Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 1310237, 10 pages
https://doi.org/10.1155/2020/1310237

Research Article

A Two-Phase Cloud Resource Provisioning Algorithm for Cost Optimization

Junjie Chen 1,2 and Hongjun Li 1

1 *School of Information Science and Technology, Nantong University, Nantong 226019, China*
2 *Nantong Research Institute for Advanced Communication Technologies, Nantong 226019, China*

Correspondence should be addressed to Junjie Chen; cjjcy@ntu.edu.cn

Received 17 June 2020; Revised 27 August 2020; Accepted 23 September 2020; Published 8 October 2020

Academic Editor: S. A. Edalatpanah

Cloud computing is a new computing paradigm to deliver computing resources as services over the Internet. Under such a paradigm, cloud users can rent computing resources from cloud providers to provide their services. The goal of cloud users is to minimize the resource rental cost while meeting the service requirements. In reality, cloud providers often offer multiple pricing models for virtual machine (VM) instances, including on-demand and reserved pricing models. Moreover, the workload of cloud users varies with time and is not known a priori. Therefore, it is challenging for cloud users to determine the optimal cloud resource provisioning. In this paper, we propose a two-phase cloud resource provisioning algorithm. In the first phase, we formulate the resource reservation problem as a two-stage stochastic programming problem, and solve it by the sample average approximation method and the dual decomposition method. In the second phase, we propose a hybrid ARIMA-Kalman model to predict the workload, and determine the number of on-demand instances based on the predicted workload. The effectiveness of the proposed two-phase algorithm is evaluated using a real-world workload trace and Amazon EC2's pricing models. The simulation results show that the proposed algorithm can significantly reduce the operational cost while guaranteeing the service level agreement (SLA).

1. Introduction

Cloud computing [1] is a new computing paradigm to deliver computing resources as services over the Internet. These services are provided at three different levels: Infrastructure as a Service (IaaS) [2], Platform as a Service (PaaS) [3], and Software as a Service (SaaS) [4]. In this paper, we focus on IaaS. IaaS providers such as Amazon EC2 [5] and Microsoft Azure [6] provide their computing resources to cloud users in the form of VMs. Cloud users can rent VMs from cloud providers on a pay-per-use basis.

Cloud providers usually have different billing cycles and offer different pricing models. Take Amazon EC2 as an example. Amazon EC2 has two billing cycles: per hour billing and per second billing. In this paper, we adopt per hour billing. Amazon EC2 offers three pricing models: (1) *On-demand pricing model*. On-demand instances let users pay for compute capacity by the hour with no long-term commitments. (2) *Reserved pricing model*. Users pay an upfront fee (all upfront, partial upfront, and no upfront) to reserve an instance for a 1-year or 3-year term and is then charged a discounted hourly rate for the instance during the reservation period. (3) *Spot pricing model*. Spot instances allow users to bid on unused EC2 instances and run those instances for as long as their bid exceeds the spot price. Spot instances are charged the spot price which is set by Amazon EC2 and adjusted gradually based on the supply and demand for spot instances. Such diverse pricing models make it challenging for cloud users to determine the optimal cloud resource provisioning.

There have been a lot of studies on cloud resource provisioning, which aim to minimize the resource provisioning cost while satisfying the service requirements. However, most existing studies [7–11] do not consider the pricing models or only consider the on-demand pricing model. Some recent studies [12–16] consider both on-demand and reserved pricing models to reduce the resource provisioning cost. These studies typically use reserved instances to meet the minimum service requirements and use on-demand instances to meet the sudden workload demand.

In this paper, we study the cloud resource provisioning problem. To reduce the resource rental cost, we use both on-demand and reserved instances and propose a two-phase cloud resource provisioning algorithm. In the resource reservation phase, we determine the optimal number of reserved instances to minimize the resource rental cost. In the on-demand resource provisioning phase, on-demand instances are purchased based on the predicted workload to guarantee the SLA. The main contributions of this paper are summarized as follows:

(i) We use both on-demand and reserved instances for cloud resource provisioning and propose a two-phase cloud resource provisioning algorithm to reduce the resource rental cost.

(ii) In the first phase, we formulate the resource reservation problem as a two-stage stochastic programming problem, and solve it by the sample average approximation method and the dual decomposition method.

(iii) In the second phase, we propose a hybrid ARIMA-Kalman model for workload prediction and determine the number of on-demand instances based on the predicted workload.

(iv) We conduct extensive experiments to evaluate the effectiveness of the proposed two-phase algorithm using a real-world workload trace and Amazon EC2's pricing models. The experimental results show that the proposed algorithm can significantly reduce the operational cost while guaranteeing the SLA.

The rest of this paper is organized as follows. Related works are reviewed in Section 2. The problem formulation is given in Section 3. The two-phase cloud resource provisioning algorithm is presented in Section 4 and Section 5. Experimental results are presented in Section 6. Finally, we conclude this paper in Section 7.

2. Related Work

In cloud computing, cloud users can reduce the cost and guarantee the QoS requirements through adaptive resource provisioning. Adaptive resource provisioning has been widely studied [7–11]. In [7], the autoscaling techniques were classified into five categories: static threshold-based rules, reinforcement learning, queuing theory, control theory, and time series analysis. Calheiros et al. [8] proposed a workload prediction model using the ARIMA model and evaluated its impact on cloud applications' QoS. Islam et al. [9] developed prediction-based resource measurement and provisioning strategies using neural network and linear regression to satisfy upcoming resource demands. To train the neural network, Shah et al. [17] presented a quick Gbest-guided artificial bee colony learning algorithm. Chen et al. [10] proposed an iterative QoS prediction model and a PSO-based runtime decision algorithm to derive a self-adaptive approach for resource allocation in cloud-based software services. Liu et al. [11] presented SPRNT, a reinforcement learning-based aggressive virtualized resource management system for IaaS clouds.

The above works mainly focus on adaptive resource provisioning. However, cloud providers usually offer multiple pricing models: on-demand, reserved, and spot. Cloud users can significantly reduce the cost based on these pricing models. Chaisiri et al. [12] proposed an optimal cloud resource provisioning algorithm by formulating a stochastic programming model in which the demand and price uncertainty is considered. In [13], a two-phase resource provisioning algorithm was presented. In the first phase, the optimal amount of long-term reserved resources was computed by a mathematical formulae. In the second phase, the authors used the Kalman filter to predict resource demand and adaptively changed the subscribed on-demand resources. Niu et al. [14] proposed a semielastic cluster computing model for organizations to reserve and dynamically resize a virtual cloud-based cluster. In [15], a dynamic instance provisioning strategy based on the large deviation principle was proposed to minimize the number of active instances subject to a QoS requirement in terms of the overload probability. Mireslami et al. [16] proposed a two-phase cloud resource allocation algorithm. In the first phase, reserved resources were allocated to meet the minimum QoS requirements. In the second phase, a stochastic optimization approach was proposed to allocate on-demand resources under demand uncertainty.

In this paper, the cloud resource provisioning problem is formulated as a two-stage stochastic programming problem. It can be transformed into a deterministic integer program and solved by exact methods such as branch and bound and cutting plane methods, or heuristic methods such as genetic algorithm, particle swarm optimization, and hybrid algorithms [18–20]. Grey [18] presented a hybrid PSO-GA algorithm for solving the various constrained optimization problems. In this approach, PSO is used to explore the solution while GA is being used for updating the solution.

3. Problem Formulation

In this section, we present the model assumptions, including the VM configurations and the pricing models. Based on these assumptions, we present the formulation of the cloud resource provisioning problem. The notations used in this paper are listed in Table 1.

3.1. Cloud Computing Environment

3.1. Cloud Computing Environment. Cloud providers offer multiple types of VMs to cloud users. Let $V = \{V_1, V_2, \dots, V_M\}$ denote the set of VM types, where$M$ is the total number of VM types. Each VM type has its own resource configuration and processing capacity. Let C_i denote the processing capacity of a VM instance of type V_i,

Table 1: The key notations.

Notation	Description
V	Set of VM types offered by cloud providers, $V = \{V_1, V_2, \ldots, V_M\}$
C_i	Processing capacity of a VM instance of type V_i
p_i^o	Usage fee of an on-demand instance of type V_i per hour
p_i^R	Upfront payment of a reserved instance of type V_i
p_i^r	Usage fee of a reserved instance of type V_i per hour
T	Number of hours in a reservation period
t	Hour index of the reservation period, $t = 1, 2, \ldots, T$
$\mathbf{R}$	Reservation decision, $\mathbf{R} = (n_1^r, n_2^r, \ldots, n_M^r)$
n_i^r	Number of reserved instances of type V_i
n_{ti}^o	Number of on-demand instances of type V_i at time t
$U(\mathbf{R}, d_t)$	Usage cost of on-demand instances at time t
$p_D(d)$	Probability distribution of the workload

which is the maximum number of concurrent users or the maximum service request rate that can be handled by a VM instance of type V_i without violating the QoS requirements.

We adopt per hour billing and consider two pricing models: on-demand instance and reserved instances (1-year term, partial upfront). Let p_i^o denote the hourly usage fee of an on-demand instance of type V_i. Let p_i^R and p_i^r denote the one-time upfront payment and the hourly usage fee of a reserved instance of type V_i, respectively. Let T be the number of hours in a reservation period. Then, the effective hourly price of a reserved instance of type V_i can be computed as $p_i^R/T + p_i^r$, which is charged for every hour during the reservation period. It is usually assumed that $p_i^R/T + p_i^r < p_i^o$.

3.2. Cloud Resource Provisioning Problem. We consider the cloud resource provisioning problem over a reservation period. Let $t = 1, 2, \ldots, T$ be the hour index of the reservation period. Let d_t be the workload at time t. Let $\mathbf{R} = (n_1^r, n_2^r, \ldots, n_M^r)$ be the reservation decision and n_i^r be the number of reserved instances of type V_i. Then, the reserved processing capacity is $\sum_{i=1}^M n_i^r C_i$, and the total cost of reserved instances for the reservation period is $\sum_{i=1}^M n_i^r (p_i^R + p_i^r T)$. For each time t, if the workload does not exceed the reserved processing capacity, there will be no need to purchase on-demand instances; otherwise, on-demand instances will be purchased, and the usage cost of on-demand instances can be written as

$$U(\mathbf{R}, d_t) = \min \sum_{i=1}^M n_{ti}^o p_i^o, \quad \text{s.t.} \sum_{i=1}^M n_{ti}^o C_i + \sum_{i=1}^M n_i^r C_i \geq d_t,\ n_{ti}^o \in \mathbb{N}_0,\ i \in \{1, 2, \ldots, M\}, \tag{1}$$

where n_{ti}^o is the number of on-demand instances of type V_i at time t.

The resource reservation problem can be formulated as

$$\min \sum_{i=1}^M n_i^r \left(p_i^R + p_i^r T\right) + \sum_{t=1}^T U(\mathbf{R}, d_t), \quad \text{s.t.}\ n_i^r \in \mathbb{N}_0, \quad i \in \{1, 2, \ldots, M\}, \tag{2}$$

where the objective is to minimize the total cost for the reservation period, including the upfront fee and the usage cost of reserved instances, and the usage cost of on-demand instances. This problem depends on the workload over the reservation period, which is not known a priori. We can estimate the probability distribution of the workload $p_D(d)$ based on historical data. Then, the resource reservation problem can be rewritten as

$$\min \sum_{i=1}^M n_i^r \left(\frac{p_i^R}{T} + p_i^r\right) + \sum_d p_D(d) U(\mathbf{R}, d), \quad \text{s.t.}\ n_i^r \in \mathbb{N}_0, \quad i \in \{1, 2, \ldots, M\}. \tag{3}$$

This problem is a two-stage stochastic programming problem, where the objective function is the average cost per hour, and the possible realizations of the workload are called scenarios. The first-stage problem corresponds to the resource reservation problem, where the first-stage decision is the reservation decision. The second-stage problem corresponds to the on-demand resource provisioning problem, where the second-stage decision depends on the realization of the workload.

4. Resource Reservation

In this section, we use the sample average approximation method and the dual decomposition method to solve the resource reservation problem.

4.1. Sample Average Approximation (SAA). If the number of scenarios is very large, it is difficult to solve (3) directly. The sample average approximation method can be used to reduce the number of scenarios [21]. Since the workload is a one-dimensional random variable, a uniform discretization grid is used to generate a set of scenarios $\{\tilde{d}_1, \tilde{d}_2, \ldots, \tilde{d}_N\}$, where N is the sample size. Then, problem (3) can be approximated as

$$\min \sum_{i=1}^M n_i^r \left(\frac{p_i^R}{T} + p_i^r\right) + \frac{1}{N} \sum_{j=1}^N U\left(\mathbf{R}, \tilde{d}_j\right), \quad \text{s.t.}\ n_i^r \in \mathbb{N}_0, \quad i \in \{1, 2, \ldots, M\}. \tag{4}$$

Problem (4) is the SAA of problem (3). Problem (4) is also a two-stage stochastic programming problem, which can be transformed into the following deterministic equivalent formulation:

$$\begin{aligned} \min\ & \sum_{i=1}^{M} n_i^r\left(\frac{p_i^R}{T}+p_i^r\right)+\frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{M} n_{ji}^o p_i^o, \\ \text{s.t. } & \sum_{i=1}^{M} n_i^r C_i+\sum_{i=1}^{M} n_{ji}^o C_i \geq \widetilde{d}_j, \quad j\in\{1,2,\ldots,N\}, \\ & n_i^r \in \mathbb{N}_0,\ i\in\{1,2,\ldots,N\}, \\ & n_{ji}^o \in \mathbb{N}_0,\ j\in\{1,2,\ldots,N\},\ i\in\{1,2,\ldots,M\}. \end{aligned} \tag{5}$$

Problem (5) is an integer linear program, which can be solved using a standard branch and bound algorithm.

4.2. Dual Decomposition-Based Branch and Bound (DDBnB). The standard branch and bound algorithm uses the linear programming relaxation for bounding. In this paper, we use the Lagrangian relaxation obtained by scenario decomposition to improve the bounds [22].

The idea of scenario decomposition is to introduce a copy $\mathbf{R}_j$ of the first-stage decision $\mathbf{R}$ for each scenario. Then, problem (5) can be reformulated as

$$\begin{aligned} \min & \sum_{j=1}^{N}\frac{1}{N}\left(\sum_{i=1}^{M} n_{ji}^r\left(\frac{p_i^R}{T}+p_i^r\right)+\sum_{i=1}^{M} n_{ji}^o p_i^o\right), \\ \text{s.t. } & \sum_{i=1}^{M} n_{ji}^r C_i+\sum_{i=1}^{M} n_{ji}^o C_i \geq \widetilde{d}_j, \quad j\in\{1,2,\ldots,N\}, \\ & n_{ji}^r, n_{ji}^o \in \mathbb{N}_0,\ j\in\{1,2,\ldots,N\},\ i\in\{1,2,\ldots,M\}, \\ & \mathbf{R}_1=\mathbf{R}_2=\cdots=\mathbf{R}_N, \end{aligned} \tag{6}$$

where the constraints $\mathbf{R}_1=\cdots=\mathbf{R}_N$ are called the nonanticipativity constraints. The nonanticipativity constraints have several equivalent expressions. Here, we represent the nonanticipativity constraints by $\sum_{j=1}^{N}\mathbf{H}_j\mathbf{R}_j=\mathbf{0}$, where $\mathbf{H}_j$ is a suitable $M(N-1)\times M$ matrix. By dualizing the nonanticipativity constraints, the Lagrange dual function of problem (6) is defined as

$$\begin{aligned} D(\lambda)=\min & \sum_{j=1}^{N}\frac{1}{N}\left(\sum_{i=1}^{M} n_{ji}^r\left(\frac{p_i^R}{T}+p_i^r\right)+\sum_{i=1}^{M} n_{ji}^o p_i^o\right)+\lambda^T\sum_{j=1}^{N}\mathbf{H}_j\mathbf{R}_j, \\ \text{s.t. } & \sum_{i=1}^{M} n_{ji}^r C_i+\sum_{i=1}^{M} n_{ji}^o C_i \geq \widetilde{d}_j, \quad j\in\{1,2,\ldots,N\}, \\ & n_{ji}^r, n_{ji}^o \in \mathbb{N}_0, \quad j\in\{1,2,\ldots,N\},\ i\in\{1,2,\ldots,M\}, \end{aligned} \tag{7}$$

where $\lambda\in\mathbb{R}^{M(N-1)}$ is the Lagrange multiplier vector associated with the nonanticipativity constraints. Problem (7) can be decomposed into multiple subproblems according to the scenarios:

$$\begin{aligned} D_j(\boldsymbol{\lambda})=\min & \frac{1}{N}\left(\sum_{i=1}^{M} n_{ji}^r\left(\frac{p_i^R}{T}+p_i^r\right)+\sum_{i=1}^{M} n_{ji}^o p_i^o\right)+\boldsymbol{\lambda}^T\mathbf{H}_j\mathbf{R}_j, \\ \text{s.t. } & \sum_{i=1}^{M} n_{ji}^r C_i+\sum_{i=1}^{M} n_{ji}^o C_i \geq \widetilde{d}_j n_{ji}^r, n_{ji}^o \in \mathbb{N}_0, \quad i\in\{1,2,\ldots,M\}. \end{aligned} \tag{8}$$

Problem (8) is called the scenario subproblem, which is a small integer linear program. The dual problem of problem (6) can be formulated as

$$z_{LD}=\max_{\lambda\in\mathbb{R}^{M(N-1)}} D(\lambda)=\sum_{j=1}^{N} D_j(\lambda). \tag{9}$$

Dual problem (9) can be solved by the subgradient method. From the definition of the subgradient, the subgradient of $D(\lambda)$ is $\sum_{j=1}^{N}\mathbf{H}_j\mathbf{R}_j(\lambda)$, where $\mathbf{R}_j(\lambda)$ is the first-stage component of the optimal solution of (8) for a given λ. The iterative formula of the subgradient method is as follows:

$$\lambda^{(k+1)}=\lambda^{(k)}+\gamma^{(k)}\sum_{j=1}^{N}\mathbf{H}_j\mathbf{R}_j\left(\lambda^{(k)}\right), \tag{10}$$

where k is the iteration index and $\gamma^{(k)}$ is a positive step size.

Dual problem (9) provides a lower bound for original problem (6). In general, the scenario solutions $\mathbf{R}_j$, $j=1,2,\ldots,N$ will not satisfy the nonanticipativity constraints unless the duality gap is zero. In this paper, we present a branch and bound algorithm that uses the Lagrangian relaxation of the nonanticipativity constraints for bounding. To obtain a feasible first-stage solution, we compute the average $\overline{\mathbf{R}}=\sum_{i=1}^{N}\mathbf{R}_j/N$ and round it by some heuristic to obtain an integer solution. The feasible first-stage solution provides an upper bound for problem (6). The branch and bound algorithm is described as follows, where $\mathscr{P}$ denotes the set of current problems and $z(P)$ is a lower bound of$P\in\mathscr{P}$:

Step 1. Initialization: set $\overline{z}=+\infty$ and let $\mathscr{P}$ consist of problem (6).

Step 2. Termination: if $\mathscr{P}=\emptyset$, then the solution that yielded $\overline{z}$ is optimal.

Step 3. Node selection: select and delete a problem P from $\mathscr{P}$, and solve its Lagrangian dual.

Step 4. Bounding: if $z_{LD}(P)\geq\overline{z}$, go to Step 2 (this step can be carried out as soon as the value of the Lagrangian dual rises above $\overline{z}$).

(i) The scenario solutions $\mathbf{R}_j$, $j=1,2,\ldots,N$, are identical: let $\overline{z}=z_{LD}(P)$ and delete from $\mathscr{P}$ all problems P' with $z(P')\geq\overline{z}$. Go to Step 2.

(ii) The scenario solutions $\mathbf{R}_j$, $j=1,2,\ldots,N$, differ: compute the average $\overline{\mathbf{R}}=\sum_{i=1}^{N}\mathbf{R}_j/N$ and round it by some heuristic to obtain $\widehat{\mathbf{R}}$. If $\sum_{i=1}^{M}\widehat{n}_i^r(p_i^R/T+p_i^r)+\sum_{j=1}^{N}U(\widehat{\mathbf{R}},t\widetilde{d}_j)/N<\overline{z}$, then let $\overline{z} = \sum_{i=1}^{M} \widehat{n}_i^r (p_i^R/T + p_i^r) + \sum_{j=1}^{N} U(\widehat{\mathbf{R}}, t\widetilde{d}_j)/N$ and delete from $\mathscr{P}$ all problems P' with $z(P') \geq \overline{z}$.

Step 5. Branching: select a component n_i^r of $\mathbf{R}$ and add two new problems to $\mathscr{P}$ obtained from P by adding the constraints $n_i^r \leq \lfloor \overline{n}_i^r \rfloor$ and $n_i^r \geq \lfloor \overline{n}_i^r \rfloor + 1$, respectively. Go to Step 2.

5. On-Demand Resource Provisioning

On-demand resource provisioning problem (1) is an integer linear program, which can be solved using any standard integer linear programming solver. However, the workload is not known a priori. In this paper, we propose a hybrid ARIMA-Kalman model for workload prediction.

It has been shown in the literature that the workload exhibits strong autocorrelation. Then, the workload can be modeled by an ARIMA model [8, 23]:

$$\overline{d}_t = \phi_1 \overline{d}_{t-1} + \phi_2 \overline{d}_{t-2} + \cdots + \phi_p \overline{d}_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}, \quad (11)$$

where $\overline{d}_t = \nabla^d d_t$, $\varepsilon_t \sim \mathrm{WN}(0, \sigma^2)$, $\Phi = (\phi_1, \ldots, \phi_p)$ are the AR coefficients, and $\Theta = (\theta_1, \ldots, \theta_q)$ are the MA coefficients. Let $r = \max(p, q+1)$, and model (11) can be rewritten as

$$\overline{d}_t = \phi_1 \overline{d}_{t-1} + \phi_2 \overline{d}_{t-2} + \cdots + \phi_r \overline{d}_{t-r} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \cdots + \theta_{r-1} \varepsilon_{t-r+1}, \quad (12)$$

where $\phi_i = 0$ for $i > p$ and $\theta_j = 0$ for $j > q$. Then, the state-space representation of model (12) can be obtained as [24]

$$\overline{d}_t = \mathbf{G}\mathbf{x}_t + W_t, \quad (13)$$

$$\mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1} + \mathbf{V}_t, \quad (14)$$

where (13) and (14) are the measurement and state equations, $\overline{d}_t$ is the measurement variable, $\mathbf{x}_t \in \mathbb{R}^r$ is the state vector, $W_t = 0$ is the measurement noise with variance $R = 0$, and $\mathbf{V}_t = (\varepsilon_t, 0, \ldots, 0)^T$ is the state noise with covariance matrix $\mathbf{Q} = \mathrm{diag}(\sigma^2, 0, \ldots, 0)$. The measurement matrix $\mathbf{G}$ and the state transition matrix $\mathbf{F}$ are given as

$$\mathbf{G} = (1, \theta_1, \ldots, \theta_{r-1}),$$
$$\mathbf{F} = \begin{pmatrix} \phi_1 & \phi_2 & \cdots & \phi_{r-1} & \phi_r \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}. \quad (15)$$

From state-space models (13) and (14), the Kalman prediction equations is obtained as follows [25]:

$$\begin{aligned} \mathbf{x}_{t/t-1} &= \mathbf{F}\mathbf{x}_{t-1/t-1}, \\ \mathbf{P}_{t/t-1} &= \mathbf{F}\mathbf{P}_{t-1/t-1}\mathbf{F}^T + \mathbf{Q}, \\ \overline{d}_{t/t-1} &= \mathbf{G}\mathbf{x}_{t/t-1}, \\ Z_{t/t-1} &= \mathbf{G}\mathbf{P}_{t/t-1}\mathbf{G}^T, \end{aligned} \quad (16)$$

$$\begin{aligned} \mathbf{x}_{t/t} &= \mathbf{x}_{t/t-1} + \mathbf{K}_t\left(\overline{d}_t - \mathbf{G}\mathbf{x}_{t/t-1}\right), \\ \mathbf{P}_{t/t} &= (\mathbf{I} - \mathbf{K}_t\mathbf{G})\mathbf{P}_{t/t-1}, \\ \mathbf{K}_t &= \mathbf{P}_{t/t-1}\mathbf{G}^T\left(\mathbf{G}\mathbf{P}_{t/t-1}\mathbf{G}^T\right)^{-1}, \end{aligned} \quad (17)$$

where (16) and (17) are the time and measurement update equations, $\mathbf{x}_{t/s}$ and $\overline{d}_{t/s}$ are the estimates of $\mathbf{x}_t$ and $\overline{d}_t$ given the observations up to time s, $\mathbf{P}_{t/s}$ is the error covariance matrix of $\mathbf{x}_{t/s}$, $Z_{t/s}$ is the error variance of $\overline{d}_{t/s}$, and $\mathbf{K}_t$ is the Kalman gain.

Let $\psi = \{\mathbf{x}_{0/0}, \mathbf{P}_{0/0}, \Phi, \Theta, \sigma^2\}$ denote the set of parameters in the Kalman prediction equations, which can be estimated by the maximum likelihood method. In this paper, we use the EM algorithm [26] to obtain the maximum likelihood estimates of the parameters. If we could observe the states $X_n = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$ in addition to the observations $D_n = \{\overline{d}_1, \overline{d}_2, \ldots, \overline{d}_n\}$, then we would consider $\{X_n, D_n\}$ as the complete data. Under the Gaussian assumption, the log-likelihood of the complete data can be written as

$$\begin{aligned} \ln L(D_n, X_n; \psi) = &-\frac{1}{2}\ln\left|\mathbf{P}_{0/0}\right| - \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}_{0/0})^T \mathbf{P}_{0/0}^{-1}(\mathbf{x}_0 - \mathbf{x}_{0/0}) \\ &\cdot\frac{n}{2}\ln|\mathbf{Q}| - \frac{1}{2}\sum_{t=1}^{n}(\mathbf{x}_t - \mathbf{F}\mathbf{x}_{t-1})^T \mathbf{Q}^{-1}(\mathbf{x}_t - \mathbf{F}\mathbf{x}_{t-1}) \\ &-\frac{n}{2}\ln R - \frac{1}{2}\sum_{t=1}^{n}\left(\overline{d}_t - \mathbf{G}\mathbf{x}_t\right)^2 R^{-1}. \end{aligned} \quad (18)$$

From (18), if we did have the complete data, it will be straight forward to obtain the maximum likelihood estimate of ψ using multivariate normal theory. However, we cannot observe the states. The EM algorithm is an iterative method for finding the maximum likelihood estimate of ψ based on the incomplete data by successively maximizing the conditional expectation of the complete data log-likelihood. Each iteration of the EM algorithm consists of two steps, the expectation step (E-step) and the maximization step (M-step). In the E-step, the conditional expectation of the complete data log-likelihood is computed given the parameter estimates from the previous iteration:

$$Q\left(\psi \mid \psi^{(j-1)}\right) = E\left\{\ln L(D_n, X_n; \psi) \mid D_n, \psi^{(j-1)}\right\}. \quad (19)$$

From (18), we can obtain

$$Q\left(\psi \mid \psi^{(j-1)}\right) = -\frac{1}{2}\ln\left|\mathbf{P}_{0/0}\right| - \frac{1}{2}\operatorname{tr}\left\{\mathbf{P}_{0/0}^{-1}\left[\mathbf{P}_{0/n} + \left(\mathbf{x}_{0/n} - \mathbf{x}_{0/0}\right)\left(\mathbf{x}_{0/n} - \mathbf{x}_{0/0}\right)^T\right]\right\}$$
$$\cdot\frac{n}{2}\ln|\mathbf{Q}| - \frac{1}{2}\operatorname{tr}\left\{\mathbf{Q}^{-1}\left[\mathbf{S}_{11} - \mathbf{S}_{10}\mathbf{F}^T - \mathbf{F}\mathbf{S}_{10}^T + \mathbf{F}\mathbf{S}_{00}\mathbf{F}^T\right]\right\} - \frac{n}{2}\ln R - \frac{1}{2}R^{-1}\sum_{t=1}^{n}\left[\left(\overline{d}_t - \mathbf{G}\mathbf{x}_{t/n}\right)^2 + \mathbf{G}\mathbf{P}_{t/n}\mathbf{G}^T\right], \tag{20}$$

where

$$\mathbf{S}_{11} = \sum_{t=1}^{n}\left(\mathbf{x}_{t/n}\mathbf{x}_{t/n}^T + \mathbf{P}_{t/n}\right),$$
$$\mathbf{S}_{10} = \sum_{t=1}^{n}\left(\mathbf{x}_{t/n}\mathbf{x}_{t-1/n}^T + \mathbf{P}_{t,t-1/n}\right), \tag{21}$$
$$\mathbf{S}_{00} = \sum_{t=1}^{n}\left(\mathbf{x}_{t-1/n}\mathbf{x}_{t-1/n}^T + \mathbf{P}_{t-1/n}\right).$$

$\mathbf{P}_{t,t-1/n}$ is the error covariance of $\mathbf{x}_{t-1/n}$ and $\mathbf{x}_{t/n}$. In the M-step, (20) is maximized with respect to the parameters and then the updated parameter estimates are obtained as

$$\begin{aligned}
\phi_i^{(j)} &= \left(\mathbf{S}_{10}\mathbf{S}_{00}^{-1}\right)_{1i}, \quad i = 1,\ldots,p,\\
\sigma^{2(f)} &= \left(n^{-1}\left(\mathbf{S}_{11} - \mathbf{S}_{10}\mathbf{S}_{00}^{-1}\mathbf{S}_{10}^T\right)\right)_{11},\\
\theta_i^{(j)} &= \left(\left(\sum_{t=1}^{n}\overline{d}_t\mathbf{x}_{t/n}^T\right)\mathbf{S}_{11}^{-1}\right)_{i+1}, \quad i = 1,\ldots,q\\
\mathbf{x}_{0/0}^{(j)} &= \mathbf{x}_{0/n},\\
\mathbf{P}_{0/0}^{(j)} &= \mathbf{P}_{0/n}.
\end{aligned} \tag{22}$$

The flowchart of the EM algorithm is shown in Figure 1.

The one-step-ahead prediction of the workload based on Kalman prediction is given by

$$d_t = -\sum_{j=1}^{d}\binom{d}{j}(-1)^j d_{t-j} + \mathbf{G}\mathbf{x}_{t/t-1}. \tag{23}$$

For each time t, even with a workload prediction method, the underprovisioning problem can occur due to underestimation, which causes the SLA violation. To reduce the SLA violation rate, (23) can be modified as

$$d_t = -\sum_{j=1}^{d}\binom{d}{j}(-1)^j d_{t-j} + \mathbf{G}\mathbf{x}_{t/t-1} + \alpha\cdot\sqrt{\mathbf{G}\mathbf{P}_{t/t-1}\mathbf{G}^T}, \quad \alpha > 0. \tag{24}$$

6. Evaluation

In this section, we conduct extensive experiments to evaluate the effectiveness of the proposed two-phase algorithm based on a real-world workload trace and Amazon EC2's pricing models.

6.1. Experiment Setup. The workload trace used in the experiments is obtained from a 4-week access log file of the NASA web server [27], as shown in Figure 2. The probability distribution of the workload can be estimated based on the workload trace. We consider four types of VM instances offered by Amazon EC2: small (m1.small), medium (m1.medium), large (m1.large), and extralarge (m1.xlarge) [5]. Table 2 shows the configuration and the pricing models of each VM type. The parameters of the algorithms are set as follows. The sample size of the SAA problem is set to 10. In the subgradient method, we use a diminishing step size $\gamma^{(k)} = (1+m)/(k+m)$ where $m = 100$, and repeat the iterations until the stopping criterion $|(D^{(k)} - D^{(k-1)}) \mid D^{(k)}| \leq \varepsilon$ is satisfied where $\varepsilon = 0.00001$. In the EM algorithm, the initial values of the parameters are set according to [25].

6.2. Performance of Resource Reservation Algorithm. We first analyze the impact of resource reservation on the operational cost. Figure 3 shows the operational cost under different resource reservations. We can observe that the operational cost can be significantly reduced by resource reservation, and there is a tradeoff between the on-demand cost and the reservation cost. The optimal resource reservation is $\{n_1^r = 0, n_2^r = 1, n_3^r = 0, n_4^r = 1\}$ with the reserved processing capacity of 275 requests/s, and the optimal operational cost is \$3407.9. By combining on-demand and reserved instances, the operational cost can be reduced by 25.58% compared with the pure on-demand strategy.

We compare the accuracy of the uniform discretization grid with that of the Monte Carlo and quasi-Monte Carlo methods [21]. As can be seen from Figure 4(a), the uniform discretization grid is the best among the three methods under the same sample size. We also study the impact of the sample size on the accuracy of the uniform discretization grid. As can be seen from Figure 4(b), the accuracy of the uniform discretization grid becomes higher as the sample size increases, and reaches 98.01% when $N = 10$.

Figure 5 shows the convergence of the dual decomposition-based branch and bound algorithm. It can be seen that the optimal solution can be obtained by the DDBnB algorithm after 9 iterations. Table 3 compares the performance of our resource reservation algorithm based on stochastic programming (RRSP) with two existing algorithms: the RIPAM algorithm considering only medium instance type [15] and the DCRA algorithm [16]. The RRSP algorithm can reduce the operational cost by 24.92%. Our algorithm can achieve 4.14% more cost saving than RIPAM, and 20.84% more cost saving than DCRA.

6.3. Workload Prediction Based on Hybrid ARIMA-Kalman Model. In this subsection, we evaluate the performance of the hybrid ARIMA-Kalman model. The data of the first three weeks are used as the training data and the data of the last

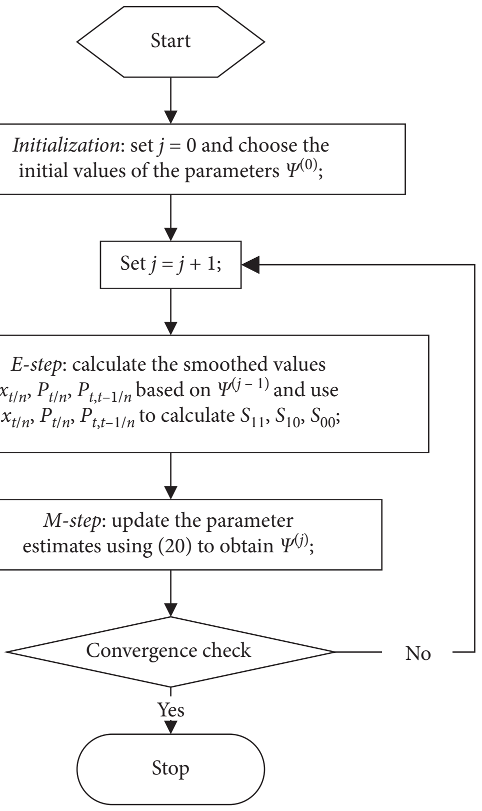
Start
Initialization: set $j = 0$ and choose the initial values of the parameters $\Psi^{(0)}$;
Set $j = j + 1$;
E-step: calculate the smoothed values $x_{t/n}$, $P_{t/n}$, $P_{t,t-1/n}$ based on $\Psi^{(j-1)}$ and use $x_{t/n}$, $P_{t/n}$, $P_{t,t-1/n}$ to calculate S_{11}, S_{10}, S_{00};
M-step: update the parameter estimates using (20) to obtain $\Psi^{(j)}$;
Convergence check
No
Yes
Stop

FIGURE 1: The EM algorithm.

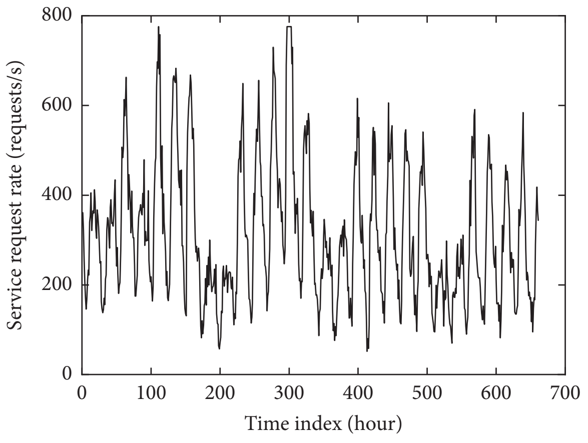

FIGURE 2: The workload trace.

week as the test data. Figure 6 shows the prediction results. We can observe that the predicted workload is very close to the actual workload. The prediction accuracy of the hybrid ARIMA-Kalman model is compared with the ARIMA model [8] and the neural network method [9] based on three metrics, mean absolute percentage error (MAPE), root mean square error (RMSE), and mean absolute error (MAE). The ARIMA model has an autoregressive order of 2 and a moving average order of 1. The neural network method uses the backpropagation neural network, the learning rate is set to 0.7, there is only one hidden layer, and the numbers of neurons in the input, hidden, and output layers are 6, 4, and 1, respectively. As can be seen from Table 4, the hybrid ARIMA-Kalman model is better than the other two methods.

Although the predicted workload is very close to the actual workload, the underprovisioning problem can occur due to underestimation of the workload. To reduce the SLA violation rate, modified workload prediction formula (24) is used. Figure 7 shows the impact of the

Table 2: The configuration and the pricing models of each VM type.

VM type	Resource configuration	Processing capacity (requests/s)	Reserved instance (1 year, partial upfront)		On-demand instance usage fee (per hour)
			Upfront payment	Usage fee (per hour)	
Small	1 vCPU, 1.7GiB RAM, 160 GB disk	20	\$123	\$0.010	\$0.044
Medium	1 vCPU, 3.75GiB RAM, 410 GB disk	50	\$247	\$0.020	\$0.087
Large	2 vCPU, 7.5GiB RAM, 840 GB disk	110	\$493	\$0.042	\$0.175
Extralarge	4 vCPU, 15GiB RAM, 1680 GB disk	225	\$987	\$0.083	\$0.35

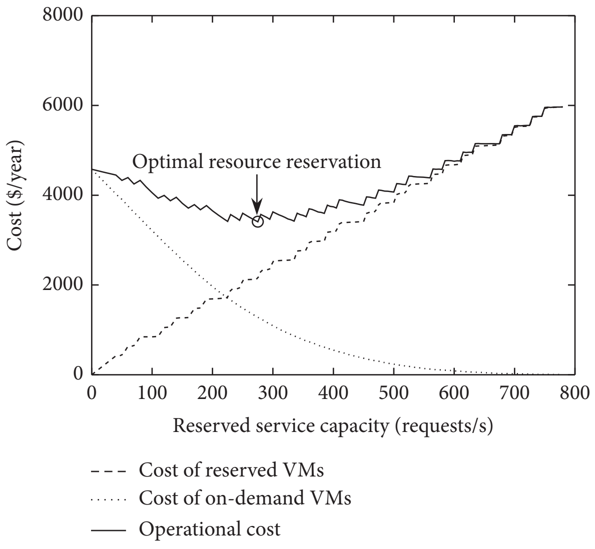

Figure 3: Impact of resource reservation on the operational cost.

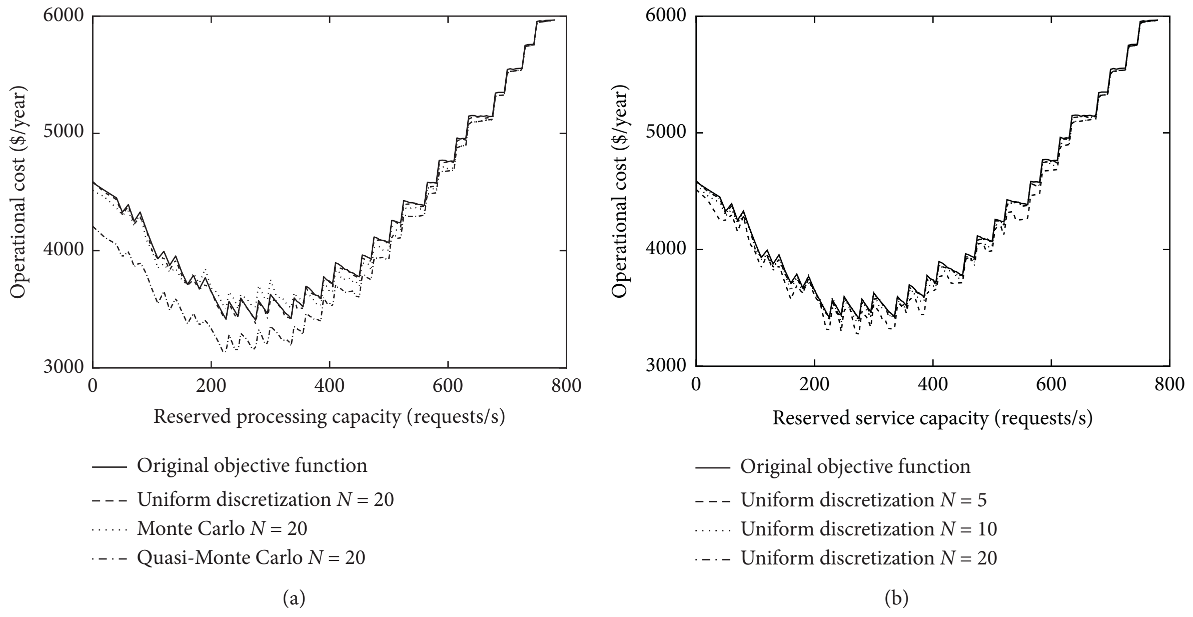

Figure 4: Performance of the SAA method. (a) Comparison of uniform discretization with Monte Carlo and quasi-Monte Carlo. (b) Impact of the sample size on the accuracy of uniform discretization.

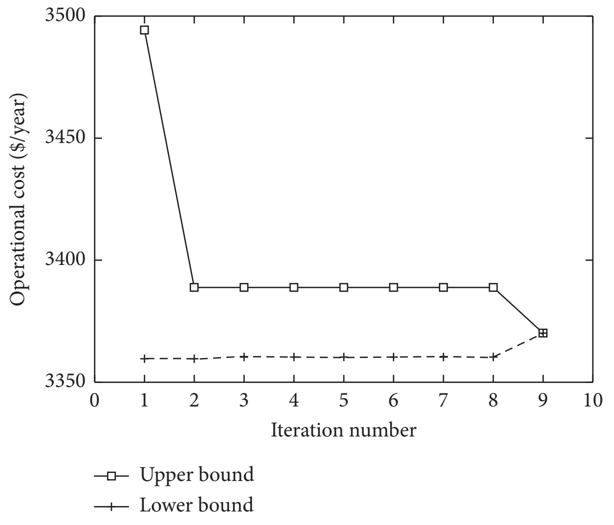

Figure 5: Convergence of the DDBnB algorithm.

Table 3: Comparison of RRSP with two existing algorithms.

	Operational cost ($/year)	Cost saving (%)
RRSP	3438.5	24.92
RIPAM [15]	3628.1	20.78
DCRA [16]	4392.6	4.08

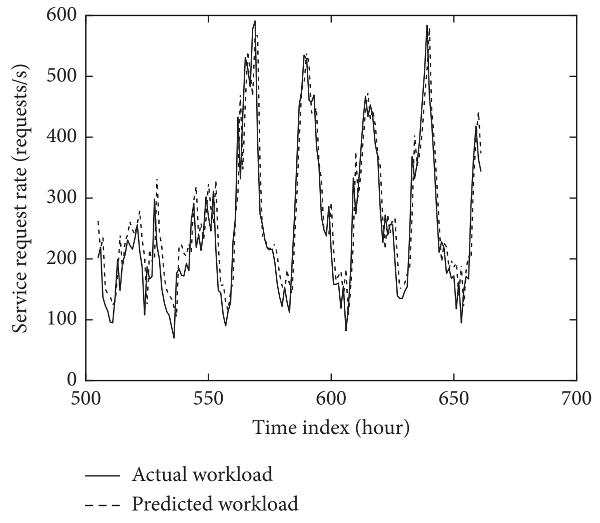

Figure 6: Prediction results.

Table 4: Comparison of prediction accuracy of the three prediction methods.

	MAPE	RMSE	MAE
ARIMA-Kalman	0.1417	52.1098	42.9142
ARIMA	0.1418	52.1102	42.9359
Neural network	0.1426	52.8228	42.6461

parameter α on the SLA violation rate and the on-demand cost. It can be seen that, as the value of α increases, the SLA violation rate decreases while the on-demand cost increases.

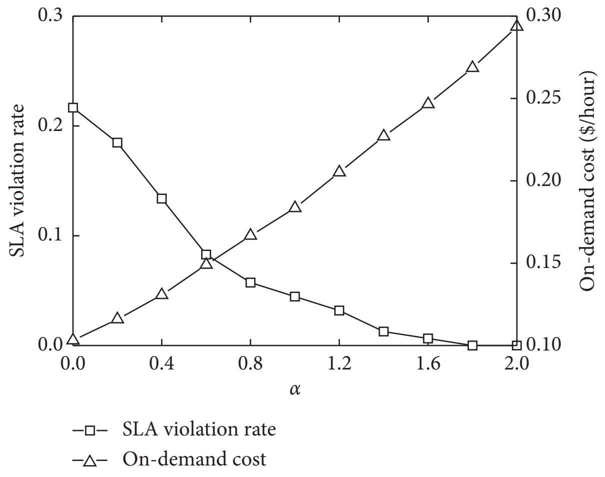

Figure 7: Impact of α on the SLA violation rate and the on-demand cost.

7. Conclusion

In this paper, we propose a two-phase cloud resource provisioning algorithm for cloud users to reduce the resource rental cost using both on-demand and reserved instances. In the first phase, we formulate the resource reservation problem as a two-stage stochastic programming problem. We use the sample average approximation method to reduce the number of scenarios, and solve the SAA problem by a dual decomposition algorithm with branch and bound to obtain the optimal resource reservation. In the second phase, we propose a hybrid ARIMA-Kalman model for workload prediction and determine the number of on-demand instances based on the predicted workload. The effectiveness of the proposed two-phase algorithm is evaluated based on a real-world workload trace and Amazon EC2's pricing models. The simulation results show that the proposed algorithm can achieve about 5%–20% more cost saving than existing algorithms while guaranteeing the SLA. In the future, we plan to investigate more pricing models offered by cloud providers such as spot pricing model, and use these pricing models to further reduce the resource rental cost of cloud users.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Science and Technology Program of Nantong (grant no. JC2018025).

References

[1] M. Armbrust, A. Fox, R. Joseph, R. Katz et al., “A view of cloud computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] S. S. Manvi and G. Krishna Shyam, “Resource management for infrastructure as a service (IaaS) in cloud computing: a survey,” *Journal of Network and Computer Applications*, vol. 41, no. 1, pp. 424–440, 2014.

[3] M. Boniface, B. Nasser, J. Papay et al., “Platform-as-a-service architecture for real-time quality of service management in clouds,” in *Proceedings of the fifth International Conference on Internet and Web Applications and Services*, pp. 155–160, Washington, DC, USA, 2010.

[4] W. Sun, X. Zhang, C. J. Guo et al., “Software as a service: configuration and customization perspectives,” in *Proceedings of the IEEE Congress Services Part II*, pp. 18–25, Washington, DC, USA, 2008.

[5] Amzon EC2, http://aws.amazon.com/ec2.

[6] Microsoft Azure, https://azure.microsoft.com.

[7] T. Lorido-Botran, J. Miguel-Alonso, and J. A. Lozano, “A review of auto-scaling techniques for elastic applications in cloud environments,” *Journal of Grid Computing*, vol. 12, no. 4, pp. 559–592, 2014.

[8] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, “Workload prediction using ARIMA model and its impact on cloud applications’ QoS,” *IEEE Transactions on Cloud Computing*, vol. 3, no. 4, pp. 449–458, 2015.

[9] S. Islam, J. Keung, K. Lee, and A. Liu, “Empirical prediction models for adaptive resource provisioning in the cloud,” *Future Generation Computer Systems*, vol. 28, no. 1, pp. 155–162, 2012.

[10] X. Chen, H. Wang, Y. Ma, X. Zheng, and L. Guo, “Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model,” *Future Generation Computer Systems*, vol. 105, pp. 287–296, 2020.

[11] J. Liu, Y. Zhang, Y. Zhou, D. Zhang, and H. Liu, “Aggressive resource provisioning for ensuring QoS in virtualized environments,” *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 119–131, 2015.

[12] S. Chaisiri, B.-S. 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.

[13] R.-H. Hwang, C.-N. Lee, Y.-R. Chen, and D.-J. Zhang-Jian, “Cost optimization of elasticity cloud resource subscription policy,” *IEEE Transactions on Services Computing*, vol. 7, no. 4, pp. 561–574, 2014.

[14] S. Niu, J. Zhai, X. Ma et al., “Building semi-elastic virtual clusters for cost-effective HPC cloud resource provisioning,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 7, p. 1915, 1928.

[15] Y. Ran, J. Yang, S. Zhang, and H. Xi, “Dynamic IaaS computing resource provisioning strategy with QoS constraint,” *IEEE Transactions on Services Computing*, vol. 10, no. 2, pp. 190–202, 2017.

[16] S. Mireslami, L. Rakai, M. Wang et al., “Dynamic cloud resource allocation considering demand uncertainty,” *IEEE Transactions on Cloud Computing*, 2019.

[17] H. Shah, N. Tairan, H. Garg, and R. Ghazali, “A quick gbest guided artificial bee colony algorithm for stock market prices prediction,” *Symmetry*, vol. 10, no. 7, p. 292, 2018.

[18] H. Garg, “A hybrid GSA-GA algorithm for constrained optimization problems,” *Information Sciences*, vol. 478, pp. 499–523, 2019.

[19] H. Garg, “A hybrid PSO-GA algorithm for constrained optimization problems,” *Applied Mathematics and Computation*, vol. 274, pp. 292–305, 2016.

[20] H. Garg, “An efficient biogeography based optimization algorithm for solving reliability optimization problems,” *Swarm and Evolutionary Computation*, vol. 24, pp. 1–10, 2015.

[21] A. Shapiro, D. Dentcheva, and A. Ruszczyǹski, *Lectures on Stochastic Programming: Modeling and Theory*, SIAM, Philadelphia, USA, 2009.

[22] C. C. Carøe and R. Schultz, “Dual decomposition in stochastic integer programming,” *Operations Research Letters*, vol. 24, no. 1-2, pp. 37–45, 1999.

[23] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control*, John Wiley & Sons, Hoboken, USA, 2008.

[24] J. D. Hamilton, *Time Series Analysis*, Princeton university press, Princeton, USA, 1994.

[25] R. H. Shumway and D. S. Stoffer, *Time Series Analysis and its Applications with R Examples*, Springer, Berlin, Germany, 2011.

[26] R. H. Shumway and D. S. Stoffer, “An approach to time series smoothing and forecasting using the EM algorithm,” *Journal of Time Series Analysis*, vol. 3, no. 4, pp. 253–264, 1982.

[27] NASA-HTTP trace, http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html.