Online Cloud Resource Provisioning Under Cost Budget for QoS Maximization

Yu Liu*, Niangjun Chen†, Zhenhua Liu‡, and Yuanyuan Yang*
*Department of Electrical and Computer Engineering, Stony Brook University, USA
†Singapore University of Technology and Design, Singapore
‡Department of Applied Mathematics and Statistics, Stony Brook University, USA

2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS) | 978-1-6654-1494-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/IWQOS52092.2021.9521347

***Abstract*—Cloud computing is becoming one of the ubiquitous computing paradigms for enterprises and organizations in recent years. Due to the volatility of system states such as cloud resource price and workload demand, it is challenging to provision cloud resources efficiently. This paper studies online cloud resource provisioning problems under cost budget where no accurate or distributional future information is available. We develop an algorithmic framework and design online algorithms based on the framework. We prove the competitive ratio of the proposed algorithms. We further show the proposed algorithms have better performance than a prominent existing algorithm named CR-Pursuit. While prior works on the problem require the objective functions to be concave, the proposed algorithms work for non-convex and non-concave objective functions. We conduct real-world trace-driven simulations. Results highlight the proposed algorithms outperform baselines significantly over a wide range of settings.**

I. Introduction

The market size of cloud services has been increasing rapidly over the past 10 years, increasing from $68.3 billion in 2010 to $257.5 billion (expected value) in 2020 [1]. Some significant advantages for enterprises to move from local hosts to the cloud are high scalability, low cost, high reliability, and easy maintenance [2]. With the advantages of cloud computing, the only problem left for enterprises is provisioning cloud resources in a quality of service (QoS) optimal manner. Then organizations can focus on their core business rather than on computing infrastructure.

Renewable electricity generation increases continuously, and the share of renewable electricity generation in global electricity generation reaches 28% in 2020 [3]. However, renewable energies, such as solar energy and wind energy, are uncertain, which may cause an imbalance between supply and demand. To maintain a balance between supply and demand, time-varying electricity pricing is adopted. Due to time-varying electric prices and time-varying utilization of cloud resources, the cloud service provider changes the price of cloud resources over time to maximize the revenue [4]. Besides, the QoS that an enterprise experienced is related to the amount of cloud resources available. Therefore, the utility function, which maps an enterprise's costs to the QoS of the enterprise, changes over time.

978-0-7381-3207-5/21/$31.00 ©2021 IEEE

Motivated by the above observations, we focus on a scenario of provisioning cloud resources over a time horizon, where an enterprise decides the expenditure for renting cloud resources in response to real-time utility functions. The goal of the problem is to maximize the QoS of the enterprise over the time horizon. Naturally, there is a constraint on the total cost over the whole time horizon, limiting how much of the money can be used in total during the system's operating time, which is a typical constraint in online resource provisioning applications [5], [6]. We refer to the problem as Online cloud Resources Provisioning problem under cost budget (ORP).

Several different methods have been proposed for online resource provisioning. Some methods that require the system states (utility functions) to have a stationary distribution [7]–[9]. Other methods require a nice structure of the utility functions, i.e., concave or linear [5], [10]. All the methods require precise current utility function at each time slot. For real-world applications, the utility functions are not randomly drawn from a set of utility functions, i.e., system states do not have a stationary distribution. For example, as we can see from Figure 1, cloud virtue machines' prices neither follow a given trajectory nor have a stationary distribution. In addition, the utility functions are not known precisely at the beginning of the corresponding time slots for real-world applications [11]. In this paper, we take the above features of real-world applications into account as follows. First, no accurate or distributional information about future utility functions is available under our setting, while [8], [9] do not work under this setting. Second, different from the literature [5], [10], the utility functions are not necessary to be

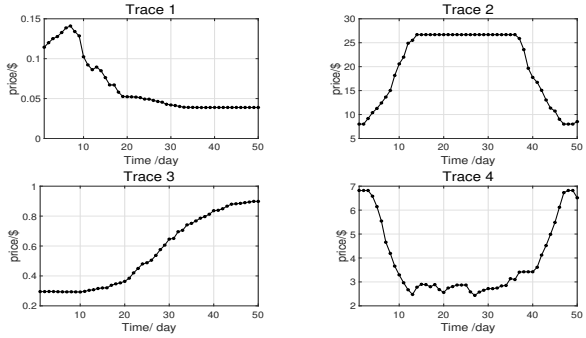


Fig. 1: Four traces of real-world prices per machine hour (in dollars) from Amazon Elastic Compute Cloud (Amazon EC2).

978-1-6654-1494-4/21/$31.00 ©2021 IEEE

concave or linear. Moreover, at the beginning of each time slot, only lower and upper bounds of the current utility function are known to the decision-maker, while the explicit utility function is required in the literature [5], [8], [10].

Our main contributions in this paper are as follows.

- We formulate an online cloud resource provisioning problem under cost budget to maximize the QoS of enterprises in Section II. Our formulation relaxes structural constraints (linear, convex, and concave) of utility functions and can naturally handle the uncertainties in estimating current utility function.
- We propose an algorithmic framework named CRT for ORP*, a special case of ORP, in Section III-A. We provide a scheme named PARL for general ORP based on CRT in Section IV. The CRT-based PARL can achieve a competitive ratio of $c(1+\ln(\theta))$, where c is the parameter that measures the maximum uncertainty of utility functions and θ is the parameter that bounds changes of utility functions over time. The proposed algorithms have better performance than a prominent existing algorithm named CR-Pursuit.
- Our proposed algorithms are evaluated by trace-driven simulations. The results highlight that the proposed algorithm outperforms baseline algorithms used in practice.

The remainder of this paper is organized as follows. Section II presents the problem formulation. Section III proposes the CRT framework for a special case of the problem. Section IV provides a scheme for general ORP based on the algorithms proposed in Section III. Section V shows the performance evaluation of our algorithms. Section VI concludes this paper.

II. System Model and Problem Formulation

In this section, we formulate the online cloud resource provisioning problem with cost budget.

A. System Model

There is an enterprise operating in slotted time, i.e., $t \in [T] \triangleq \{1, 2, \cdots, T\}$. The enterprise serves the requests of its users on rented cloud resources and its permanently owned resources. The cost of permanently owned resources is fixed, and the enterprise has to pay for using cloud resources. At the beginning of each time slot $t \in [T]$, the enterprise spends some money for renting cloud resources, i.e., virtual machines, to serve its users. The quality of service at time slot t is a function of the amount of resources available at time slot t. The goal of the enterprise is to maximize the time cumulative quality of service, subject to the total cost budget for renting cloud resources. Motivated by the fact that the enterprise does not know when the users stop requesting service, we assume that the number of time slots, T, is unknown to the enterprise beforehand.

a) Cost and Cost Constraints: At the beginning of each time slot $t \in [T]$, the enterprise chooses an online decision x_t (in dollars) where x_t represents the amount of money spent on renting cloud resources at time slot t. x_t is chosen from a compact space $[0, \delta]$, i.e., $0 \leq x_t \leq \delta$ for $t \in [T]$. The constraint $x_t \leq \delta$, i.e., the enterprise can only spend a certain amount of money per time slot. In addition, there exists a finite total starting cost budget Δ (in dollars) from which all the costs over $t \in [T]$ are sourced, where $\delta \leq \Delta$. The cost budget constraint can be mathematically stated as $\sum_{t=1}^{T} x_t \leq \Delta$. Δ is the cost budget preemptively specified for renting cloud resources. That is, Δ is known to the enterprise in advance.

b) Quality of Service: At each time slot $t \in [T]$, there is a quality of service corresponding with the cost of renting cloud resources. The quality of service at time slot t is denoted by y_t, which is a function of x_t. In particular, $y_t = f_t(x_t)$ where $f_t(\cdot)$ is the utility function of time slot t that maps costs at time slot t to qualities of service at time slot t, i.e., $f_t : [0, \delta] \rightarrow \mathbb{R}^+$ where $\mathbb{R}^+$ is the set of non-negative real numbers. We assume $f_t(0) = 0$ for $t \in [T]$, i.e., we set the quality of service as 0 if there is no cloud resources rented for serving the users. Note that, $x_t = 0$ does not mean the users' requests are not served at all, because the enterprise has an amount of permanently owned resources which can satisfy the minimum QoS requirement of the users. Utility function $f_t(\cdot)$ changes over time, because the quality of service is a function of the amount of cloud resources available and the unite price of cloud resources changes over time [4]. Let $f_{1:t}$ be the sequence of utility functions from time slot 1 to time slot t. For $t \in [T]$, $f_t(x_t)$ is differentiable over $[0, \delta]$. We have the following assumptions for $f(t), t \in [T]$.

Assumption 1. For $t \in [T]$, $f_t(\cdot)$ satisfies $p(t) \leq f_t'(x_t) \leq cp(t)$ for $x_t \in [0, \delta]$, where $c \geq 1$ is known in advance.

Assumption 2. $p(t) \in [L, U]$ for $t \in [T]$, and the ratio of U to L, denoted by θ, is known in advance.

In the only prior work that considers non stationary system states and non-linear utility functions [5], utility functions are assumed to be differentiable, concave and $p(t) \leq f_t'(x_t) \leq cp(t)$ for $t \in [T]$. However, in real-world systems, the utility functions may not be concave, e.g., utility functions in [12]–[14] are onconvex-nonconcave and hitting and movement costs of online problems with predictions could be non-concave in [15]. Therefore, in Assumption 1, we relax the concavity assumption in the literature, and we only require that $f_t'(x_t)$ is bounded by $p(t)$ and $cp(t)$. Parameter c can not only represents the fluctuation of marginal utility of utility functions as in [5], but also it can reflect the short term prediction error, e.g., at the beginning of time slot t, the decision maker can forecast the marginal utility of time slot t is between $p(t)$ and $cp(t)$ but is not sure what is the explicit value. As in [5], we assume c is known to the decision maker at the very beginning, and c is small for many interesting problems. c can be obtained by historical information about utility functions when it represents the fluctuation of marginal utility of utility functions within time slots as in [5], and c can be obtained by performance parameters of the short term predictor when it measures the short-term prediction error. Assumption 2 requires that $p(t)$, the lower bound of marginal utilities at time slot t, is bounded

by L and U for $t \in [T]$, and the ratio of U to L is known in advance, which is a standard assumption of competitive online optimization problems in the literature [5], [10], [16]–[18]. The duration of each time slot depends on how rapidly does the utility function changes over time. The duration could be several minutes, an hour, a day, and so on depending on the system.

c) Information Revealing Manner: At the beginning of time slot $t \in [T]$, the decision-maker only know lower bound of utility function $f_t(\cdot)$, while previous works require precise $f_t(x_t)$ [5], [10], [18]. At the beginning of each time slot $t \in [T]$, $p(t)$, the lower bound of $f_t'(x_t)$ for $x_t \in [0, \delta]$, is revealed. Upon observing $p(t)$, online algorithms have to choose an online decision x_t.

B. Online Optimization Problem Formulation

The online optimization problem is to maximize the time cumulative quality of service while simultaneously respecting the given cost budget. We refer to the problem as online Cloud Resources Provisioning problem under cost budget (ORP). ORP can be mathematically stated as follows.

$$\begin{aligned} \max_{x_t, t\in[T]} \quad & \sum_{t=1}^{T} f_t(x_t) \\ \text{s.t.} \quad & \sum_{t=1}^{T} x_t \le \Delta, \\ & x_t \in [0, \delta], \forall t \in [T]. \end{aligned} \tag{ORP}$$

At the beginning of each time slot $t \in [T]$, the enterprise observes information about the current utility function $f_t(\cdot)$, i.e., $p(t)$, and decides x_t, the amount of money spent on renting cloud resources. The quality of service at time slot t is $f_t(x_t)$. For an online algorithm alg and input utility function sequence $f_{1:T}$, the time cumulative quality of service, the goal of the system, is denoted by $Q^{alg}(f_{1:T})$. For input utility function $f_{1:T}$, the corresponding optimal offline time cumulative quality of service is denoted by $Q^*(f_{1:T})$.

In this paper, we design online algorithms for ORP, and the performance of the proposed online algorithms is measured in terms of competitive ratio. For online algorithms, the competitive ratio is defined as follows.

Definition 1. A online algorithm *alg* for ORP is called π-competitive if $\frac{Q^*(f_{1:T})}{Q^{alg}(f_{1:T})} \le \pi$ holds for all $f_{1:T} \in \mathcal{F}^T$, where $Q^{alg}(f_{1:T})$ represents the quality of service under *alg* on $f_{1:T}$ and $Q^*(f_{1:T})$ is the optimal offline quality of service on $f_{1:T}$.

For the sake of simplify, we first consider a special case of ORP where $\delta = \Delta$ in Section III. **We use ORP* to denote the ORP problem with $\delta = \Delta$ in the rest of this paper.**

III. Online Algorithm Design and Analysis for ORP*

In this section, we propose an algorithmic framework to design online algorithms for ORP*, i.e., ORP with $\delta = \Delta$. We present the algorithmic framework in Section III-A. A method for general ORP problems is proposed in Section IV based on the algorithms in this section.

Algorithm 1: CRT(ω) for ORP*

```
Set parameter ω;
Set Q(h_{1:0}) = 0;
while t ≤ T do
    Solve (1) to get Q(h_{1:t}) and solve (3) to get x_t^ω;
    Set the decision at time slot t as x_t^ω;
end
```

A. Online Algorithm Design for ORP*

In this subsection, we propose an algorithmic framework called Competitive Ratio Tracker (CRT) for ORP*. CRT has a single parameter ω, which adjusting how aggressively CRT tries to match the optimal offline objective value of the problem, i.e., $Q^*(f_{1:T})$. We use CRT(ω) to denote the CRT-based algorithm with parameter ω. We use x_t^ω to denote the decision at time slot t under CRT(ω).

At the beginning of each time slot, CRT(ω) estimates the optimal offline objective value. To be more specific, CRT(ω) assumes $T = t$, i.e., the system only lasts for t time slots, and $h_t(x_t) \triangleq c \cdot p(\tau) x_\tau, \tau \in [t]$ are the utility functions. We use $Q(h_{1:t})$ to denote the optimal offline objective value estimated at the beginning of time slot t. In other words, $Q(h_{1:t})$ equals the optimal objective value of ORP with input utility functions as $h_{1:t}$, i.e., the optimal objective value of

$$\begin{aligned} \max_{x_\tau, \tau\in[t]} \quad & \sum_{\tau=1}^{t} h_\tau(x_\tau) = \sum_{\tau=1}^{t} c \cdot p(\tau) x_\tau \\ \text{s.t.} \quad & \sum_{\tau=1}^{t} x_\tau \le \delta, \\ & x_\tau \in [0, \delta], \forall \tau \in [t]. \end{aligned}$$

Since the objective functions are linear, $Q(h_{1:t})$ is equivalent to

$$\begin{aligned} Q(h_{1:t}) =& c \cdot \delta \cdot \max\{p(\tau) | \tau \le t\} \\ =& \max\{Q(h_{1:t-1}), c\delta p(t)\}, \end{aligned} \tag{1}$$

where $Q(h_{1:0}) \triangleq 0$. Since $Q(h_{1:t-1})$ is known at the beginning of tome slot t, the time complexity of computing $Q(h_{1:t})$ at each time slot t is $O(1)$. After having $Q(h_{1:t})$, CRT(ω) chooses x_t^ω by solving

$$\sum_{\tau=1}^{t} p(\tau) x_\tau^\omega = \frac{1}{\omega} Q(h_{1:t}). \tag{2}$$

(2) is equivalent to

$$x_t^\omega = \frac{Q(h_{1:t}) - Q(h_{1:t-1})}{\omega \cdot p(t)}. \tag{3}$$

That is, CRT(ω) sets the cost of each time slot $t \in [T]$ as x_t^ω by (3). We formally state CRT(ω) in Algorithm 1. Note that the time complexity for solving an online decision at the beginning of each time slot is $O(1)$, which is extraordinarily time-efficient.

Next, we show a sufficient condition for CRT(ω) to be ω-competitive.

Lemma 1. For any ω, if $\sum_{t=1}^{T} x_t^\omega \le \delta$ under all possible input utility functions, CRT(ω) is feasible and ω-competitive for ORP*.

The proof of Lemma 1 is found in our technical report [19]. For improper parameter ω, the decisions given by CRT(ω) may be infeasible, i.e. $\sum_{t=1}^{T} x_t^{\omega} > \delta$. Since Lemma 1 holds, we then analyze the performance of CRT-based algorithms by finding the minimum ω such that $\sum_{t=1}^{T} x_t^{\omega} \leq \delta$ under all possible input utility functions. The minimum ω such that $\sum_{t=1}^{T} x_t^{\omega} \leq \delta$ under all possible input utility functions is shown in Lemma 2 as follows.

Lemma 2. For any $\omega \geq \omega^* = c \cdot (1 + \log(\theta)$, we have $\sum_{t=1}^{T} x_t^{\omega} \leq \delta$.

The proof of Lemma 2 is found in our technical report [19].Then, we have the **main result** as follow.

Theorem 1. *For any $\omega \geq \omega^* = c \cdot (1 + \ln(\theta))$, CRT($\omega$) is feasible and ω-competitive for ORP*.*

Proof. By combining Lemma 1 and Lemma 2, Theorem 1 is straightforward. Lemma 1 says that CRT(ω) is feasible and ω-competitive for ORP* if $\sum_{t=1}^{T} x_t^{\omega} \leq \delta$ and Lemma 2 shows that $\sum_{t=1}^{T} x_t^{\omega} \leq \delta$ if $\omega \geq \omega^*$. That is, CRT(ω) is feasible and ω-competitive for ORP* if $\omega \geq \omega^*$. □

From [5], c is bounded and small for lots of interesting applications. From Theorem 1, the optimal algorithm among CRT(ω), $\omega \geq \omega^*$ is CRT(ω^*), which is feasible and $c \cdot (1 + \ln(\theta))$-competitive for ORP*. Since the optimal algorithm among CRT(ω), $\omega \geq \omega^*$ is CRT(ω^*), **we automatically set $\omega = \omega^*$ when we apply CRT.** Next, we show a lower bound of the optimal competitive for ORP* and general ORP.

Theorem 2. *The optimal competitive ratio for ORP is lower bounded by $1 + \ln(\theta)$ and upper bounded by $\omega^* = c\,(1 + \ln(\theta))$.*

The proof of Theorem 2 is omitted due to space limitations. The competitive ratio of ORP* with concave objective functions is proved to be lower bounded by $1 + \ln(\theta)$ in [5], so the optimal competitive ratio of ORP is at least $1 + \ln(\theta)$.

CRT(ω^*) has the following **highlights** compared to CR-pursuit in [5]. First, CRT(ω^*) does not require to know $f_t(x_t)$ precisely at the beginning of time slot t, while CR-pursuit requires precise $f_t(x_t)$. Second, although the competitive ratio of CRT(ω^*) is equivalent to that of CR-pursuit, the decision of each time slot made by CRT(ω^*) is no less that of CR-pursuit. That is, the performance of CRT(ω^*) is better than CR-pursuit under all possible input utility functions. In addition, as for time complexity, CRT(ω^*) needs $O(1)$ time complexity to compute an online decision, while CR-pursuit needs to solve a convex optimization for computing an online decision. In conclusion, compared with CR-pursuit, CRT(ω^*) is of *low computing time complexity*, *better performance*, and *has more applications*.

IV. Algorithm Design for general ORP with $\delta < \Delta$

In this section, we provide a scheme to solve general ORP with $\delta < \Delta$ based on CRT. We use PARL to denote the scheme. We use PARL(CRT) to denote the CRT-based PARL scheme for general ORP with $\delta < \Delta$. We consider ORP with $\Delta/\delta \in \mathbb{Z}^+$ and $\Delta/\delta \notin \mathbb{Z}^+$ in Section IV-A and Section IV-B, respectively.

A. PARL for ORP with $\Delta/\delta \in \mathbb{Z}^+$

Since $\Delta/\delta \in \mathbb{Z}^+$, let integer N equals $\frac{\Delta}{\delta}$. PARL considers ORP as N parallel ORP* systems as shown in Figure 2, where each ORP* has a total cost budget of δ. Let x_t^n be the decision of the i^{th} ORP* at time slot t. $p^n(t)$ is the input of the i^{th} ORP* system at time slot t. At the beginning of each time slot t, PARL observes $p(t)$ and updates $p^n(t), n \in [N]$. In particular, the $p^n(t), n \in [N]$ are updated as follows. Set $p^n(0) = 0$ for $n \in [N]$. Let $q^n(t) = \max\{p^n(\tau) | \tau \in [t]\}$. At the beginning of each time slot $t \in [T]$, let $n_t^* = \arg\min\{q^n(t-1) | n \in [N]\}$ where ties are broken randomly. Then, update $p^n(t), n \in [N]$ by letting $p^{n_t^*}(t) = p(t)$ and $p^n(t) = p^n(t-1)$ for $n \neq n_t^*$.

Under PARL(CRT), each of the N ORP* systems chooses its decision at time t by calling CRT. At each time slot, the decision of the ORP system is the summation of decisions of the N parallel ORP* systems. PARL(CRT) for ORP with $\Delta/\delta \in \mathbb{Z}^+$ is formally stated in Algorithm 2.

Algorithm 2: PARL(CRT) for ORP with $\Delta/\delta \in \mathbb{Z}^+$

```
Let N = Δ/δ;
Let δ = Δ/N be the cost budget of the i^th ORP* system;
Set p^n(0) = 0 for n ∈ [N];
while t ≤ T do
    For n ∈ [N], q^n(t−1) = max{p^n(τ) | τ ∈ [t−1]} ;
    Let n_t^* = arg min{q^n(t−1) | n ∈ [N]} (ties are broken randomly);
    Let p^{n_t^*}(t) = p(t) and let p^n(t) = p^n(t−1) for n ∈ [N] \ {n_t^*};
    For n ∈ [N], call CRT to get decision x_t^n;
    Set x_t = Σ_{n=1}^{N} x_t^n;
end
```

In what follows, we analyze the performance of the CRT-based PARL for ORP with $\frac{\Delta}{\delta} \in \mathbb{Z}^+$.

Theorem 3. *PARL(CRT) is feasible and $c \cdot (1 + \ln(\theta))$-competitive for ORP with $\Delta/\delta \in \mathbb{Z}^+$.*

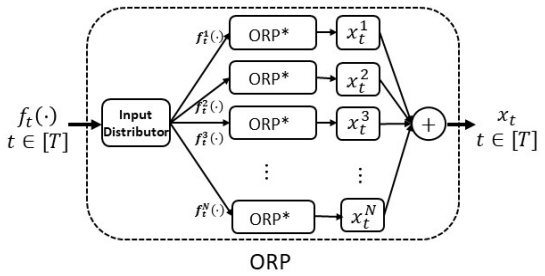


Fig. 2: The idea of PARL for general ORP with cost budget as $\Delta = N \cdot \delta$, is to consider it as N parallel ORP* with cost budget as δ.

Algorithm 3: PARL(CRT) for ORP with $\Delta/\delta \notin \mathbb{Z}^+$

Let N, M be integers such that $\delta = \frac{M}{N}\Delta$;
Let Δ/N be the cost budget of the i^{th} ORP* system;
Set $p^n(0) = 0$ for $n \in [N]$;
while $t \leq T$ **do**
  For $n \in [N]$, $q^n(t-1) = \max\{p^n(\tau)|\tau \in [t-1]\}$;
  Let $\hat{\mathcal{N}}_t$ be the index set of the smallest M values of $q^n(t-1), n \in [N]$ (ties are broken randomly);
  Let $p^n(t) = p(t)$ for $t \in \hat{\mathcal{T}}_t$ and $p^n(t) = p^n(t-1)$ for $n \in [N] \setminus \hat{\mathcal{N}}_t$;
  For $n \in [N]$, call CRT to get decision x_t^n;
  Set $x_t = \sum_{n=1}^{N} x_t^n$;
end

The proof of Theorem 3 is found in our technical report [19]. Since N is a constant, the time complexity of PARL(CRT) is equal to that of CRT.

B. PARL for ORP with $\Delta/\delta \notin \mathbb{Z}^+$

Next, we propose PARL for the case that Δ is not an integer multiple of δ, i.e., $\Delta/\delta \notin \mathbb{Z}^+$. The PARL requires that $f_t(x_t), t \in [T]$ to be concave when $\Delta/\delta \notin \mathbb{Z}^+$. The concavity assumption of utility functions is from the law of diminishing marginal returns [20] and is true in a wide range of applications, e.g. the revenue function is concave on the inventory consumption [5]. From [21], the QoS is a concave and differentiable function of resource allocated to the server.

For any Δ/δ and any $\epsilon > 0$, there exist $M, N \in \mathbb{Z}^+$ such that $|\delta - \frac{M}{N}\Delta| < \epsilon$. Therefore, Let M, N be the integers such that $|\delta - \frac{M}{N}\Delta|$ is sufficiently close to 0. Similar to the idea of Algorithm 2, we consider the system as N parallel ORP* systems where the total cost budget of each subsystem is Δ/N. $p^n(t)$ represents the input of the i^{th} ORP* system at time slot t, and x_t^n is the decision of the n^{th} ORP* system at time slot t. We update $p^n(t), n \in [N]$ as follows. Let $q^n(t) = \max\{p^n(\tau)|\tau \in [t]\}$. At each time slot t, use $\hat{\mathcal{N}}_t$ to represent the index set of the smallest M values of $q^n(t-1), n \in [N]$. At the beginning of each time slot t, let $p^n(t) = p(t)$ for $t \in \hat{\mathcal{N}}_t$ and $p^n(t) = p^n(t-1)$ for $t \notin \hat{\mathcal{N}}_t$. Under PARL(CRT), decisions of the N parallel ORP* systems are given by CRT, and the decision of the ORP system is $x_t = \sum_{n\in[N]} x_t^n$.

PARL(CRT) for ORP with $\frac{\Delta}{\delta} \notin \mathbb{Z}^+$ is formally stated in Algorithm 3.

Theorem 4. *PARL(CRT) is feasible and $c \cdot (1 + \ln(\theta))$-competitive for for ORP with $\Delta/\delta \notin \mathbb{Z}^+$ and concave utility functions.*

The proof of Theorem 4 is similar to that of Theorem 3, so we omit it due to space limitation. In addition, for ORP with $\frac{\Delta}{\delta} \notin \mathbb{Z}^+$, the time complexities of PARL(CRT) is equal to that of CRT.

V. Performance Evaluation

In this section, we conduct simulations to evaluate the performance of the proposed algorithms using real-world data.

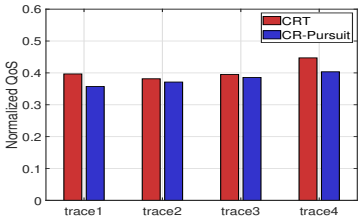


Fig. 3: Comparison of QoS of CRT and CR-Pursuit for ORP* under the four real-world traces.

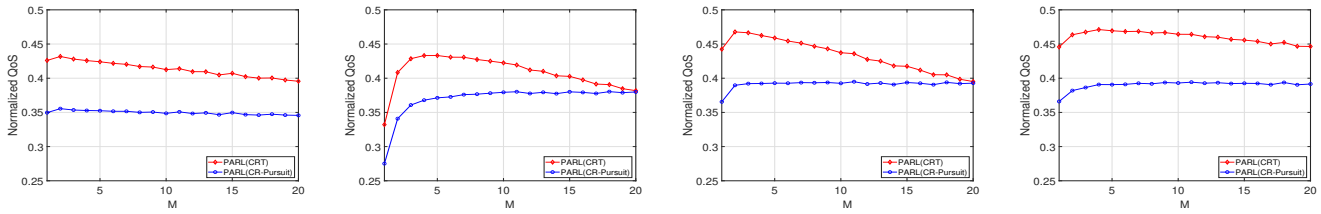
A. Simulation Setup

We consider an enterprise renting a given kind of virtual machine from a cloud computing company, e.g., Amazon Elastic Compute Cloud, to perform some computational jobs. The unit price of the given kind of virtual machine is time-varying [4] as shown in Figure 1. We set the length of each time slot to be a day. Let u_t be the unit price of the given type of virtual machine at time slot t. At the beginning of each day, the enterprise gets access to an imprecisely estimated time average unit price of the current day using a given method, e.g., methods by [22]. Let $\hat{u}_t$ be the imprecisely estimated unit price of time slot t. The imprecisely estimated unit price has a parameter E defined as $E \triangleq \max_{t\in[T]} |(u_t - \hat{u}_t)/\hat{u}_t|$. E measures the maximum error of the imprecisely estimated prices. Parameter E is known in advance, and we set $E = 0.1$ in the following simulations. The online decision of time slot t, x_t, represents the amount of money spent on renting cloud virtual machines at time slot t. The amount of rented virtual machine (in machine hours) at time slot t is $u_t x_t$. From the definition of E, $u_t x_t$ is upper bounded and lower bounded by $\hat{u}_t(1+E)x_t$ and $\hat{u}_t(1-E)x_t$, respectively. The predicted unit price at each time slot t is randomly drawn from $[u_t/(1-E), u_t/(1+E)]$.

The quality of service at time slot t, denoted by y_t, is defined as the amount of machine hours available at time slot t. That is, $y_t = f_t(x_t) = u_t x_t$. At the beginning of each time slot t, the enterprise only knows that $f_t(x_t)$ is upper bounded $c \cdot p(t)$ and lower bounded by $p(t)x_t$, where $c = \frac{1+E}{1-E}$ and $p(t) = \hat{u}_t(1-E)$. We set θ as $\max\{u_t|t \in [T]\}/\min\{u_t|t \in [T]\}$, and θ is known in advance.

In the simulations, we use real-world prices of four types of spot instances from Amazon Elastic Compute Cloud [4] as shown in Figure 1. Trace 1 is the prices of EC2 spot instance with a type of 'm4.xlarge' in location "us-east-2b". Trace 2 is the prices of EC2 spot instance with a type of 'x1e.32xlarge' in location "us-east-2a". Trace 3 is the prices of EC2 spot instance with a type of 'i3en.2xlarge' in location "us-east-2c". Trace 4 is the prices of EC2 spot instance with a type of 'i2.8xlarge' in location "us-east-2a".

The baseline is *CR-Pursuit* [5], which is dedicated for online optimization problems with inventory constraints. Since *CR-Pursuit* requires $f_t(\cdot)$ precisely at the beginning of time slot t for choosing decisions, we assume the accurate price $f_t(\cdot)$ is known to *CR-Pursuit* at the beginning of time slot t, which is a stronger assumption compared to our proposed algorithms.



(a) (b) (c) (d)

Fig. 4: QoS of PARL(CRT) and PARL(CR-Pursuit) using (a) trace 1, (b) trace 2, (c) trace 3, and (d) trace 4.

B. Simulation Results

First, we apply CRT and CR-Pursuit to ORP*. We compare the machine hours got by CRT and CR-Pursuit under the four real-world traces in Figure 1. We normalize all machine hours (QoS) by the corresponding optimal offline values. As shown in Figure 3, although CR-Pursuit knows $f_t(\cdot)$ precisely at the beginning of time slot t, CRT has higher QoS than CR-Pursuit under all the four real-world traces, which verifies the theoretical result that CRT outperforms CR-Pursuit under all possible input utility functions.

Next, we apply CRT-based and CR-Pursuit-based, i.e., PARL(CRT) and PARL(CR-Pursuit), to ORP with $\delta \leq \Delta$. We set $N = 20$ and $\delta = \frac{M}{N}\Delta$. We compare the performance of the algorithms under $M = \{1, 2, \cdots, 20\}$. We normalize all machine hours (QoS) by the corresponding optimal offline values. As shown in Figure 4, the QoS of PARL(CRT) is higher than that of PARL(CR-Pursuit) under all the four real-world traces and $M = \{1, 2, \cdots, 20\}$. EXP has a constant QoS under all the four traces because decisions under EXP is static to M/N, the ratio of δ to Δ.

VI. Conclusion

In this paper, we study the online cloud resources provisioning problem under cost budget named ORP. Under ORP, utility functions can be non-convex and non-concave. In addition, ORP does not require explicit utility functions. We design an algorithmic framework named CRT for ORP*, a spacial case of ORP. In addition, we provide a scheme named PARL to solve general ORP based on CRT. We prove the competitive ratios of the proposed algorithms are $c(1+\ln(\theta))$, where the optimal competitive ratio is lower bounded by $1 + \ln(\theta)$ and c is small in lots of applications. Numerical results using real-world data show that the proposed algorithms outperform existing baselines significantly.

Acknowledgments

This research was partially funded by NSF grants CCF-1526162, CCF-1717731, CNS-1717588, CNS-1730128, CNS-1919752.

References

[1] Statista, "Public cloud services end-user spending worldwide from 2009 to 2022," *Statista Research Department*, 2020, https://www.statista.com/statistics/510350/worldwide-public-cloud-computing/.

[2] S. Mathew and J. Varia, "Overview of amazon web services," *Amazon Whitepapers*, 2014.

[3] IEA (2020), "Global energy review 2020," *IEA, Paris*, 2020, https://www.iea.org/reports/global-energy-review-2020.

[4] Amazon, "Amazon ec2 spot," Website, 2020, https://aws.amazon.com/.

[5] Q. Lin, H. Yi, J. Pang, M. Chen, A. Wierman, M. Honig, and Y. Xiao, "Competitive online optimization under inventory constraints," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 1, p. 10, 2019.

[6] X. Shang, Y. Liu, Y. Mao, Z. Liu, and Y. Yang, "Greening reliability of virtual network functions via online optimization," in *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, 2020, pp. 1–10.

[7] S. Chaisiri, B. Lee, and D. Niyato, "Optimization of resource provisioning cost in cloud computing," *IEEE Transactions on Services Computing*, vol. 5, no. 2, pp. 164–177, 2012.

[8] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[9] Y. Liu, Z. Liu, and Y. Yang, "Non-stationary stochastic network optimization with imperfect estimations," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 431–441.

[10] R. El-Yaniv, A. Fiat, R. M. Karp, and G. Turpin, "Optimal search and one-way trading online algorithms," *Algorithmica*, vol. 30, no. 1, pp. 101–139, 2001.

[11] J. Comden, S. Yao, N. Chen, H. Xing, and Z. Liu, "Online optimization in cloud resource provisioning: Predictions, regrets, and algorithms," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 1, p. 16, 2019.

[12] H. Shajaiah, A. Abdelhadi, and C. Clancy, "Utility functions and resource allocation for spectrum sharing," in *Resource Allocation with Carrier Aggregation in Cellular Networks*. Springer, 2018, pp. 19–24.

[13] P. D. Tao *et al.*, "Network utility maximisation: A dc programming approach for sigmoidal utility function," in *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*. IEEE, 2013, pp. 50–54.

[14] H. Shajaiah, A. Abdel-Hadi, and C. Clancy, "Utility proportional fairness resource allocation with carrier aggregation in 4g-lte," in *MILCOM 2013-2013 IEEE Military Communications Conference*. IEEE, 2013, pp. 412–417.

[15] Y. Lin, G. Goel, and A. Wierman, "Online optimization with predictions and non-convex losses," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, no. 1, pp. 1–32, 2020.

[16] R. El-Yaniv, "Competitive solutions for online financial problems," *ACM Computing Surveys (CSUR)*, vol. 30, no. 1, pp. 28–69, 1998.

[17] A. Borodin and R. El-Yaniv, *Online computation and competitive analysis*. cambridge university press, 2005.

[18] H. Yi, Q. Lin, and M. Chen, "Balancing cost and dissatisfaction in online ev charging under real-time pricing," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1801–1809.

[19] L. Yu, C. Niangjun, L. Zhenhua, and Y. Yuanyuan, "Technical report," 2021, https://drive.google.com/file/d/1gFDx0x0nbD0xaEU885wU8y7_2VlyWeAO/view?usp=sharing.

[20] R. W. Shephard and R. Färe, "The law of diminishing returns," *Zeitschrift für Nationalökonomie*, vol. 34, no. 1-2, pp. 69–90, 1974.

[21] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek, "A resource allocation model for qos management," in *Proceedings Real-Time Systems Symposium*, 1997, pp. 298–307.

[22] S. Agarwal, A. K. Mishra, and D. K. Yadav, "Forecasting price of amazon spot instances using neural networks," *Int. J. Appl. Eng. Res*, vol. 12, no. 20, pp. 10 276–10 283, 2017.