)) = \sum_{t=1}^{T} \{C_t(X(t), A(t)) \\ + \beta|X(t) - X(t-1)|\}. \end{aligned} \tag{9}$$

For an online algorithm π, at each time t the decision $X(t)$ can only be based on the already-known inputs $A(1{:}t)$ and knowledge about the future uncertainty given by (8). Let $C_\pi(A(1{:}T))$ be the total cost of algorithm π. We compare it with an offline solution that is assumed to know the entire input $A(1{:}T)$ ahead of time. We denote the cost of the optimal

offline solution as $C_{\text{OPT}}(A(1{:}T))$, which is the optimal value of the following optimization problem,

$$\min_{X(0:T):(1),(2)} C(X(0), X(1{:}T), A(1{:}T)). \tag{10}$$

Then, the competitive ratio of algorithm π, given by

$$\text{CR}_\pi \triangleq \max_{A(1:T)\in\mathcal{U}} \frac{C_\pi(A(1{:}T))}{C_{\text{OPT}}(A(1{:}T))}, \tag{11}$$

is the worst-case ratio between the online cost and the offline optimal cost, over all possible inputs from the uncertainty set.

We are thus interested in online solutions to OCO with small competitive ratios. Although the notions of uncertainty sets and affine policies (used later) are from the robust optimization literature [13], our objective in (11) is quite different. In the robust optimization literature, the objective is usually to minimize the worst-case (absolute) cost, i.e., $\max_{A(1:T)\in\mathcal{U}} C_\pi(A(1{:}T))$. Our objective of the competitive ratio, which is commonly used in the CS literature, instead focuses on a relative ratio comparing with the offline optimal solution. This difference leads to new technical difficulties in the optimization problem. In some way, competitive ratios can be viewed as less conservative than robust optimization because we do *not* only care about the worst-case cost.

D. A Case Study

We now use the Network Functions Virtualization (NFV) orchestration and scaling problem [5] as a case study to illustrate how our model can be used to study practical costs and constraints. We will also use it in our numerical evaluation in Sec. V. Nonetheless, the computational framework that we present later can be used for other problems, such as in power system operation [8].

We first follow the simplified NFV model in [7] (without precedence constraints). The network is modeled as a graph $G(V, E)$. There are S servers, each of capacity W_s occupying some node in V, $s = 1, 2, ..., S$. There are L clients, $l = 1, 2, ..., L$, each of which generates traffic at some node in V. Each client requires one or more Virtualized Network Functions (VNFs), e.g., firewalls, intrusion detection systems, caches and load balancers. Each VNF is indexed by a positive integer f in the set $\mathcal{F} = \{1, 2, ..., F\}$. We use $a_l^f(t)$ to denote the processing need for function f at time t due to the traffic generated by client l. Note that we can set $a_l^f(t) = 0$ if client l does not need function f.

Decisions and constraints: The NFV system administrator maps the VNFs to physical servers to serve the incoming demand. Let $y_s^f(t)$ denote the size (resource requirements) of VNF f at server s at time t. Let $x_{l,s}^f(t)$ denote the amount of demand from client l that is routed to server s for processing of function f at time t. The decision $X(t)$ in our general OCO model then corresponds to a vector $[x_{l,s}^f(t), y_s^f(t)]$. These actions must satisfy the following constraints. First, all the arriving demand needs to be served. Therefore, we must have

$$\sum_{s=1}^{S} x_{l,s}^f(t) \geq a_l^f(t),\ \forall\, l, f, t. \tag{12}$$

Second, when the demand arrives at the server s, the size of the corresponding functions that are placed on server s must be able to meet the demand. Assume that each unit-size function can serve one unit-size demand. Thus, we must satisfy

$$\sum_{l=1}^{L} x_{l,s}^f(t) \leq y_s^f(t),\ \forall\, s, f, t. \tag{13}$$

Third, resources on each server s must be able to support all functions placed there [26]. Assume that each unit-size function f needs w_s^f amount of resources at server s. Thus, we have

$$\sum_{f=1}^{F} y_s^f(t) w_s^f \leq W_s,\ \forall\, s, t. \tag{14}$$

The above constraints form the linear constraint set $\mathbb{X}_t(A(t))$.

In-stage Costs: There are costs for resource consumption of VNFs, and distance costs for routing the demand of clients, i.e.,

$$\Phi(X(1{:}T)) = \sum_{t=1}^{T}\sum_{s=1}^{S}\sum_{f=1}^{F} p_s^f y_s^f(t) + \sum_{t=1}^{T}\sum_{l=1}^{L}\sum_{s=1}^{S}\sum_{f=1}^{F} d_{l,s} x_{l,s}^f(t) \tag{15}$$

where p_s^f is the cost of hosting a unit of virtualized function f on server s, and $d_{l,s}$ is the distance (e.g., latency) cost of routing a unit of traffic load from client l to server s.

The model so far is similar to [7], which focuses on each snapshot in time, and assumes perfect knowledge of demand. In contrast, in this paper we wish to model decisions under uncertain and dynamic demand. Thus, we use an uncertainty set $\mathcal{U}$ as in Sec. II-B to model demand uncertainty. Note that a key benefit of NFV is that virtualized functions can be instantiated and torn down on-demand, and service can be moved across VNF instances as needed. However, such changes still incur non-negligible overhead. Specifically, because most VNFs are stateful, rerouting traffic from one VNF instance to another requires state migration [27]. Further, instantiating and tearing down VNFs incurs significant overhead. Therefore, we introduce the following additional costs and constraints.

Ramp Constraints: We may impose ramp constraints on the changes in routing decisions. If a certain amount of network bandwidth is reserved for state migration, rerouting cannot occur too fast. Such a constraint can be written as

$$|x_{l,s}^f(t+1) - x_{l,s}^f(t)| \leq \Delta_{X,l,s}^f,\ \forall\, l, s, f, t. \tag{16}$$

Switching/Migration Costs: There is overhead for state migration and changes in instantiation, i.e.,

$$\begin{aligned}\Psi(X(0), X(1{:}T)) = &\sum_{t=1}^{T}\sum_{l=1}^{L}\sum_{s=1}^{S}\sum_{f=1}^{F} \beta_{1,l,s}^f |x_{l,s}^f(t) \\ &- x_{l,s}^f(t-1)| + \sum_{t=1}^{T}\sum_{s=1}^{S}\sum_{f=1}^{F} \beta_{2,s}^f |y_s^f(t) - y_s^f(t-1)|,\end{aligned} \tag{17}$$

where $\beta_{1,l,s}^f$ and $\beta_{2,s}^f$ are cost parameters for state migration in rerouting and for the overhead in instantiating and tearing-down VNF instances, respectively. (Note that the absolute

values $|x|$ may also be replaced by $\max\{x, 0\}$ and one can still apply the proposed methodology in this paper.)

Hence, in our online NFV orchestration and scaling problem, the cost function is the sum of (15) and (17). Note that our model is more general than [21]–[23] because we simultaneously consider the distance costs, constraints on the available server capacity, and the ramp constraints, when we study the competitive performance.

III. A Computationally Tractable Framework

In this section, we introduce a computationally tractable framework to attain small competitive ratios for general OCO problems with linear in-stage costs, switching costs and ramp constraints. Note that a key benefit for online decisions is that the decision at time t can be adjusted based on the new input $A(t)$ just revealed. Thus, they can be more efficient than classical robust optimization where all decisions must be made ahead of time. This adjustability is similar to the adjustable robust optimization [13, p. 355], [15], where some decisions are made in a "wait-and-see" manner. However, searching among all possible online decisions is usually an intractable problem: each action $X(t)$ can be an arbitrary function of the past $A(1{:}t)$, and searching over such a functional space is very difficult when the problem size is large [13, p. 363], [15]. Instead, next we will borrow the idea of affine policies from [13, p.368] and [15] in order to obtain a computationally tractable framework.

Specifically, in our proposed robust affine policy (RAP), we restrict $X(t)$ to be an affine function of $A(t)$, i.e.,

$$X(t) = \eta(t) + H(t)A(t),\ \forall\, t, \tag{18}$$

where $\eta(t) \in \mathbb{R}^N$ and $H(t) \in \mathbb{R}^{N\times M}$ are determined before hand. Note that once $\eta(t)$ and $H(t)$ are determined, the online decision (18) becomes extremely simple. Instead, the complexity moves to the pre-calculation of $\eta(t)$ and $H(t)$ based on the knowledge of the uncertainty set $\mathcal{U}$.

Given $\eta(t)$ and $H(t)$, the cost of the online decisions can be readily calculated as

$$C_{\text{RAP}}(A(1{:}T)|X(0), \boldsymbol{\eta}, \mathbf{H}) = C(X(0), X(1{:}T), A(1{:}T)),$$

where $X(1{:}T)$ is given by (18). However, the online decisions must still satisfy both (1) and (2). In other words, we need that

$$(1),\ (2) \text{ hold for all } A(1{:}T) \in \mathcal{U},\ \text{given (18)}. \tag{19}$$

We can thus formulate the optimization problem for minimizing the competitive ratio as

$$\min_{\{X(0),\boldsymbol{\eta},\mathbf{H}:(19)\}} \max_{A(1:T)\in\mathcal{U}} \frac{C_{\text{RAP}}(A(1{:}T)|X(0),\boldsymbol{\eta},\mathbf{H})}{C_{\text{OPT}}(A(1{:}T))}. \tag{20}$$

Although affine policies have been used in [13, p. 368] and [15], using a similar approach as in the optimization problem (20) introduces new technical difficulties. Note that for each $A(1{:}T) \in \mathcal{U}$, the ratio $\frac{C_{\text{RAP}}(A(1:T)|X(0),\boldsymbol{\eta},\mathbf{H})}{C_{\text{OPT}}(A(1:T))}$ is convex in $\boldsymbol{\eta}$, $\mathbf{H}$ and $X(0)$. Thus, the inner maximization produces a convex objective in $\boldsymbol{\eta}$, $\mathbf{H}$, $X(0)$ for outer minimization. However, it is unclear how to solve the inner maximization problem itself because it involves a ratio of convex functions. Next, we will show step-by-step how to optimize an upper bound of (20) via a tractable convex optimization problem.

Step-1: Even without considering the ratio, the numerator $C_{\text{RAP}}(A(1{:}T)|X(0),\boldsymbol{\eta},\mathbf{H})$ in (20) is convex in $A(1{:}T)$. Maximizing a convex function is in general intractable. We resolve this issue by introducing a linear upper bound on $C_{\text{RAP}}(A(1{:}T)|X(0),\boldsymbol{\eta},\mathbf{H})$ (see also [28, p. 228]). Specifically, note that the only non-linearity in $C_{\text{RAP}}(A(1{:}T)|X(0),\boldsymbol{\eta},\mathbf{H})$ is from the switching cost $\beta|X(t)-X(t-1)| = \beta|\eta(t) + H(t)A(t) - \eta(t-1) - H(t-1)A(t-1)|$. We now introduce a new variable $\mu(t) \in \mathbb{R}^N$ that upper-bounds each element of this switching cost for all $A(1{:}T) \in \mathcal{U}$, i.e.,

$$\left.\begin{aligned} \mu(t) &\geq \beta|\eta(t) + H(t)A(t) - \eta(t-1) \\ &\quad - H(t-1)A(t-1)|,\ \forall\, t > 1 \\ \mu(1) &\geq \beta|\eta(t) + H(t)A(t) - X(0)| \end{aligned}\right\} \forall_{\in\mathcal{U}.}^{A(1:T)} \tag{21}$$

Let $\tilde{C}_{\text{RAP}}(A(1{:}T)|\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu}) = \sum_{t=1}^{T}\{C_t(X(t),A(t)) + \mu(t)\}$, where $X(t)$ is given by (18). Then, $C_{\text{RAP}}(A(1{:}T)|X(0),\boldsymbol{\eta},\mathbf{H}) \leq \tilde{C}_{\text{RAP}}(A(1{:}T)|\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu})$ for all $A(1{:}T) \in \mathcal{U}$. Hence, we can obtain an upper bound of (20) by solving the following optimization problem instead

$$\min_{\{X(0),\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu}:(19),(21)\}} \max_{A(1:T)\in\mathcal{U}} \frac{\tilde{C}_{\text{RAP}}(A(1{:}T)|\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu})}{C_{\text{OPT}}(A(1{:}T))}. \tag{22}$$

Note that the numerator is now a linear function in $A(1{:}T)$.

Step-2: The ratio in the inner maximization problem in (22) is usually not a concave function of $A(1{:}T)$. Thus, it is still not obvious how to maximize the ratio. Using the following lemma from our earlier work [14], we now show that this inner maximization problem can be converted to an equivalent convex problem.

Lemma 1: For fixed $\mathbf{B} \in \mathbb{R}^{M\times N}$, $b \in \mathbb{R}^M$, $c \in \mathbb{R}^{1\times N}$ and $\alpha \in \mathbb{R}$, suppose that the following conditions are simultaneously satisfied:

(a) $f(x)$ is a convex function of $x \in \mathbb{R}^N$;

(b) $f(x) > 0$ in the constrained region of $\mathbf{B}x \leq b$;

(c) There exists x satisfying $\mathbf{B}x \leq b$ and $cx + \alpha > 0$.

Then, $\sup_{x,y}\{\frac{cx+\alpha}{y} : y = f(x), \mathbf{B}x \leq b\} = \sup_{x',u}\{cx' + \alpha u : 1 \geq uf(\frac{x'}{u}), \mathbf{B}x' \leq bu, u > 0\}$.

A detailed proof is given in [29]. Note that the second supremum is a convex problem because $uf(\frac{x}{u})$ is a convex function whenever $f(x)$ is convex [30, p. 89]. The result of this lemma is somewhat similar to the convex transformation of linear-fractional program [30, p. 89]. However, here the denominator is non-linear, and thus Lemma 1 is more general.

We now verify that the conditions of Lemma 1 hold for (22). For condition (a), we note that $C_{\text{OPT}}(A(1{:}T))$ is the minimum of a convex function $C(X(0), X(1{:}T), A(1{:}T))$ over $X(0{:}T)$ in a convex set. Thus, $C_{\text{OPT}}(A(1{:}T))$ is a convex function of $A(1{:}T)$ [30, p. 87]. For conditions (b) and (c), $C_{\text{OPT}}(A(1{:}T))$ and $\tilde{C}_{\text{RAP}}(A(1{:}T)|\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu})$ are both positive, so these conditions trivially hold. Hence, based on Lemma 1, we can convert the inner maximization of (22) to an equivalent

convex optimization problem. We note that although this transformation has been used in [14], the *step-1* from (20) to (22) is also crucial because otherwise the numerator of (20) is not linear and thus Lemma 1 cannot be applied.

Step-3: Note that the inner maximization of (22) can be converted to a convex program, we can then focus on the outer minimization. As we discussed earlier, the objective of the outer minimization is convex in $\boldsymbol{\eta}$, $\mathbf{H}$ and $X(0)$. It remains to check its constraints. These constraints are of the form that some inequalities must hold for all $A(1{:}T) \in \mathcal{U}$. It turns out that these constraints are also convex in $\boldsymbol{\eta}$, $\mathbf{H}$ and $X(0)$, and can be converted to linear constraints (See [15] for related techniques). We take one part of the constraint (19) as an example. Note that by (19), the linear inequality (1) must hold for all $A(1{:}T) \in \mathcal{U}$. For any $\boldsymbol{\eta}$, $\mathbf{H}$, the inequality (1) for each t becomes,

$$\begin{aligned} &D_1[\eta(t) + H(t)A(t)] + D_2 A(t) \le 0,\ \forall\ A(1{:}T) \in \mathcal{U} \\ &\quad \Leftrightarrow \max_{A(1:T)\in\mathcal{U}} \{D_1[\eta(t) + H(t)A(t)] + D_2 A(t)\} \le 0 \\ &\quad \Leftrightarrow \max_{A(1:T)\in\mathcal{U}} \{[D_1 H(t) + D_2]A(t)\} \le -D_1\eta(t). \end{aligned} \tag{23}$$

Note that $A(1{:}T) \in \mathcal{U}$ can be written as a set of linear constraints[1], thus the left-hand-side of (23) is of the form $\max C^T A$, subject to $EA \le b$ where A corresponds to $A(1{:}T)$, and $\boldsymbol{\eta}$, $\mathbf{H}$ enter into the matrix C. By duality [30],

$$\max_{EA\le b} C^T A = \min_{E^T\lambda \ge C} b^T\lambda, \tag{24}$$

where C^T, b^T, E^T are the transposes of C, b, E, respectively. Thus, (23) is equivalent to: there exists λ such that

$$\begin{cases} b^T\lambda \le -D_1\eta(t) \\ E^T\lambda \ge C \end{cases}, \tag{25}$$

which is a convex constraint in λ, $\eta(t)$ and C (i.e., $\boldsymbol{\eta}$, $\mathbf{H}$). (See our technical report [29] for further details.)

In summary, through the above three steps, we have obtained a convex problem (22), which can be effectively solved to obtain $\boldsymbol{\eta}$ and $\mathbf{H}$. (See our technical report [29] for details of the optimization algorithms that we used.) Let $\overline{\mathrm{CR}}$ be the optimal value of (22). Then, the competitive ratio of RAP (18) based on the optimal $\boldsymbol{\eta}$ and $\mathbf{H}$ is no larger than $\overline{\mathrm{CR}}$.

Remark: Although the solution approach in this section assumes continuous decision variables, it can be generalized to deal with certain integer constraints. Due to page limits, we refer readers to [29] for details.

IV. Algorithm Robustification

In Sec. III, we developed a tractable computational framework to calculate an optimized competitive ratio $\overline{\mathrm{CR}}$ among the class of affine policies. Let the corresponding robust affine policy (RAP) be denoted by π^*_{RAP}, which will attain a competitive ratio no larger than $\overline{\mathrm{CR}}$. However, as is often the case with competitive online algorithms in the literature, the policy π^*_{RAP} may be too conservative in nature. For example, consider the scenario where the uncertainty set $\mathcal{U}$ is given by a predicted input trajectory plus/minus possible errors. In order for π^*_{RAP} to attain the competitive ratio $\overline{\mathrm{CR}}$, it must "defend" against the worst case where the input is far away from the prediction. Specifically, it may have to over-provision resources. As a result, if the input is actually very close to the prediction (which usually corresponds to a larger probability mass on average), π^*_{RAP} may incur a higher cost than necessary. In contrast, a popular method in the literature to deal with sequential decisions under uncertainty is RHC (Receding Horizon Control) [12]. At each time t, RHC assumes that the future demand is exactly the same as the most-recent near-term prediction, which is based on revealed demand, day-ahead prediction and possible constraints, e.g., (3), (6), (7). Then, RHC minimizes the cost over the entire future horizon and commits to the first decision $X(t)$. Intuitively, if the input is close to the prediction (which we refer to as the average case), RHC may actually perform very well. The problem, of course, is that RHC cannot guarantee as low a competitive ratio as $\overline{\mathrm{CR}}$. In summary, we see a dilemma between worst-case and average-case performance. In this section, we will address this dilemma by significantly generalizing the "robustification" procedure of our earlier work [14] to obtain good performance for both worst-case and average-case inputs.

In our proposed robustification procedure, we begin with an online algorithm π_0 that is believed to achieve good average-case performance (e.g., π_0 could be a variant of RHC from [12]). We are also given the competitive ratio $\overline{\mathrm{CR}}$, which is optimized among the class of affine policies as in Sec. III. We aim to produce a new online policy π that attains comparable average-case performance as π_0, but at the same time the worst-case competitive ratio $\overline{\mathrm{CR}}$. Our basic idea for this new policy π is to follow the decisions of π_0 as much as possible, unless doing so will violate the competitive ratio $\overline{\mathrm{CR}}$. Our first step is thus to develop a way to check whether the decision of π_0 will violate the competitive ratio $\overline{\mathrm{CR}}$.

Toward this end, let us focus on a time t. Note that the decisions of this algorithm π before time t have already been made. The algorithm π_0 now produces a decision $X_{\pi_0}(t)$ for time t. In order to verify whether this new decision will still attain $\overline{\mathrm{CR}}$, we need to check whether the following holds:

$$\begin{aligned} C_\pi(A(1{:}T)) &\le \overline{\mathrm{CR}} \cdot C_{\mathrm{OPT}}(A(1{:}T)), \\ &\forall\ A(t+1:T) \in \mathcal{U}_{A(1:t)}, \end{aligned} \tag{26}$$

where $\mathcal{U}_{A(1:t)}$ is given in (8). Let $C_\pi(A(1:t-1))$ denote the past cost of the online algorithm π from time 1 to $t-1$, excluding the switching cost from time $t-1$ to t. (Again, this cost is known at time t, regardless of whether or not π has followed π_0 before time t.) Based on the decision $X_{\pi_0}(t)$ of algorithm π_0, let $C_{\pi_0}(A(t)) = C_t(X_{\pi_0}(t), A(t)) + \beta|X_{\pi_0}(t) - X_\pi(t-1)|$. Further, let $C_\pi(A(t+1:T))$ denote the future cost from time $t+1$ to time T (including the switching cost from $X_{\pi_0}(t)$ to $X_\pi(t+1)$). Then, because we want to follow

[1]Note that $\mathcal{U}$ may involve absolute values, but can still be converted to a linear form (see [29] for details).

the decision $X_{\pi_0}(t)$, (26) is equivalent to checking

$$C_\pi(A(1:t-1)) + C_{\pi_0}(A(t)) + C_\pi(A(t+1:T)) - \overline{\text{CR}} \cdot C_{\text{OPT}}(A(1{:}T)) \leq 0, \forall A(t{+}1:T) \in \mathcal{U}_{A(1:t)}. \quad (27)$$

However, the difficulty of checking (27) is that not only the future input has not been revealed yet, we do not even know what decision the algorithm π will take on these future inputs! To circumvent this difficulty, we estimate $C_\pi(A(t+1:T))$ based on affine policies. In this case, the affine policy can be written as

$$X(t') = \eta(t') + H(t')A(t'),\ t' = t+1, t+2, ..., T. \quad (28)$$

Note that in general this pair of ($\boldsymbol{\eta}$, $\mathbf{H}$) may be different from those calculated in the previous section. Let $C_{\text{RAP}}(A(t+1:T)|\boldsymbol{\eta}, \mathbf{H})$ denote the future cost $C_\pi(A(t+1:T))$ if π follows the affine policy (28). We can then formulate the following optimization problem:

$$\zeta'_{1,t} \triangleq \min_{\substack{\{\boldsymbol{\eta},\mathbf{H}:(19) \text{ restricted} \\ \text{to } A(t+1:T)\in\mathcal{U}_{A(1:t)}\}}} \max_{\substack{A(t+1:T) \\ \in\mathcal{U}_{A(1:t)}}} \{C_\pi(A(1:t-1)) + C_{\pi_0}(A(t)) + C_{\text{RAP}}(A(t+1:T)|\boldsymbol{\eta}, \mathbf{H}) - \overline{\text{CR}} \cdot C_{\text{OPT}}(A(1{:}T))\}. \quad (29)$$

Similar to *Step-1* of Sec. III, (29) may be intractable because the maximization part is a non-convex problem. Nonetheless, we can use the technique in *Step-1* of Sec. III [see (21)] to introduce a new set of variables $\boldsymbol{\mu}$ that upper-bounds the switching costs in (29). In this way, we can obtain an upper bound of $\zeta'_{1,t}$ via a convex program, given by

$$\zeta_{1,t} \triangleq \min_{\substack{\{\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu}:(19),(21) \\ (31) \text{ restricted to} \\ A(t+1:T)\in\mathcal{U}_{A(1:t)}\}}} \max_{\substack{A(t+1:T) \\ \in\mathcal{U}_{A(1:t)}}} \{C_\pi(A(1:t-1)) + C_{\pi_0}(A(t)) + \tilde{C}_{\text{RAP}}(A(t+1:T)|\boldsymbol{\eta}, \mathbf{H}, \boldsymbol{\mu}) - \overline{\text{CR}} \cdot C_{\text{OPT}}(A(1{:}T))\} \quad (30)$$

where $\tilde{C}_{\text{RAP}}(A(t{+}1{:}T)|\boldsymbol{\eta}, \mathbf{H}, \boldsymbol{\mu}) = \sum_{t'=t+1}^{T}\{C_{t'}(X(t'), A(t')) + \mu(t')\}$, $X(t')$ is given by (28), and the additional constraint (31) is given below by

$$\mu(t+1) \geq \beta|\eta(t+1) + H(t+1)A(t+1) - X_{\pi_0}(t)| \quad \forall A(t+1) \in \mathcal{U}_{A(1:t)}. \quad (31)$$

Thus, if $\zeta_{1,t} \leq 0$, we can be assured that following the decision of algorithm π_0 at time t will retain the same competitive ratio $\overline{\text{CR}}$.

We still need to determine what to do if $\zeta_{1,t} > 0$. In that case, we no longer follow the decision of algorithm π_0 at time t. Instead, we find $X(t)$ as well as a different affine policy based on the following optimization problem

$$\zeta_{2,t} \triangleq \min_{\substack{\{X(t),\boldsymbol{\eta},\mathbf{H},\boldsymbol{\mu}:(19), \\ (21) \text{ restricted to} \\ A(t+1:T)\in\mathcal{U}_{A(1:t)}\}}} \max_{\substack{A(t+1:T) \\ \in\mathcal{U}_{A(1:t)}}} \{C_\pi(A(1:t-1)) + C_t(X(t), A(t)) + \beta|X(t) - X_\pi(t-1)| + \tilde{C}_{\text{RAP}}(A(t+1{:}T)|\boldsymbol{\eta}, \mathbf{H}, \boldsymbol{\mu}) - \overline{\text{CR}} \cdot C_{\text{OPT}}(A(1{:}T))\}. \quad (32)$$

Note that as long as $\zeta_{2,t} \leq 0$, we can use the decision $X(t)$ from the optimization problem (32) at time t, and we are assured that following the corresponding affine policy will attain the competitive ratio $\overline{\text{CR}}$. Thus, at time t, if $\zeta_{1,t} > 0$, our new algorithm π will follow $X(t)$ from the optimization problem (32). The detailed robustification procedure is shown in Algorithm 1.

Algorithm 1 Robustification Procedure

Input: $\overline{\text{CR}}$, $\mathcal{U}$, and Algorithm π_0
Output: π: Robustified version of π_0
FOR $t = 1:T$
Update $\mathcal{U}_{A(1:t)}$, $C_\pi(A(1:t-1))$, $X_{\pi_0}(t)$ and $C_{\pi_0}(A(t))$.
Solve (30) to get $\zeta_{1,t}$
if $\zeta_{1,t} \leq 0$ **then**
  $X_\pi(t) \leftarrow X_{\pi_0}(t)$
else
  Solve (32) to get a new optimal $X(t)$
  $X_\pi(t) \leftarrow X(t)$
end if
END

Intuitively, if we can show that $\zeta_{2,t} \leq 0$ at all time t, then Algorithm 1 will attain the same competitive ratio $\overline{\text{CR}}$. This can be shown by induction: if at the previous time we already verify $\zeta_{1,t-1} \leq 0$, it implies that the corresponding affine policy should make the objective of (32) ≤ 0. Thus, optimizing (32) will only produce an even lower $\zeta_{2,t} \leq 0$. We then obtain the following result.

Theorem 1: Algorithm π is $\overline{\text{CR}}-$competitive, that is, for all $A(1:T) \in \mathcal{U}$,

$$C_\pi(A(1:T)) \leq \overline{\text{CR}} \cdot C_{\text{OPT}}(A(1:T)). \quad (33)$$

The proof is available in our technical report [29].

V. Simulation Results

In this section, we use the NFV orchestration and scaling problem in Sec. II-D as a case study to evaluate the performance of our proposed competitive online algorithm. Since there is little public data on NFV topologies and traces, we use synthetic scenarios. There are 100 clients ($L = 100$), 10 servers ($S = 10$), and 10 different VNFs ($F = 10$). Other parameters are listed in Table I.[2] In Table I, $U[a, b]$ denotes the uniform distribution in the interval $[a, b]$.

We will compare our proposed online algorithm π with both the offline optimal solution and RHC (Receding Horizon Control). Note that [11] proposes an extended enhanced version of RHC, called AFHC (Averaging Fixed Horizon Control), which is shown to attain a bounded competitive difference. However, AFHC assumes a perfect look-ahead window with size ω, i.e, at each time t, the immediate future inputs in a window of size ω is assumed to be known. In practice such a

[2]Note that with container technology (such as Docker [6]), the granularity of VNF resource allocation is much finer [6] than with VMs. Hence, we use a large value of W_s for the server capacity.

TABLE I: Simulation Parameters.

Time horizon	T	24
Resources needed for a unit of VNF f on server s	w_s^f	U[0,1]
Resources available on server s	W_s	5000
Distance cost of routing a unit of traffic load from client l to server s	$d_{l,s}$	U[0,10]
Cost of host resources consumed by a unit of VNF f on server s	p_s^f	U[0,50]
Uncertainty level	$\epsilon_l^f(t)$	0.2
Parameter of the ramp constraint	$\Delta_{X,l,s}^f$	U[1,5]
Parameter of the demand changing speed	$\Delta_{A,l}^f$	20
Parameter of the budget constraint	Γ	15

perfect look-ahead window is often not available. Hence, we do not compare with AFHC with $\omega > 1$. On the other hand, RHC may be viewed as a special case of AFHC with $\omega = 1$. RHC is often found to exhibit good average-case performance [12]. Thus, we can use it as the online algorithm π_0, which we will robustify as in Sec. IV.

In order to evaluate these solutions, we generate random demand as follows. First, we generate the predicted trajectory $A_{\text{DAP}}(1{:}T)$. For each time t, $A_{\text{DAP}}(1{:}T)$ is uniformly distributed in a given set (see [29] for details). Second, we generate the uncertain demand $A'_{\text{real}}(1{:}T)$ around $A_{\text{DAP}}(1{:}T)$ by adding i.i.d white Gaussian noise with variance $\sigma_l^f(t)^2 = (\epsilon_l^f(t) \cdot A_{\text{DAP},l}^f(t) \cdot \rho)^2$, where we call ρ the "variability" of the demand sequence. When ρ is large, the demand sequence fluctuates more significantly in time and is more likely to hit the extreme cases in the uncertainty set $\mathcal{U}$. On the other hand, the uncertainty set $\mathcal{U}$ is given independent of ρ, and hence the value of $\overline{\text{CR}}$ is also independent of ρ. Thus, the demand trajectory $A'_{\text{real}}(1{:}T)$ may not always obey the constraints imposed on the uncertainty set $\mathcal{U}$, including (3), (6), (7). Hence, we further adjust it to obtain the "real" demand iteratively in time. Specifically, for each t, if $A'_{\text{real}}(t)$ (along with $A_{\text{real}}(t-1)$) does not satisfy the constraints of $\mathcal{U}$, we change it to the closest value that satisfies the constraints of $\mathcal{U}$. (See our technical report [29] for details.)

Note that the offline optimal solution is assumed to know $A_{\text{real}}(1{:}T)$ in advance. In contrast, at each time t, our proposed policies and RHC only know the real demand up to time t, as well as the predicted demand for $t+1$, ..., T. As we mentioned in Sec. I, RHC may cause infeasibility due to the hard feasibility constraints. Here, whenever RHC finds no feasible solution at time t, we allow it to violate the ramp constraint (2) by paying another high penalty of 10^3 for each unit of violation of the ramp constraint. Note that our proposed algorithms never pay this penalty because they always respect the ramp constraints.

In Fig. 1, we report the competitive ratio $\overline{\text{CR}}$ of our proposed robust affine policy π^*_{RAP}. As we vary the uncertainty level ϵ of $\mathcal{U}$, the competitive ratio increases almost linearly. Note that even when $\epsilon = 0.6$, i.e., the real demand may vary 60% from the predicted value, the competitive ratio is around 2.5, which is relatively small.

The value of $\overline{\text{CR}}$ reported above is the theoretical upper bound of the competitive ratio over all inputs. We also collect the empirical competitive ratio (ECR) under the random demand trajectory that we generated, which is the ratio of the total cost of an online algorithm π to that of the optimal offline solution for each generated trajectory.

We plot in Fig. 2 the empirical CDF (Cumulative Distribution Function) of the value of ECR over 20 trials, where ECR_{RAP}, ECR_{RHC} and $\text{ECR}_{\text{Robustified-RHC}}$ correspond to RAP, RHC and the robustified version of RHC, respectively. We first observe that at all values of variability ρ, the ECR of our proposed robustified-RHC algorithm never exceeds the value of $\overline{\text{CR}}$ (which is 1.2786 when $\epsilon = 0.2$). In contrast, the ECR of RHC increases as ρ increases, and it exceeds $\overline{\text{CR}}$ for a significant fraction of trials when $\rho = 1$ and $\rho = 20$. Clearly, RHC fails to control the worst-case competitive ratio when the future demand is highly variable. Specifically, due to the ramp constraint, such high variability may lead to infeasibility for RHC, which produces the high online costs. On the other hand, π^*_{RAP} incurs much higher ECR than both robustified-RHC and RHC when $\rho = \frac{1}{30}$. This suggests that RAP is too conservative when the prediction turns out to be quite accurate. In summary, what is particularly appealing for our proposed robustified-RHC policy is that it not only attains much smaller ECR than RHC when ρ is large, but also attains almost the same performance as RHC when ρ is small.

In Fig. 3, we plot how $\overline{\text{CR}}$ varies with the magnitude of switching costs. Intuitively, when the parameter β of the switching cost is large, the online decisions become more difficult, and thus the competitive ratio will increase. This is shown in Fig. 3. For example, when $\beta = 10$, Fig. 3a shows that the total amount of switching cost (17) is almost 60% of the in-stage cost (15). From Fig. 3b, the corresponding competitive ratio increases to about 3. However, when β further increases, the switching cost dominates (See Fig. 3a), and our competitive ratio further increases. We note that for certain online problems (e.g., [31]), constant competitive ratios may be obtained even when the switching cost is arbitrarily high. This suggests that there may be room to improve our online algorithms, e.g., by integrating ideas from ski-rental problems [2], [31], to obtain even better competitive ratios.

VI. Conclusion

We study competitive online algorithms for OCO problems with linear in-stage costs, switching costs and ramp constraints. First, we present a powerful computational framework to obtain an optimized competitive ratio given an uncertainty set. Second, we provide a robustification procedure to obtain robustified online algorithms with both good average-case performance and an optimized competitive ratio. We demonstrate the power of our proposed approach through a case study for NFV. The robustified version of a popular heuristic algorithm RHC is shown to attain good performance for both average-case and worst-case inputs. For future work, we plan to study matching lower bounds for the optimal competitive ratio and compare that with ours. For NFV, we are also interested

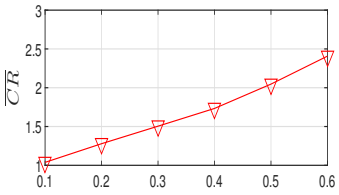
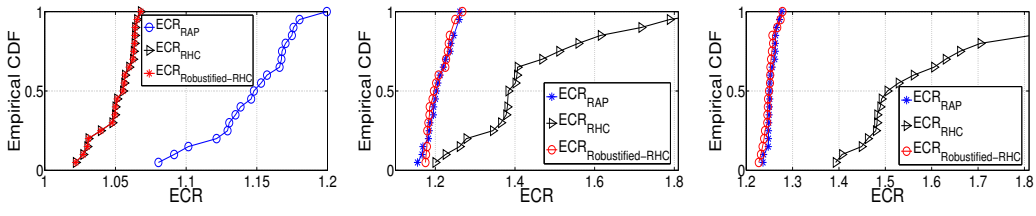
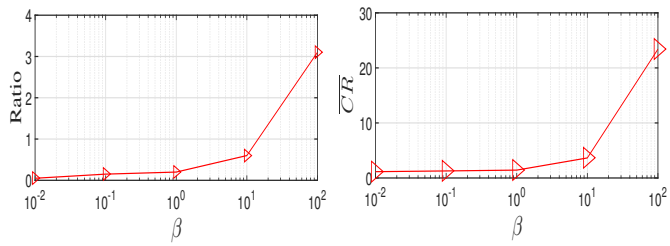


Fig. 1: The competitive Ratio.



(a) $\rho = 1/30$ (b) $\rho = 1$ (c) $\rho = 20$

Fig. 2: Empirical CDF of ECR (Empirical Competitive Ratio)



(a) Ratio: Switching Cost/In-stage Cost (b) Competitive Ratio $\overline{\text{CR}}$

Fig. 3: Impact of Switching Cost Parameter β.

in generalizing our methodology to incorporate precedence constraints [21], [22].

Acknowledgements: This work has been partially supported by the National Science Foundation (grants CNS-1457137, ECCS-1509536, OAC-1738981 and CNS-1717493) and the Army Research Office (grant W911NF-14-1-0368).

REFERENCES

[1] S. Shalev-Shwartz, "Online Learning and Online Convex Optimization," *Foundations and Trends® in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[2] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, "Dynamic Right-Sizing for Power-Proportional Data Centers," *IEEE/ACM Transactions on Networking (TON)*, vol. 21, no. 5, pp. 1378–1391, 2013.

[3] Z. Liu, I. Liu, S. Low, and A. Wierman, "Pricing Data Center Demand Response," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 1, pp. 111–123, 2014.

[4] S.-J. Kim and G. Giannakis, "An Online Convex Optimization Approach to Real-Time Energy Pricing for Demand Response," *IEEE Transactions on Smart Grid*, 2016.

[5] "ETSI Network Functions Virtualisation (NFV) Architectural Framework," http://www.etsi.org/deliver/etsi$_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf.

[6] https://www.docker.com/.

[7] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near Optimal Placement of Virtual Network Functions," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, 2015, pp. 1346–1354.

[8] S. Zhao, X. Lin, D. Aliprantis, H. N. Villegas, and M. Chen, "Online Multi-Stage Decisions for Robust Power-Grid Operations under High Renewable Uncertainty," in *IEEE Conference on Computer Communications (INFOCOM)*, 2016, pp. 1–9.

[9] E. Hazan, A. Agarwal, and S. Kale, "Logarithmic Regret Algorithms for Online Convex Optimization," *Machine Learning*, vol. 69, no. 2, pp. 169–192, 2007.

[10] M. J. Neely and H. Yu, "Online Convex Optimization with Time-Varying Constraints," *arXiv preprint arXiv:1702.04783*, 2017.

[11] M. Badiei, N. Li, and A. Wierman, "Online Convex Optimization with Ramp Constraints," in *IEEE 54th Annual Conference on Decision and Control (CDC)*, 2015, pp. 6730–6736.

[12] N. Chen, J. Comden, Z. Liu, A. Gandhi, and A. Wierman, "Using Predictions in Online Optimization: Looking Forward with an Eye on the Past," in *Proceedings of the ACM SIGMETRICS*, 2016, pp. 193–206.

[13] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*. Princeton University Press, 2009.

[14] S. Zhao, X. Lin, and M. Chen, "Peak-Minimizing Online EV Charging: Price-of-Uncertainty and Algorithm Robustification," in *IEEE Conference on Computer Communications (INFOCOM)*, 2015, pp. 2335–2343.

[15] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski, "Adjustable Robust Solutions of Uncertain Linear Programs," *Mathematical Programming*, vol. 99, no. 2, pp. 351–376, 2004.

[16] A. Gupta, M. F. Habib, P. Chowdhury, M. Tornatore, and B. Mukherjee, "Joint Virtual Network Function Placement and Routing of Traffic in Operator Networks," *UC Davis, Davis, CA, USA, Tech. Rep*, 2015.

[17] Y. Sang, B. Ji, G. R. Gupta, X. Du, and L. Ye, "Provably Efficient Algorithms for Joint Placement and Allocation of Virtual Network Functions," *arXiv preprint arXiv:1702.01154*, 2017.

[18] H. Moens and F. De Turck, "VNF-P: A Model for Efficient Placement of Virtualized Network Functions," in *IEEE 10th International Conference on Network and Service Management (CNSM)*, 2014, pp. 418–423.

[19] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "On Orchestrating Virtual Network Functions," in *IEEE 11th International Conference on Network and Service Management (CNSM)*, 2015.

[20] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic Virtual Network Function Placement," in *IEEE 4th International Conference on Cloud Networking (CloudNet)*, 2015, pp. 255–260.

[21] X. Zhang, C. Wu, Z. Li, and F. C. Lau, "Proactive VNF Provisioning with Multi-timescale Cloud Resources: Fusing Online Learning and Online Optimization," in *IEEE Conference on Computer Communications (INFOCOM)*, 2017.

[22] X. Wang, C. Wu, F. Ley, and F. C. Lau, "Online Learning-Assisted VNF Service Chain Scaling with Network Uncertainties," in *IEEE 10th International Conference on Cloud Computing*, 2017.

[23] Y. Jia, C. Wu, Z. Li, F. Le, and A. Liu, "Online scaling of nfv service chains across geo-distributed datacenters," *arXiv preprint arXiv:1611.08086*, 2016.

[24] D. Fotakis, "On the Competitive Ratio for Online Facility Location," in *ICALP*. Springer, 2003, pp. 637–652.

[25] S. Alaei, M. Hajiaghayi, and V. Liaghat, "The Online Stochastic Generalized Assignment Problem," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2013, pp. 11–25.

[26] L. Cao, P. Sharma, S. Fahmy, and V. Saxena, "NFV-VITAL: A Framework for Characterizing the Performance of Virtual Network Functions," in *Proceedings of IEEE NFV-SDN*, 2015, p. 7.

[27] Y. Wang, G. Xie, Z. Li, P. He, and K. Salamatian, "Transparent Flow Migration for NFV," in *IEEE 24th International Conference on Network Protocols (ICNP)*, 2016, pp. 1–10.

[28] E. F. Camacho and C. B. Alba, *Model Predictive Control*. Springer Science & Business Media, 2013.

[29] M. Shi, X. Lin, S. Fahmy and D.-H. Shin, "Competitive Online Convex Optimization with Switching Costs and Ramp Constraints," Purdue University, Tech. Rep., 2017. Available at https://engineering.purdue.edu/%7elinx/papers.html.

[30] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[31] L. Lu, J. Tu, C.-K. Chau, M. Chen, and X. Lin, "Online Energy Generation Scheduling for Microgrids with Intermittent Energy Sources and Co-Generation," in *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*, 2013, pp. 53–66.