Optimization of Resource Provisioning Cost in Cloud Computing

Sivadon Chaisiri, *Student Member*, *IEEE*,
Bu-Sung Lee, *Member*, *IEEE*, and Dusit Niyato, *Member*, *IEEE*

Abstract—In cloud computing, cloud providers can offer cloud consumers two provisioning plans for computing resources, namely reservation and on-demand plans. In general, cost of utilizing computing resources provisioned by reservation plan is cheaper than that provisioned by on-demand plan, since cloud consumer has to pay to provider in advance. With the reservation plan, the consumer can reduce the total resource provisioning cost. However, the best advance reservation of resources is difficult to be achieved due to uncertainty of consumer's future demand and providers' resource prices. To address this problem, an optimal cloud resource provisioning (OCRP) algorithm is proposed by formulating a stochastic programming model. The OCRP algorithm can provision computing resources for being used in multiple provisioning stages as well as a long-term plan, e.g., four stages in a quarter plan and twelve stages in a yearly plan. The demand and price uncertainty is considered in OCRP. In this paper, different approaches to obtain the solution of the OCRP algorithm are considered including deterministic equivalent formulation, sample-average approximation, and Benders decomposition. Numerical studies are extensively performed in which the results clearly show that with the OCRP algorithm, cloud consumer can successfully minimize total cost of resource provisioning in cloud computing environments.

Index Terms—Cloud computing, resource provisioning, virtualization, virtual machine placement, stochastic programming.

1 Introduction

Cloud computing is a large-scale distributed computing paradigm in which a pool of computing resources is available to users (called *cloud consumers*) via the Internet [1]. Computing resources, e.g., processing power, storage, software, and network bandwidth, are represented to cloud consumers as the accessible public utility services. Infrastructure-as-a-Service (IaaS) is a computational service model widely applied in the cloud computing paradigm. In this model, virtualization technologies can be used to provide resources to cloud consumers. The consumers can specify the required software stack, e.g., operating systems and applications; then package them all together into virtual machines (VMs). The hardware requirement of VMs can also be adjusted by the consumers. Finally, those VMs will be outsourced to host in computing environments operated by third-party sites owned by *cloud providers*. A cloud provider is responsible for guaranteeing the Quality of Services (QoS) for running the VMs. Since the computing resources are maintained by the provider, the total cost of ownership to the consumers can be reduced.

In cloud computing, a resource provisioning mechanism is required to supply cloud consumers a set of computing resources for processing the jobs and storing the data. Cloud providers can offer cloud consumers two resource provisioning plans, namely short-term on-demand and long-term reservation plans. Amazon EC2 [2] and GoGrid [3] are, for instances, cloud providers which offer IaaS services with both plans. In general, pricing in on-demand plan is charged by pay-per-use basis (e.g., 1 day). Therefore, purchasing this on-demand plan, the consumers can dynamically provision resources at the moment when the resources are needed to fit the fluctuated and unpredictable demands. For reservation plan, pricing is charged by a one-time fee (e.g., 1 year) typically before the computing resource will be utilized by cloud consumer. With the reservation plan, the price to utilize resources is cheaper than that of the on-demand plan. In this way, the consumer can reduce the cost of computing resource provisioning by using the reservation plan. For example, the reservation plan offered by Amazon EC2 can reduce the total provisioning cost up to 49 percent when the reserved resource is fully utilized (i.e, steady-state usage) [4].

With the reservation plan, the cloud consumers a priori reserve the resources in advance. As a result, the *underprovisioning problem* can occur when the reserved resources are unable to fully meet the demand due to its uncertainty. Although this problem can be solved by provisioning more resources with on-demand plan to fit the extra demand, the high cost will be incurred due to more expensive price of resource provisioning with on-demand plan. On the other hand, the *overprovisioning problem* can occur if the reserved resources are more than the actual demand in which part of a resource pool will be underutilized. It is important for the cloud consumer to minimize the total cost of resource provisioning by reducing the *on-demand cost* and *oversubscribed cost* of underprovisioning and overprovisioning. To achieve this goal, the optimal computing resource management is the critical issue.

In this paper, minimizing both underprovisioning and overprovisioning problems under the demand and price uncertainty in cloud computing environments is our

• *The authors are with the School of Computer Engineering, Nanyang Technological University (NTU), Nanyang Avenue, Singapore 639798. E-mail: {siva0020, ebslee, dniyato}@ntu.edu.sg.*

Manuscript received 3 Apr. 2010; revised 6 Sept. 2010; accepted 28 Jan. 2011; published online 7 Feb. 2011.
For information on obtaining reprints of this article, please send e-mail to: tsc@computer.org and reference IEEECS Log Number TSCSI-2010-04-0030. Digital Object Identifier no. 10.1109/TSC.2011.7.

motivation to explore a resource provisioning strategy for cloud consumers. In particular, an optimal cloud resource provisioning (OCRP) algorithm is proposed to minimize the total cost for provisioning resources in a certain time period. To make an optimal decision, the demand uncertainty from cloud consumer side and price uncertainty from cloud providers are taken into account to adjust the tradeoff between on-demand and oversubscribed costs. This optimal decision is obtained by formulating and solving a stochastic integer programming problem with multistage recourse [5]. Benders decomposition [6] and sample-average approximation [7] are also discussed as the possible techniques to solve the OCRP algorithm. Extensive numerical studies and simulations are performed, and the results show that OCRP can minimize the total cost under uncertainty.

The major contributions of this paper lie in the mathematical analysis which can be summarized as follows:

- The optimal cloud resource provisioning algorithm is proposed for the virtual machine management. The optimization formulation of stochastic integer programming is proposed to obtain the decision of the OCRP algorithm as such the total cost of resource provisioning in cloud computing environments is minimized. The formulation considers multiple provisioning stages with demand and price uncertainties.
- The solution methods based on Benders decomposition and sample-average approximation algorithms are used to solve the optimization formulation in an efficient way.
- The performance evaluation is performed which can reveal the importance of optimal computing resource provisioning. The performance comparison among the OCRP algorithm and the other approaches is also presented.

The proposed mathematical analysis will be useful to the cloud consumers (e.g., organization and company) for the management of virtual machines in cloud computing environment. The proposed OCRP algorithm will facilitate the adoption of cloud computing of the users as it can reduce the cost of using computing resource significantly.

The rest of this paper is organized as follows: Related works are reviewed in Section 2. The system model and assumption of cloud computing environment are described in Section 3. In Section 4, the stochastic programming formulation of the OCRP algorithm is presented. Section 5 presents the Benders decomposition algorithm. Section 6 presents the sampling-average approximation approach. Experiments and simulations to evaluate the performance of the OCRP algorithm are presented in Section 7. Finally, conclusions are stated in Section 8.

2 Related Work

Available resource provisioning options were discussed in [8]. The resource provisioning strategies in distributed systems were addressed in [9], [10], [11], [12], [13], [14], [15]. In [9], an architectural design of on-demand service for grid computing was proposed. In [10], a profile-based approach to capture expert's knowledge of scaling applications was proposed in which extra demanded resources can be more efficiently provisioned. In [11], the concept of resource slot was proposed. The objective is to address uncertainty of resources availability. In [12], a binary integer program to maximize revenues and utilization of resource providers was formulated. However, [9], [10], [11], [12] did not consider uncertainty of future consumer demands. In [13], an optimization framework for resource provisioning was developed. This framework considered multiple client QoS classes under uncertainty of workloads (e.g., demands of computing resources). The arrival pattern of workloads is estimated by using online forecasting techniques. In [14], heuristic method for service reservation was proposed. Prediction of demand was performed to define reservation prices. In [15], K-nearest-neighbors algorithm was applied to predict the demand of resources. In contrast, our work specifies that demands are given as probability distributions. In addition, the price difference between reservation and on-demand plans was not taken into account in all works in the literature.

As virtualization is a core technology of cloud computing, the problem of virtual machine placement (VM placement) becomes crucial [16], [17], [18], [19], [20]. In [16], the broker-based architecture and algorithm for assigning VMs to physical servers were developed. In [17], a resource management consisting of resource provisioning and VM placement was proposed. In [18], techniques of VM placement and consolidation which leverage *min-max* and *shares* features provided by hypervisors were explored. In [19], a dynamic consolidation mechanism based on constraint programming was developed. This consolidation mechanism was originally designed for homogeneous clusters. However, heterogeneity which is common in a multiple cloud provider environment was ignored. Moreover, [16], [17], [18], [19] did not consider uncertainty of future demands and prices. In [20], a dynamic VM placement was proposed. However, the placement in [20] is heuristic-based which cannot guarantee the optimal solution.

Stochastic programming has been developed to solve resource planning under uncertainty [5] in various fields, e.g., production planning, financial management, and capacity planning. For example, in [21], the authors applied the stochastic programming approach for planning of electrical power generation and transmission line expansion while some uncertainties affecting to the planning are taken into account. It is shown that stochastic programming is the promising mathematical tool which is able to address the optimal decision making in the stochastic environment. However, to the best of our knowledge, the application of stochastic programming to computing resource provisioning has never been exclusively studied.

The *optimal virtual machine placement* (OVMP) algorithm was proposed [22]. This OVMP algorithm can yield the optimal solution for both resource provisioning and VM placement in two provisioning stages. Motivated by this previous work, we introduce the OCRP algorithm in this paper which achieves many improvements. First, the problem is generalized into the multiple stage formulation. Second, the different approaches to obtain the solution of computing resource provisioning are considered. Finally, the performance evaluation is extended to consider various realistic scenarios.

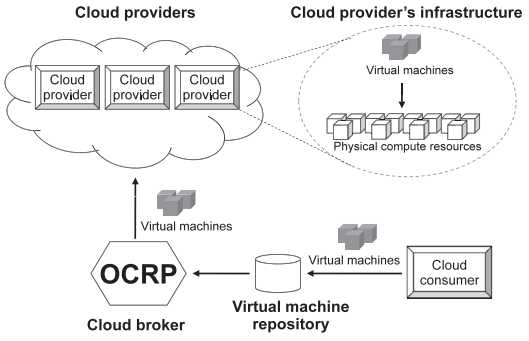


Fig. 1. System model of cloud computing environment.

3 System Model and Assumption

3.1 Cloud Computing Environment

As shown in Fig. 1, the system model of cloud computing environment consists of four main components, namely cloud consumer, virtual machine (VM) repository, cloud providers, and cloud broker. The cloud consumer has demand to execute jobs. Before the jobs are executed, computing resources has to be provisioned from cloud providers. To obtain such resources, the consumer firstly creates VMs integrated with software required by the jobs. The created VMs are stored in the VM repository. Then, the VMs can be hosted on cloud providers' infrastructures whose resources can be utilized by the VMs. In Fig. 1, the cloud broker is located in the cloud consumer's site and is responsible on behalf of the cloud consumer for provision resources for hosting the VMs. In addition, the broker can allocate the VMs originally stored in the VM repository to appropriate cloud providers. The broker implements the OCRP algorithm to make an optimal decision of resource provisioning.

In OCRP, there are multiple VM classes used to classify different types of VM. Let $\mathcal{I} \subset \mathbb{N}_1$[1] denote the set of VM classes. It is assumed that one VM class represents a distinct type of jobs (e.g., one class for web application and the other for database application). A certain amount of resources is required for running the VM, and this required amount of resources can be different for VM in different classes. With this resource requirement, the cloud broker can reserve computing resources from cloud providers to be used in the future according to the actual demand. This demand can be determined as the number of created VMs. In this case, it is possible that additional resources can be provisioned instantly from cloud providers if the reserved resources is not enough to accommodate the actual demand.

Let $\mathcal{J} \subset \mathbb{N}_1$ denote the set of cloud providers. Each cloud provider supplies a pool of resources to the consumer. Let $\mathcal{R}$ denote the set of resource types which can be provided by cloud providers. Resource types can be computing power (in unit of CPU-hours), storage (in unit of GBs/month), and network bandwidth for Internet data transfer (in unit of GBs/month). Each VM class specifies the amount of resources in each resource type. Let b_{ir} be the amount of resource type r required by the VM in class $i \in \mathcal{I}$. It is assumed that every cloud provider prepares facilities, e.g., virtualization management software, network facility, and load balancer, to support the consumer's hosting VM. Note that the key notations used in the paper are listed in Table 1.

1. $\mathbb{N}_1 = \{1, 2, 3, \ldots\}$.

3.2 Provisioning Plans

A cloud provider can offer the consumer two provisioning plans, i.e., reservation and/or on-demand plans. For planning, the cloud broker considers the reservation plan as medium- to long-term planning, since the plan has to be subscribed in advance (e.g., 1 or 3 years [2]) and the plan can significantly reduce the total provisioning cost [4]. In contrast, the broker considers the on-demand plan as short-term planning, since the on-demand plan can be purchased anytime for short period of time (e.g., one week) when the resources reserved by the reservation-plan are insufficient (e.g., during peak load).

3.3 Provisioning Phases

The cloud broker considers both reservation and on-demand plans for provisioning resources. These resources are used in different time intervals, also called *provisioning phases*. There are three provisioning phases: *reservation*, *expending*, and *on-demand* phases. As shown in Fig. 2, these phases with their actions perform in different points of time (or events) as follows. First in the reservation phase, without knowing the consumer's actual demand, the cloud broker provisions resources with reservation plan in advance. In the expending phase, the price and demand are realized, and the reserved resources can be utilized. As a result, the reserved resources could be observed to be either over-provisioned or underprovisioned. If the demand exceeds the amount of reserved resources (i.e., underprovisioned), the broker can pay for additional resources with on-demand plan, and then the on-demand phase starts.

3.4 Provisioning Stages

A provisioning stage is the time epoch when the cloud broker makes a decision to provision resources by purchasing reservation and/or on-demand plans, and also allocates VMs to cloud providers for utilizing the provisioned resources. Therefore, each provisioning stage can consist of one or more provisioning phases. The number of provisioning stages is based on the number of planning epoches considered by the cloud broker, e.g., a yearly plan consists of 12 provisioning stages (i.e., 12 months). Let $\mathcal{T} \subset \mathbb{N}_1$ denote the set of all provisioning stages where $|\mathcal{T}| \geq 2$. For resource provisioning under uncertainty, the broker is assumed to be able to reserve the resources in the first provisioning stage. Also, the broker obtains a solution, called *recourse action* [5], for provisioning resources against uncertainty parameters (i.e., demand and price) in every stage. These uncertainty parameters in each stage will be observed by the broker after the resource reservation has been made. The observed uncertainty parameters are called *realization* (e.g., the actual number of created VMs after the jobs are submitted by the consumers). Then, the broker will take the recourse action according to the realization, e.g., utilizing the reserved resource and/or provisioning more resource with on-demand plan.

Fig. 3 shows the relationship between provisioning phases and provisioning stages, where a yearly plan with 12 provisioning stages, namely $\mathcal{T} = \{T_1, T_2, \ldots, T_{12}\}$, is considered. Fig. 3a shows the example of all three provisioning phases existing in each stage. In Fig. 3b, each

TABLE 1
List of Key Notations

Symbol	Definition
$\mathcal{I}$	Set of virtual machine (VM) classes while $i \in \mathcal{I}$ denotes the VM class index
$\mathcal{J}$	Set of cloud providers while $j \in \mathcal{J}$ denotes the cloud provider index
$\mathcal{K}$	Set of reservation contracts while $k \in \mathcal{K}$ denotes the reservation contract index
$\mathcal{T}$	Set of provisioning stages while $t \in \mathcal{T}$ denotes the provisioning stage index
$\mathcal{R}$	Set of resource types while $r \in \mathcal{R}$ denotes the resource type index
Ω	Set of scenarios while $\omega \in \Omega$ denotes the scenario index
$c_{ijk}^{(\mathrm{R})}$	Reservation cost subscribed to reservation contract k charged by cloud provider j to cloud consumer's VM class i in the first provisioning stage
$c_{ijkt}^{(\mathrm{r})}(\omega)$	Reservation cost subscribed to reservation contract k charged by cloud provider j to cloud consumer's VM class i in provisioning stage t and scenario ω
$c_{ijkt}^{(\mathrm{e})}(\omega)$	Expending cost subscribed to reservation contract k charged by cloud provider j to cloud consumer's VM class i in provisioning stage t and scenario ω
$c_{ijt}^{(\mathrm{o})}(\omega)$	On-demand cost charged by cloud provider j to cloud consumer's VM class i in provisioning stage t and scenario ω
b_{ir}	Amount of resource type r required by VM class i
$d_{it}(\omega)$	Number of VMs (or demand) required to execute class i in provisioning stage t and scenario ω
$a_{jrt}(\omega)$	Maximum capacity of resource type r that cloud provider j can offer to cloud consumer in provisioning stage t and scenario ω
$x_{ijk}^{(\mathrm{R})}$	Decision variable representing the number of VMs in class i provisioned in reservation phase subscribed to reservation contract k offered by cloud provider j in the first provisioning stage
$x_{ijkt}^{(\mathrm{r})}(\omega)$	Decision variable representing the number of VMs in class i provisioned in reservation phase subscribed to reservation contract k offered by cloud provider j in provisioning stage t and scenario ω
$x_{ijkt}^{(\mathrm{e})}(\omega)$	Decision variable representing the number of VMs in class i run in expending phase subscribed to reservation contract k offered by cloud provider j in provisioning stage t and scenario ω
$x_{ijt}^{(\mathrm{o})}(\omega)$	Decision variable representing the number of VMs in class i provisioned in on-demand phase offered by cloud provider j in provisioning stage t and scenario ω

stage may not consist of all provisioning phases. For example, the reservation phase is performed in T_1 and the number of resources are reserved. From T_1-T_3, the expending phase starts in some points of time when some resources reserved in T_1 are utilized. Then, the on-demand phase starts in T_3 due to the insufficient reserved resources, and some resources are provisioned with on-demand plan. In T_4, the reservation phase starts again and so on.

3.5 Reservation Contracts

A cloud provider can offer the consumer multiple reservation plans with different reservation contracts. Each reservation contract refers to the advance reservation of resources with the specific time duration of usage. For example, the reservation plan offered by Amazon EC2 has two reservation contracts [2], namely 1-year contract and 3-year contract. The certain amount of resources are reserved for 1 year in the one-year contract and 3 years in the three-year contract starting from the time when they are provisioned.

Let $\mathcal{K} \subset \mathbb{N}_1$ denote the set of all reservation contracts which are offered by cloud providers. Let L_k denote the time duration (in unit of provisioning stages) specified in reservation contract $k \in \mathcal{K}$. Let $\mathcal{T}_k$ denote the set of stages at which the cloud broker can provision resources by contract k. Let $\mathcal{F}_{kt}$ be the set of stages at which some resources reserved by contract k could be utilized at stage $t \in \mathcal{T}$. Given the total number of stages $|\mathcal{T}|$, both $\mathcal{T}_k$ and $\mathcal{F}_{kt}$ are expressed as follows:

$$\mathcal{T}_k = \{1, \ldots, |\mathcal{T}| - L_k + 1\}, \tag{1}$$

$$\mathcal{F}_{kt} = \{\max(1, t - L_k + 1), \ldots, \min(t, |\mathcal{T}| - L_k + 1)\}. \tag{2}$$

In Fig. 4, the example of advance reservation for the yearly plan with 3-month (K_1) and 6-month (K_2) reservation

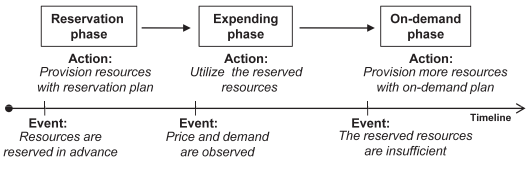


Fig. 2. Transition of provisioning phases.

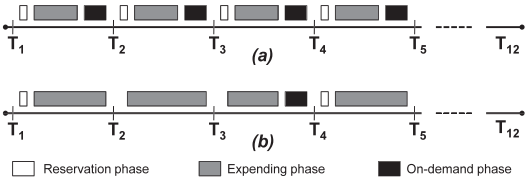


Fig. 3. Relationship between provisioning phases and provisioning stages.

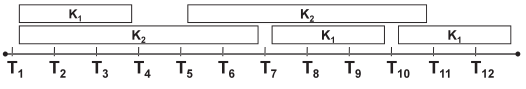


Fig. 4. Example of advance reservations with 3-month (K_1) and 6-month (K_2) contracts.

contracts is shown (i.e., $L_{K_1} = 3$ and $L_{K_2} = 6$). The boxes above the timeline represent the time coverage of some reserved contracts. As shown in (1), $\mathcal{T}_{K_1} = \{T_1, T_2, \ldots, T_{10}\}$ and $\mathcal{T}_{K_2} = \{T_1, T_2, \ldots, T_7\}$ are the sets of stages at which resources can be provisioned by K_1 and K_2, respectively. Contract K_1 is subscribed three times (e.g., T_7-T_9), while contract K_2 is subscribed twice (e.g., T_5-T_{10}). Fig. 4 also shows that some stages can be covered by two (or more) subscribed contracts, e.g., T_1-T_3 are covered by contracts K_1 and K_2. In Fig. 4, any set $\mathcal{F}_{kt}$ in (2) can be obtained, e.g., $\mathcal{F}_{K_1T_4} = \{T_2, T_3, T_4\}$, $\mathcal{F}_{K_1T_{11}} = \{T_9, T_{10}\}$, and $\mathcal{F}_{K_2T_{12}} = \{T_7\}$.

3.6 Uncertainty of Parameters

The optimal solution used by the cloud broker is obtained from the OCRP algorithm based on stochastic integer programming [5]. Stochastic programming takes a set of uncertainty parameters (called *scenarios*), described by a probability distribution into account. Let Ω denote the set of all scenarios in every provisioning stage and Ω_t denote the set of all scenarios in provisioning stage t. Set Ω is defined as the Cartesian product of all Ω_t, namely

$$\Omega = \prod_{t \in \mathcal{T}} \Omega_t = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_{|\mathcal{T}|}. \tag{3}$$

It is assumed that the probability distribution of Ω has *finite support*, i.e., set Ω has a finite number of scenarios with respective probabilities $p(\omega) \in [0, 1]$ where ω is a composite variable defined as $\omega = (\omega_1, \ldots, \omega_{|\mathcal{T}|}) \in \Omega$. In this paper, demand and price are considered as scenarios in Ω whose probability distribution is assumed to be available. The actual scenario of uncertainty parameter after it is observed by the broker is called realization.

3.7 Provisioning Costs

With three aforementioned provisioning phases, there are three corresponding provisioning costs incurred in these phases, namely reservation, expending, and on-demand costs. The main objective of the OCRP algorithm is to minimize all of these costs while the consumer's demand is met, given the uncertainty of demand and price.

For cloud provider, the price is defined in dollars ($) per resource unit. Let $c^{(\mathrm{R})}_{jkr}$ denote the unit price (i.e., costs to the consumer) of resource type r subscribed to reservation contract k provided by cloud provider j in reservation phase of the first provisioning stage. It is assumed that the price of reservation plan in the first stage is charged by a fixed one-time fee. The reservation cost $c^{(\mathrm{R})}_{ijk}$ is the cost for provisioning every resource type defined as follows:

$$c^{(\mathrm{R})}_{ijk} = \sum_{r \in \mathcal{R}} b_{ir} c^{(\mathrm{R})}_{jkr}. \tag{4}$$

The prices in reservation and expending phases could be adjusted by cloud providers without informing the consumer in advance, except the price of the reservation plan in the first provisioning stage. For instance, the cost of electric power to supply a cloud provider's data center could be increased by power plants in the next few months, and the cloud provider will be able to increase the costs of computing resources in the future as well. For the prices in provisioning stage t given scenario ω in both reservation and expending phases, $c^{(\mathrm{r})}_{jkrt}(\omega)$ and $c^{(\mathrm{e})}_{jkrt}(\omega)$ denote the unit prices of resource type r with reservation contract k provided by cloud provider j, respectively. Let $c^{(\mathrm{r})}_{ijkt}(\omega)$ and $c^{(\mathrm{e})}_{ijkt}(\omega)$, defined with the similar way as (4), denote the reservation and expending costs for provisioning every resource type, respectively.

In on-demand phase, let $c^{(\mathrm{o})}_{jrt}(\omega)$ denote the unit price of resource type r provided by cloud provider j in provisioning stage t given scenario ω. Also, the price can be changed by cloud providers (i.e., uncertain to consumer when the resource is reserved). Let $c^{(\mathrm{o})}_{ijt}(\omega)$, defined with the similar way as (4), denote the on-demand cost for provisioning every resource type. Given VM class i, cloud provider j, provisioning stage t, and scenario ω, the expending cost of any reservation contract k is assumed to be cheaper than the on-demand cost (i.e., $c^{(\mathrm{e})}_{ijkt}(\omega) < c^{(\mathrm{o})}_{ijt}(\omega)$).

From the system model and assumption, the optimal cloud resource provisioning algorithm is developed to minimize the total provisioning costs under the price and demand uncertainty in multiple provisioning stages. In the next section, the stochastic programming formulation of the OCRP algorithm is presented.

4 Stochastic Programming Model

In this section, the stochastic programming with multistage recourse [5] is presented as the core formulation of the OCRP algorithm. First, the original form of stochastic integer programming formulation is derived. Then, the formulation is transformed into the deterministic equivalent formulation (DEF) which can be solved by traditional optimization solver software.

4.1 Stochastic Integer Programming for OCRP

Minimize:

$$z = \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} \sum_{k \in \mathcal{K}} c^{(\mathrm{R})}_{ijk} x^{(\mathrm{R})}_{ijk} + \mathbb{E}_{\Omega}\big[\mathcal{Q}\big(x^{(\mathrm{R})}_{ijk}, \omega\big)\big], \tag{5}$$

subject to:

$$x^{(\mathrm{R})}_{ijk} \in \mathbb{N}_0, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall k \in \mathcal{K}. \tag{6}$$

The general form of stochastic integer program of the OCRP algorithm is formulated in (5) and (6). The objective function (5) is to minimize the cloud consumer's total provisioning cost. Decision variable $x^{(\mathrm{R})}_{ijk}$ denotes the number of VMs provisioned in the first provisioning stage. In other words, this number refers to as the total amount of reserved resources. The expected cost under the uncertainty Ω is defined as $\mathbb{E}_{\Omega}[\mathcal{Q}(x^{(\mathrm{R})}_{ijk}, \omega)]$ where $\mathcal{Q}(x^{(\mathrm{R})}_{ijk}, \omega)$ is expressed as follows:

$$\mathcal{Q}\big(x^{(\mathrm{R})}_{ijk}, \omega\big) = \min_{Y = (x^{(\mathrm{r})}_{ijkt}(\omega), x^{(\mathrm{e})}_{ijkt}(\omega), x^{(\mathrm{o})}_{ijt}(\omega))} \mathcal{C}(Y), \quad Y \in \Upsilon\big(x^{(\mathrm{R})}_{ijk}, \omega\big). \tag{7}$$

In (7), the objective of $\mathcal{Q}(x^{(\mathrm{R})}_{ijk}, \omega)$ is to minimize the cost under uncertainty given scenario ω and $x^{(\mathrm{R})}_{ijk}$. Such cost is represented by $\mathcal{C}(\cdot)$ and defined in (9). Composite variable Y representing the solution of $\mathcal{Q}(x^{(\mathrm{R})}_{ijk}, \omega)$ consists of variables, namely $x^{(\mathrm{r})}_{ijkt}(\omega)$, $x^{(\mathrm{e})}_{ijkt}(\omega)$, and $x^{(\mathrm{o})}_{ijt}(\omega)$, which denote the numbers of VMs provisioned in reservation, expending, and on-demand phases, respectively. Set $\Upsilon(x^{(\mathrm{R})}_{ijk}, \omega)$ controls the relationship among the variables by constraints as expressed in (8)-(15). The constraint in (10)

maintains the amount of resources utilized in expending phase to be less than or equal to the number of reserved resources as stated in (2). In (11), the constraint implies that the reservation in the first stage can be performed without any uncertainty. The constraint in (12) ensures that the consumer's demand for VM class $i \in \mathcal{I}$ in stage $t \in \mathcal{T}$ is met. The constraint in (13) states that the allocation of resources for VMs must not exceed the maximum resource capacity offered by a cloud provider. Constraints (14) and (15) indicate that variables take the values from a set of nonnegative integer numbers (i.e., $\mathbb{N}_0$).

$$\mathcal{Q}\big(x_{ijk}^{(\mathrm{R})}, \omega\big) = \min \mathcal{C}(Y), \tag{8}$$

where

$$\begin{aligned}\mathcal{C}(Y) = &\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}_k} c_{ijkt}^{(\mathrm{r})}(\omega)x_{ijkt}^{(\mathrm{r})}(\omega) \\ &+\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}}\left(\sum_{k\in\mathcal{K}} c_{ijkt}^{(\mathrm{e})}(\omega)x_{ijkt}^{(\mathrm{e})}(\omega) + c_{ijt}^{(\mathrm{o})}(\omega)x_{ijt}^{(\mathrm{o})}(\omega)\right),\end{aligned} \tag{9}$$

subject to:

$$x_{ijkt}^{(\mathrm{e})}(\omega) \le \sum_{\hat{t}\in\mathcal{F}_{kt}} x_{ijk\hat{t}}^{(\mathrm{r})}(\omega), \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}, \tag{10}$$

$$x_{ijk}^{(\mathrm{R})} = x_{ijk\bar{t}}^{(\mathrm{r})}(\omega), \quad \bar{t}=1, \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \tag{11}$$

$$\sum_{j\in\mathcal{J}}\left(\sum_{k\in\mathcal{K}} x_{ijkt}^{(\mathrm{e})}(\omega) + x_{ijt}^{(\mathrm{o})}(\omega)\right) \ge d_{it}(\omega), \quad \forall i\in\mathcal{I}, \forall t\in\mathcal{T}, \tag{12}$$

$$\sum_{i\in\mathcal{I}} b_{ir}\left(\sum_{k\in\mathcal{K}} x_{ijkt}^{(\mathrm{e})}(\omega) + x_{ijt}^{(\mathrm{o})}(\omega)\right) \le a_{jrt}(\omega), \quad \forall j\in\mathcal{J}, \forall r\in\mathcal{R}, \forall t\in\mathcal{T}, \tag{13}$$

$$x_{ijkt}^{(\mathrm{r})}(\omega) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}_k, \tag{14}$$

$$x_{ijkt}^{(\mathrm{e})}(\omega) \in \mathbb{N}_0, x_{ijt}^{(\mathrm{o})}(\omega) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}. \tag{15}$$

4.2 Deterministic Equivalent Formulation

Given a probability distribution of all scenarios in set Ω, the formulation in (5)-(15) can be transformed into the deterministic integer programming called *deterministic equivalent formulation* as expressed in (16)-(23). To solve this DEF, probability distributions of both price and demand must be available, i.e., $p(\omega)$ in (16). Then, the DEF can be solved by using traditional optimization solver software. For example, the formulation is implemented using MathProg script, and then the script is solved by GNU Linear Programming Kit (GLPK) [23].

Minimize:

$$\begin{aligned}\hat{z}_\Omega = &\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} c_{ijk}^{(\mathrm{R})}x_{ijk}^{(\mathrm{R})} + \sum_{\omega\in\Omega}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}_k} p(\omega)c_{ijkt}^{(\mathrm{r})}(\omega)x_{ijkt}^{(\mathrm{r})}(\omega) \\ &+\sum_{\omega\in\Omega}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} p(\omega)\left(\sum_{k\in\mathcal{K}} c_{ijkt}^{(\mathrm{e})}(\omega)x_{ijkt}^{(\mathrm{e})}(\omega) + c_{ijt}^{(\mathrm{o})}(\omega)x_{ijt}^{(\mathrm{o})}(\omega)\right),\end{aligned} \tag{16}$$

subject to: (6)

$$x_{ijkt}^{(\mathrm{e})}(\omega) \le \sum_{\hat{t}\in\mathcal{F}_{kt}} x_{ijk\hat{t}}^{(\mathrm{r})}(\omega), \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}, \forall\omega\in\Omega, \tag{17}$$

$$x_{ijk}^{(\mathrm{R})} = x_{ijk\bar{t}}^{(\mathrm{r})}(\omega), \quad \bar{t}=1, \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall\omega\in\Omega, \tag{18}$$

$$\sum_{j\in\mathcal{J}}\left(\sum_{k\in\mathcal{K}} x_{ijkt}^{(\mathrm{e})}(\omega) + x_{ijt}^{(\mathrm{o})}(\omega)\right) \ge d_{it}(\omega), \quad \forall i\in\mathcal{I}, \forall t\in\mathcal{T}, \forall\omega\in\Omega, \tag{19}$$

$$\sum_{i\in\mathcal{I}} b_{ir}\left(\sum_{k\in\mathcal{K}} x_{ijkt}^{(\mathrm{e})}(\omega) + x_{ijt}^{(\mathrm{o})}(\omega)\right) \le a_{jrt}(\omega), \quad \forall j\in\mathcal{J}, \forall r\in\mathcal{R}, \forall t\in\mathcal{T}, \forall\omega\in\Omega, \tag{20}$$

$$x_{ijkt}^{(\mathrm{r})}(\omega) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}_k, \forall\omega\in\Omega, \tag{21}$$

$$x_{ijkt}^{(\mathrm{e})}(\omega) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}, \forall\omega\in\Omega, \tag{22}$$

$$x_{ijt}^{(\mathrm{o})}(\omega) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall t\in\mathcal{T}, \forall\omega\in\Omega. \tag{23}$$

5 Benders Decomposition

In this section, the Benders decomposition algorithm [6] is applied to solve the stochastic programming problem formulated in Section 4. The goal of this algorithm is to break down the optimization problem into multiple smaller problems which can be solved independently and parallelly. As a result, the time to obtain the solution of the OCRP algorithm can be reduced. The Benders decomposition algorithm can decompose integer programming problems with complicating variables into two major problems: *master problem* and *subproblem*.

Property 1. *The DEF derived in (16)-(23) is the problem whose structure has multiple complicating variables.*

Proof. Variables $x_{ijkt}^{(\mathrm{e})}(\omega)$ from the DEF defined in (16)-(23) are considered as complicating variables [6]. Since variables $x_{ijkt}^{(\mathrm{e})}(\omega)$ exist in constraints (17), (19), and (20), the variables prevent the decomposability of the DEF. If variables $x_{ijkt}^{(\mathrm{e})}(\omega)$ are given the fixed values are denoted by $x_{ijkt}^{(\mathrm{fix})}(\omega)$, the DEF can be decomposed into two types of independent optimization subproblems, namely S_1 and $S_2(\omega)$ presented as follows:

$[S_1]$ Minimize:

$$\begin{aligned}z_\nu^{(\mathrm{r})} = &\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} c_{ijk}^{(\mathrm{R})}x_{ijk}^{(\mathrm{R})} \\ &+\sum_{\omega\in\Omega}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}_k} p(\omega)c_{ijkt}^{(\mathrm{r})}(\omega)x_{ijkt}^{(\mathrm{r})}(\omega),\end{aligned} \tag{24}$$

subject to: (6), (17), (18), (21)

$$x_{ijkt}^{(\mathrm{e})}(\omega) = x_{ijkt}^{(\mathrm{fix})}(\omega), \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}, \forall\omega\in\Omega, \tag{25}$$

$[S_2(\omega)]$ Minimize:

$$z_\nu^{(\mathrm{o})}(\omega) = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} p(\omega)c_{ijt}^{(\mathrm{o})}(\omega)x_{ijt}^{(\mathrm{o})}(\omega), \tag{26}$$

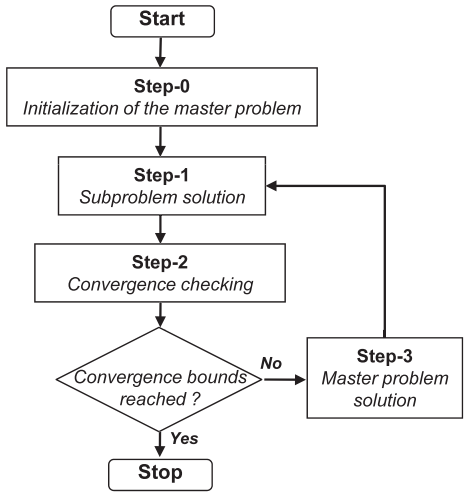


Fig. 5. Flowchart of Benders decomposition algorithm.

subject to: (19), (20), (23)

$$x^{(\mathrm{e})}_{ijkt}(\omega) = x^{(\mathrm{fix})}_{ijkt}(\omega), \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \tag{27}$$

From this decomposition, we conclude that the DEF has the structure with multiple complicating variables. □

From Property 1, the problem can be solved by Benders decomposition algorithm. The algorithm consists of steps which are performed iteratively. At each iteration, the master problem constituted by the complicating variables and subproblems constituted by the other decision variables are solved, then lower and upper bounds are calculated. The algorithm stops when optimal solution converges, i.e., the lower and upper bounds are satisfactorily close to each other.

In Fig. 5, the flowchart of Benders decompostion algorithm is shown. The algorithm for solving OCRP is presented in four steps (i.e., Step-0 to Step-3) as follows.

Step-0: Initialization of the master problem. In Step-0, the step is the initialization of the master problem. This Step-0 is performed only once, while Step-1 to Step-3 are repeatable in the algorithm. Let ν denote the iteration counter and initially set $\nu = 1$. The master problem as expressed in (28)-(32) is an alternative form of the formulation DEF shown in (16)-(23).

Minimize:

$$z^{(\mathrm{e})}_{\nu} = \sum_{\omega\in\Omega}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}} p(\omega) c^{(\mathrm{e})}_{ijkt}(\omega) x^{(\mathrm{e})}_{ijkt\nu}(\omega) + \alpha_{\nu}, \tag{28}$$

subject to:

$$\alpha_{\nu} \geq \alpha^{(\mathrm{lb})}, \tag{29}$$

$$\sum_{j\in\mathcal{J}} x^{(\mathrm{e})}_{ijkt\nu}(\omega) \leq d_{it}(\omega), \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \tag{30}$$

$$\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}} b_{ir} x^{(\mathrm{e})}_{ijkt}(\omega) \leq a_{jrt}(\omega), \quad \forall j \in \mathcal{J}, \forall r \in \mathcal{R}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \tag{31}$$

$$x^{(\mathrm{e})}_{ijkt\nu}(\omega) \in \mathbb{N}_0, \quad \forall i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \forall \omega \in \Omega, \tag{32}$$

The objective function (28) is directly derived from that in (16). $x^{(\mathrm{e})}_{ijkt\nu}(\omega)$ represents variable $x^{(\mathrm{e})}_{ijkt}(\omega)$ in iteration ν of master problem, while variable α_{ν} provides the minimum cost given reservation and on-demand costs. This α_{ν} will be improved in consequent iterations. Initially, α_{ν} can be fixed by constant $\alpha^{(\mathrm{lb})}$ as shown in constraint (29). This $\alpha^{(\mathrm{lb})}$ can be estimated from an economical analysis or historical data of prior solutions [6]. Constraints (30)-(32) define the boundary of $x^{(\mathrm{e})}_{ijkt\nu}(\omega)$. After solving the master problem, the algorithm proceeds to the Step-1.

Step-1: Subproblem solution. In Step-1, multiple subproblems are formulated and solved. Lets assign solution $x^{(\mathrm{e})}_{ijkt\nu}(\omega)$ obtained from the master problem to variables

$$x^{(\mathrm{fix})}_{ijkt}(\omega).$$

Given the fixed solution $x^{(\mathrm{fix})}_{ijkt}(\omega)$, two aforementioned subproblems, namely, S_1 and $S_2(\omega)$ can be solved concurrently.

The subproblem S_1 is presented in (24)-(25) in which the objective function is to minimize the reservation cost. Let variable $\lambda^{(\mathrm{r})}_{ijkt\nu}(\omega)$ denote the optimal solution of the dual problem of S_1 in iteration ν associated with constraint (25). The solution of $\lambda^{(\mathrm{r})}_{ijkt\nu}(\omega)$ will be used in Step-3.

The subproblem $S_2(\omega)$ is presented in (26)-(27) in which the objective function is to minimize the on-demand cost when the realization is set to ω. $S_2(\omega)$ associates with the number of scenarios $|\Omega|$, and hence $|\Omega|$ subproblems are generated. Note that $|\Omega|$ is the Cardinality of set Ω. Let variable $\lambda^{(\mathrm{o})}_{ijkt\nu}(\omega)$ denote the optimal value of the dual problem of $S_2(\omega)$ in iteration ν associated with constraint (27). The solution of $\lambda^{(\mathrm{o})}_{ijkt\nu}(\omega)$ will be used in Step-3.

Step-2: Convergence checking. In Step-2, the convergence of lower and upper bounds of the solutions obtained from master problem and subproblems is checked. Both bounds are adjusted in each iteration. The lower bound in iteration ν denoted as $z^{(\mathrm{lb})}_{\nu}$ can be obtained from the objective function of the master problem, i.e., $z^{(\mathrm{lb})}_{\nu} = z^{*(\mathrm{e})}_{\nu}$. The upper bound in iteration ν denoted by $z^{(\mathrm{ub})}_{\nu}$ can be obtained from

$$z^{(\mathrm{ub})}_{\nu} = z^{*(\mathrm{e})}_{\nu} - \alpha_{\nu} + z^{*(\mathrm{r})}_{\nu} + \sum_{\omega\in\Omega} z^{*(\mathrm{o})}_{\nu}(\omega). \tag{33}$$

Let ϵ denote a small tolerance value to verify the convergence of both lower and upper bounds. The Benders decomposition algorithm stops when $z^{(\mathrm{ub})}_{\nu} - z^{(\mathrm{lb})}_{\nu} < \epsilon$, which means both bounds are acceptably close to each other and the optimal solution can be found in iteration ν. Otherwise, the algorithm proceeds to the next iteration in which Step-3 will perform.

Step-3: Master problem solution.

$$\begin{aligned}\alpha_{\nu} \geq \sum_{\omega\in\Omega}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}} \big(\big(\lambda^{(\mathrm{r})}_{ijkt\overline{\nu}}(\omega) + \lambda^{(\mathrm{o})}_{ijkt\overline{\nu}}(\omega)\big) \\ \big(x^{(\mathrm{e})}_{ijkt\nu}(\omega) - x^{(\mathrm{e})}_{ijkt\overline{\nu}}(\omega)\big)\big) \\ + z^{*(\mathrm{r})}_{\overline{\nu}} + \sum_{\omega\in\Omega} z^{*(\mathrm{o})}_{\overline{\nu}}(\omega), \quad \overline{\nu} \in \{1, \ldots, \nu-1\}.\end{aligned} \tag{34}$$

Let the iteration counter be increased by $\nu \leftarrow \nu + 1$. Then, the master problem in (28)-(32) can be further relaxed

by additional constraints called *Benders cuts* [6]. In addition, the solution of the master problem will adjust the cost α_ν and also the expending cost according to solution of $x^{(e)}_{ijkt\nu}(\omega)$. As shown in (34), the Benders cuts are constructed from the optimal costs obtained from master problem and subproblems in the prior iterations. After solving this master problem, Step-1 is repeated and the same iterative process continues.

6 Sample-Average Approximation

In the case that the number of scenarios is numerous, it may not be efficient to obtain the solution of the OCRP algorithm by solving the stochastic programming formulation defined in (16)-(23) directly if all scenarios in the problem are considered. To address this complexity issue, the sample-average approximation (SAA) approach is applied [7]. This approach selects a set of scenarios, e.g., N scenarios, where N is smaller than the total number of scenarios $|\Omega|$. Then, these N scenarios can be solved in a deterministic equivalent formulation. The optimal solution can be obtained if N is large enough which can be verified numerically.

In this section, the SAA approach is applied to approximate the expected cost in every considered provisioning stage, i.e., $\mathcal{Q}(x^{(R)}_{ijk}, \omega)$ in (7). A sampling method (e.g., Monte Carlo [24] and Latin hypercube [25]), is used to generate scenarios $\Omega_N = \{\omega_1, \ldots, \omega_N\}$, where N denotes the sample size. Let $\mathcal{N} = \{1, \ldots, N\}$ be the set of indices of samples. Then, the expected cost can be redefined as shown in (42).

The function $\hat{\mathcal{Q}}(x^{(R)}_{ijk}, N)$ is the SAA to the objective function in (5). Then, the problem can be transformed into a deterministic equivalent formulation, as called *approximation problem* (AP) formulation, as expressed in (35)-(41).
Minimize:

$$\hat{z}_N = \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} c^{(R)}_{ijk}x^{(R)}_{ijk} + \frac{1}{N}\sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}\sum_{t\in\mathcal{T}_k} c^{(r)}_{ijkt}(\omega_n)x^{(r)}_{ijkt}(\omega_n) + \frac{1}{N}\sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}}\left(\sum_{k\in\mathcal{K}} c^{(e)}_{ijk}(\omega_n)x^{(e)}_{ijkt}(\omega_n) + c^{(o)}_{ijt}(\omega_n)x^{(o)}_{ijt}(\omega_n)\right) \quad (35)$$

subject to: (6)

$$x^{(e)}_{ijkt}(\omega_n) \le \sum_{\hat{t}\in\mathcal{F}_{kt}} x^{(r)}_{ijk\hat{t}}, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}, \forall n\in\mathcal{N}, \quad (36)$$

$$x^{(R)}_{ijk} = x^{(r)}_{ijk\bar{t}}(\omega_n), \quad \bar{t}=1, \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall n\in\mathcal{N}, \quad (37)$$

$$\sum_{j\in\mathcal{J}}\left(\sum_{k\in\mathcal{K}} x^{(e)}_{ijkt}(\omega_n) + x^{(o)}_{ijt}(\omega_n)\right) \ge d_{it}(\omega_n), \quad \forall i\in\mathcal{I}, \forall t\in\mathcal{T}, \forall n\in\mathcal{N}, \quad (38)$$

$$\sum_{i\in\mathcal{I}} b_{ir}\left(\sum_{k\in\mathcal{K}} x^{(e)}_{ijkt}(\omega_n) + x^{(o)}_{ijt}(\omega_n)\right) \le a_{jrt}(\omega_n), \quad \forall j\in\mathcal{J}, \forall r\in\mathcal{R}, \forall t\in\mathcal{T}, \forall n\in\mathcal{N}, \quad (39)$$

$$x^{(r)}_{ijkt}(\omega_n) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}_k, \forall n\in\mathcal{N}, \quad (40)$$

$$x^{(e)}_{ijkt}(\omega_n), x^{(o)}_{ijt}(\omega_n) \in \mathbb{N}_0, \quad \forall i\in\mathcal{I}, \forall j\in\mathcal{J}, \forall k\in\mathcal{K}, \forall t\in\mathcal{T}, \forall n\in\mathcal{N}. \quad (41)$$

$$\hat{\mathcal{Q}}(x^{(R)}_{ijk}, N) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{Q}(x^{(R)}_{ijk}, \omega_n). \quad (42)$$

Let z^* and x^* denote the optimal objective function value and optimal solution of the original formulation (defined in (5) and (6)), respectively. Let $\hat{z}^*_N$ and $\hat{x}^*_N$ denote the optimal objective function value and optimal solution of the AP formulation, respectively. Note that although $\hat{z}^*_N$ becomes closer to z^* when N is large, value $\hat{z}^*_N$ naturally varies according to set of samples Ω_N. Therefore, an estimation method is required to achieve the SAA lower and upper bounds of the optimal solution. Obviously, $\hat{z}^*_N$ forms the SAA upper bound of z^* as follows:

$$z^* \le \hat{z}^*_N. \quad (43)$$

In addition, the SAA lower bound of z^* is formed by the following unbiased property

$$\mathbb{E}[\hat{z}^*_N] \le z^*. \quad (44)$$

For the properties in (43) and (44), bounding method is required to obtain the estimates of both SAA upper and lower bounds on z^* with a certain confidence interval. The next two following sections present the estimation of the SAA bounds by applying the similar method to that in [7], [21], [24].

6.1 SAA Lower Bound Estimates

The expected value $\mathbb{E}[\hat{z}^*_N]$ can be estimated by generating M independent batches,[2] each of size N, denoted as $\omega_{1,m}, \ldots, \omega_{N,m}$ where $m \in \{1, \ldots, M\}$, then solving the AP formulation. Let $\hat{z}^*_{N,m}$ denote the solution given batch m. Next, the SAA lower bound can be obtained from

$$L_{N,M} = \frac{1}{M}\sum_{m=1}^{M}\hat{z}^*_{N,m}. \quad (45)$$

Due to (44), this $L_{N,M}$ is an unbiased estimator of the mean $\mathbb{E}[\hat{z}^*_N]$ which forms a statistical lower bound for z^*. When the generated M batches are independent and identically distributed (i.i.d.) by the Central Limit Theorem, the distribution of SAA lower bound estimate converges to a normal distribution $\mathcal{N}(0, \sigma^2_L)$,[3] namely

$$\sqrt{M}(L_{N,M} - \mathbb{E}[\hat{z}^*_N]) \xrightarrow{D} \mathcal{N}(0, \sigma^2_L), \text{ as } M \to \infty, \quad (46)$$

where $\sigma^2_L = \mathrm{Var}[\hat{z}^*_N]$[4] which can be approximated by the sample variance estimator $s^2_L(M)$ as follows:

$$s^2_L(M) = \frac{1}{M-1}\sum_{m=1}^{M}(\hat{z}^*_{N,m} - L_{N,M})^2. \quad (47)$$

Finally, the $(1-\alpha)$-confidence interval of the SAA lower bound can be defined as

2. A batch is a set of samples generated by a sampling technique.
3. $\mathcal{N}(0, \sigma^2_L)$ denotes the normal distribution with mean 0 and variance σ^2_L.
4. $\mathrm{Var}[\hat{z}^*_N]$ denotes the variance of samples.

$$\left[L_{N,M} - \frac{z_{\alpha/2} s_L(M)}{\sqrt{M}}, L_{N,M} + \frac{z_{\alpha/2} s_L(M)}{\sqrt{M}}\right], \quad (48)$$

where z_α satisfies $\text{Prob}\{N(0,1) \leq z_\alpha\} = 1 - \alpha$. Note that the value $z_{\alpha/2}$ from the Z-distribution can be replaced by critical value $t_{\alpha/2,M-1}$ from the Student's t-distribution if M is small.

6.2 SAA Upper Bound Estimates

As shown in (43), $\hat{z}_N^*$ forms the SAA upper bound of z^*. By selecting a solution $\hat{x}_N^*$ obtained by solving the AP formulation, the SAA upper bound can be estimated by using the unbiased estimator of $\hat{z}_N^*$. To achieve this estimator, we can generate $\tilde{M}$ independent batches, each of size $\tilde{N}$, denoted as $\omega_{1,m}, \ldots, \omega_{\tilde{N},m}$ where $m \in \{1, \ldots, \tilde{M}\}$. Since solution $\hat{x}_N^*$ is already fixed to the SAA problem, i.e., $\hat{z}_N(\hat{x}_N^*)$, given $\tilde{N}$ samples of each generated batch, the objective function (35) can be decomposed into $\tilde{N}$ independent subproblems in which each of them can be solved independently and in parallel. Thus, these subproblems can be solved more efficiently than the original AP formulation. After all subproblems are solved, each batch yields the solution denoted as $\hat{z}_{\tilde{N},m}(\hat{x}_N^*)$. Then, the SAA upper bound can be estimated from

$$U_{\tilde{N},\tilde{M}}(\hat{x}_N^*) = \frac{1}{\tilde{M}} \sum_{m=1}^{\tilde{M}} \hat{z}_{\tilde{N},m}(\hat{x}_N^*). \quad (49)$$

Again, the distribution of SAA upper bound estimate converges to a normal distribution $\mathcal{N}(0, \sigma_U^2(\hat{x}_N^*))$ given solution $\hat{x}_N^*$, namely

$$\sqrt{\tilde{M}}\big(U_{\tilde{N},\tilde{M}}(\hat{x}_N^*) - \hat{z}_N(\hat{x}_N^*)\big) \xrightarrow{D} \mathcal{N}\big(0, \sigma_U^2(\hat{x}_N^*)\big), \text{ as } \tilde{M} \to \infty, \quad (50)$$

where $\sigma_U^2(\hat{x}_N^*) = \text{Var}[\hat{z}_N(\hat{x}_N^*)]$ which can be approximated by the sample variance estimator $s_U^2(\tilde{M}, \hat{x}_N^*)$ as follows:

$$s_U^2(\tilde{M}, \hat{x}_N^*) = \frac{1}{\tilde{M} - 1} \sum_{m=1}^{\tilde{M}} (\hat{z}_{\tilde{N},m}(\hat{x}_N^*) - U_{\tilde{N},\tilde{M}}(\hat{x}_N^*))^2. \quad (51)$$

Finally, the $(1 - \alpha)$-confidence interval of the SAA upper bound can be obtained from

$$\left[U_{\tilde{N},\tilde{M}}(\hat{x}_N^*) - \frac{z_{\alpha/2} s_U(\tilde{M}, \hat{x}_N^*)}{\sqrt{\tilde{M}}}, U_{\tilde{N},\tilde{M}}(\hat{x}_N^*) + \frac{z_{\alpha/2} s_U(\tilde{M}, \hat{x}_N^*)}{\sqrt{\tilde{M}}}\right]. \quad (52)$$

6.3 Monte Carlo Sampling

The Monte Carlo sampling technique can be applied to generate scenarios Ω_N. Each scenario is from Ω as follows. First, the random number is uniformly selected from $[0, 1]$. Then, the scenario is derived by the inverse transformation method given the random number and cumulative probability distribution of Ω. The same process is iteratively performed until the N scenarios are completely chosen.

6.4 Obtaining the Optimal Solution

The optimal solution based on the SAA approach can be obtained when N is sufficiently large. However, we can select an optimal solution by solving different SAA problems with different size of N. Until the same solution is found in these problems, we can choose the solution as the desired solution. The detail and result of this solution method is presented in the next section.

7 Performance Evaluation

In this section, the performance evaluation of the proposed OCRP algorithm is presented. Two case studies are considered in this evaluation, namely *two provisioning stage problem* (2-PSP) and *12 provisioning stage problem* (12-PSP). The former, 2-PSP, has only two provisioning stages. We assume that the cloud broker is making a decision for provisioning resources at the end of year. Under price and demand uncertainty, the cloud broker performs the advance reservation of resources in the first stage for being used in the next whole year which is the second stage. Therefore, the 1-year reservation contract is sufficiently required by the broker since the contract can cover the time duration. At the second stage, the price and demand are observed. Then, the number of reserved resources are utilized and some additional amount of resources can be provisioned in an on-demand fashion.

For 12-PSP, we consider 12 months in a year and hence the stage is defined as $\mathcal{T} = \{T_1, T_2, \ldots, T_{12}\}$. In each stage, the cloud broker can perform an advance reservation of resources for being used in the next incoming months within these 12 months. Moreover, additional resources can be provisioned by purchasing on-demand plans if the reserved resources cannot meet the actual demand. For both case studies, the optimal solution obtained from the OCRP algorithm is the amount of reserved resources in different provisioning stages (or the first stage for 2-PSP). Since the amount of resources is reserved for the number of VMs, this optimal solution can be considered to *the number of reserved VMs* in other words.

7.1 Experiment Setup

7.1.1 Setting of Cloud Computing Environment

We first present the parameter setting of a cloud computing environment used in this performance evaluation. The environment consists of only one cloud consumer (i.e., organization) who is renting computing resources offered by cloud providers. The consumer has two different types of applications representing two distinct VM classes, namely $\mathcal{I} = \{I_1, I_2\}$. For instance, I_1 is a database server and I_2 is a web server. Each VM class requires different amount of resources. Processing time units required by a VM in classes I_1 and I_2 are 8,748 and 6,570 CPU-hours per year, respectively. Permanent storage capacities required by a VM in classes I_1 and I_2 are 1,920 and 1,200 GBs per year, respectively. Network bandwidth regarding outbound data-transfer required by a VM in classes I_1 and I_2 are 24,000 and 30,000 GBs per year, respectively. The cost of inbound data transfer is assumed to be free of charge. Furthermore, a software package is needed to be installed in a VM of each VM class. A software package consists of operating system, database software (for class I_1 only), web application software (for class I_2 only), and other utility software. The software cost is additionally charged to the consumer as the license cost per a running VM. The license software costs corresponding to a VM in classes I_1 and I_2 are \$500 and \$1,200, respectively. The consumer is assumed to purchase these software licenses from software vendors when VMs are running in expending and on-demand phases.

The environment consists of four cloud providers, namely $\mathcal{J} = \{J_1, J_2, J_3, J_4\}$. J_1 represents the private cloud

TABLE 2
Pricing Defined by Each Cloud Provider

Provider	Price per VM in reservation phase			Unit price in expending phase			Unit price in on-demand phase		
	3-M	6-M	1-Y	Processing time	Storage	Network	Processing time	Storage	Network
J_1	N/A	N/A	\$357	\$0	\$0	\$0.10	N/A	N/A	N/A
J_2	\$56.90	\$87.63	\$227.50	\$0.03	\$0.10	\$0.15	\$0.085	\$0.10	\$0.15
J_3	\$69.38	\$106.85	\$277.50	\$0.02	\$0.10	\$0.15	\$0.10	\$0.14	\$0.19
J_4	N/A	N/A	N/A	N/A	N/A	N/A	\$0.09	\$0.075	\$0.15

and the other represent the three public clouds. The private cloud J_1 is the data center belonging to the cloud consumer. This data center has housed only 10 physical servers. Each server is assumed to offer 100 percent uptime system availability, i.e., $24 \times 365 = 8{,}760$ CPU-hours per year. Therefore, for the whole data center, J_1 can offer 87,600 CPU-hours as the maximum processing capacity. The other resource types in J_1 is assumed to be abundant to serve all VMs run by the consumer. The total cost to utilize the servers in J_1 is only considered from the annual energy cost. This annual cost is assumed to be the average energy cost taken by Dell PowerEdge M600 blade server as presented in [26]. The cost is calculated as $454.39/1{,}000$ kilowatts (average power consumption per server[5]) $\times$ \$0.0897 (electric charge per watt-hour)$\times 24$ (hours per day) $\times$ 365 (days per year) $\approx$ \$357 per server per year. For only J_1, this cost (\$357) is considered as the reservation cost to reserve resources for a VM, while the expending cost of processing time and storage capacity are omitted. Only the cost of network bandwidth is charged to \$0.10 per GB per month of outbound data transfer. Furthermore, the on-demand plan for provisioning resources is unavailable in J_1.

The public cloud providers (J_2, J_3, and J_4) are assumed to offer unlimited capacity of all resource types, so the constraint in (13) is omitted. Pricing of resources in provider J_2 is based on the prices defined by Amazon EC2 (in February 2010), and the price of processing time is based on that of the *Small Instance* type [2]. However, pricing in providers J_3 and J_4 is artificially and reasonably defined. Providers J_2 and J_3 offer customers both reservation and on-demand plans, while J_4 has only an on-demand plan. Providers J_2 and J_3 offer customers three different reservation contracts, namely 3-month (3M), 6-month (6M), and 1-year (1Y) contracts. Pricing of resource in expending and on-demand phases is charged as the pay-per-use basis based on the actual usage per resource unit. The resource unit of processing time is CPU-hour, while one of storage capacity and network bandwidth is GBs per month. For the network bandwidth, only the outbound data transfer is charged, while the inbound one is free of charge in every cloud provider.[6] Pricing defined by every cloud provider is listed in Table 2.

7.1.2 Uncertainty Parameters

In this part, two main uncertainty parameters including the price of resource and the demand as the number of required VMs per VM class, are defined. The price in the on-demand phase of all resource types can be doubly increased, with probability 0.1, from the price defined in Table 2. With probability 0.9, the price in the on-demand phase remains unchanged. For the demand uncertainty, the actual required number of VMs in the second provisioning stage of VM class (i.e., I_1 and I_2) varies from 1 to 50. The demand of one VM class is assumed to be the same as the other class. Three distributions of demand are considered in the experiment, namely discrete normal distribution, uniform distribution, and distribution from test data. Means of both normal and uniform distributions are set to 25.50. The variance of normal distribution is set to 6, while the variance of uniform distribution is 208.25. The last distribution is generated from the logged data obtained from Institute of High Performance Computing (IHPC) in Singapore. The data are collected from the actual usage of shared computing resources located in the computer cluster maintained by IHPC. The probability distribution (with mean = 21.64 and variance = 389.93) of the resource usage representing the number of required VMs is derived as shown in Fig. 6.

7.2 Case Study: Two Provisioning Stage Problem

7.2.1 Cost Structure

First, the cost structure to provision resources is studied. To ease the illustration, this study considers only single VM class I_1 and single cloud provider J_2. The number of required VMs (i.e., demand) is varied following the normal distribution. In Fig. 7, given different number of reserved VMs, cost in the first stage called *first stage cost* (which is actually reservation cost), cost in the second stage called *second stage cost* including expending and on-demand costs, and total cost, are presented. As expected, the first stage cost increases, as the number of reserved VMs increases. However, the second stage cost decreases after the demand is realized, since the cloud consumer needs smaller number of VMs provisioned by on-demand plan. In this case, the

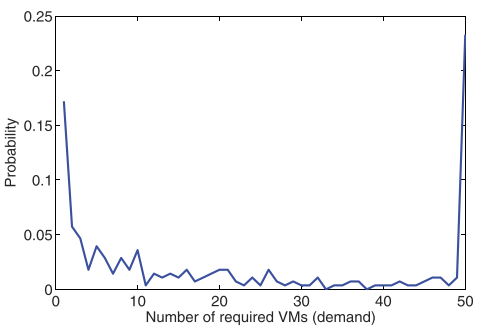


Fig. 6. The probability distribution of the real data.

5. We omit the power consumption of cooling system as it is the fixed cost of private cloud in the organization.

6. The charge of inbound data transfer offered by Amazon EC2 has been free through June 30, 2010.

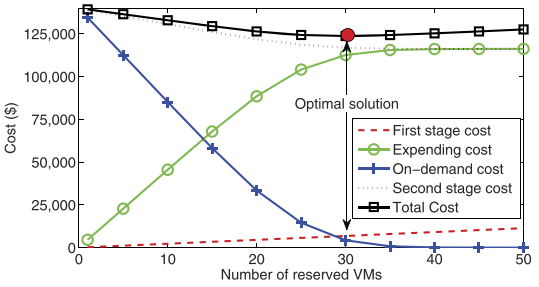


Fig. 7. The optimal solution in a simple cloud computing environment.

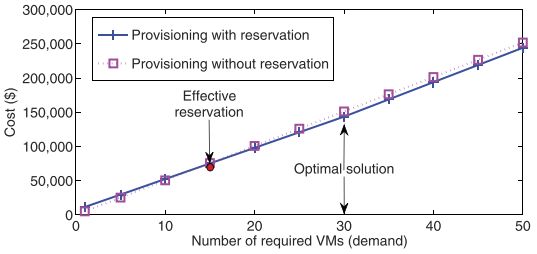


Fig. 8. Comparison between total costs of resource provision with and without reservation.

optimal number of reserved VMs can be determined to be 30 reserved VMs as shown in Fig. 7, which is the point that the total cost is minimum. Clearly, even in this small setting (one VM class and one provider), the optimal solution is not trivial to obtain due to the demand uncertainty. Therefore, the OCRP algorithm would be required to guarantee the minimum cost to the consumer.

Given the optimal reservation of 30 VMs as shown in Fig. 7, the comparison between resource provisioning with and without reservation can be made as illustrated in Fig. 8. Without reservation, the number of VMs is to dynamically provision resources in the second stage by only purchasing resources in the on-demand plan. Given different demands (or realization of required number of VMs) from 1 to 50, the cost in resource provision with reservation becomes cheaper than that without reservation due to the discounted price of processing time. However, the cost with reservation may not always be the cheapest. As shown in Fig. 8, the total cost in the resource provision without reservation is lower than the other one until the demand is 15 in which the effective reservation begins. This fact indicates that even if the solution is optimal, it cannot guarantee the best solution in all realizations of observed parameters. Therefore, the effective way to tackle the uncertainty is not to search for the best solution for every possible situation happening in the future, but to obtain the solution which is able to minimize the tradeoff between advance reservation and on-demand provision while the uncertainty is carefully considered.

7.2.2 Impact of Probability Distributions

For the next experiments, all parameters of the cloud computing environment are applied. The deterministic equivalent formulation derived in Section 4.2 is implemented and solved by GLPK.

The stochastic effect of demand under different probability distributions is investigated. In Table 3, the number of reserved VMs (indicated as label N.R. in Table 3) provisioned in the first stage is presented. Also, the expected second stage costs incurred due to different probability distributions are shown. The costs include the reservation cost (R.C.), expending cost (E.C.), on-demand cost (O.C.), oversubscribed cost (O.S.C.), and total cost. The probability distributions include normal distribution (Norm), uniform distribution (Uniform), and distribution from test data. Furthermore, three variances of the normal distribution are considered, namely variance = 4 (Norm v4), 6 (Norm v6), and 8 (Norm v8).

In Table 3, we observe that as the variance increases the number of VMs also needs to be increased. Therefore, the number of reserved VMs in the test data is highest, while that in Norm v4 is lowest. Larger variance increases the chance that the demand will be smaller or larger than the mean. Consequently, Norm v8 incurs more total cost and reserves more VMs than those in Norm v4 and Norm v6. Again, the increment of number of reserved VMs can ensure that the on-demand cost can be minimized.

7.2.3 Comparison with Other Provisioning Algorithms

Next, the comparison between provisioning algorithms is performed. The algorithms include the proposed OCRP, expected-value of uncertainty provisioning (EVU), maximum advance reservation provisioning (MaxRes), and nonreservation provisioning (NoRes) algorithms. EVU uses the average values of uncertainty parameters and solves them by a traditional deterministic program. MaxRes reserves the maximum number of available VMs, while NoRes does not reserve any resources. Both MaxRes and NoRes also apply the traditional deterministic program for allocating VMs to cloud providers. All algorithms with the defined input parameters are coded and solved by GLPK. The given distributions are applied to the possible scenarios of demand and price, respectively. The solution obtained from each solved algorithm yields the number of reserved VMs (N.R.) and the allocation of VMs to providers. Then, a simulation program is developed to evaluate the solution of

TABLE 3
Number of Reserved VMs and Costs Given Different Probability Distributions

Distribution	N.R.	Expected costs				
		R.C.	E.C.	O.C.	OS.C.	Total
Norm v4	56	$16,173.50	$236,495.21	$7,933.96	$1,528.69	$260,602.67
Norm v6	59	$16,706.00	$233,807.69	$11,427.56	$2,361.33	$261,941.25
Norm v8	62	$17,338.50	$231,890.24	$14,200.54	$3,220.51	$263,429.28
Uniform	77	$20,430.50	$230,305.62	$19,414.95	$7,168.24	$270,151.07
Test data	95	$23,825.50	$203,197.30	$9,387.16	$12,815.84	$236,409.96

TABLE 4
Number of Reserved VMs and Average Costs Given Different Resource Provisioning Algorithms

Algorithm	N.R.	Average costs				
		R.C.	E.C.	O.C.	OS.C.	Total
OCRP	59	$16,706.00	$232,222.94	$11,044.30	$2,438.25	$259,973.24
EVU	52	$15,463.50	$219,197.27	$25,785.50	$1,549.33	$260,446.27
MaxRes	100	$28,783.50	$241,430.05	$0.00	$13,684.96	$270,213.55
NoRes	0	$0.00	$0.00	$312,308.08	$0.00	$312,308.08

each algorithm. The simulation contains 1,000 iterations. In each iteration, the random number is uniformly selected from $[0, 1]$. Next, the scenario is derived by the inverse transformation method given the random number and cumulative probability distributions of the scenarios. The provisioning costs incurred by purchasing the provisioning plans given by the solution of each algorithm are recorded. After finishing the last iteration, the simulation calculates the average costs as presented in Table 4. The costs include reservation cost (R.C.), expending cost (E.C.), on-demand cost (O.C.), oversubscribed cost (OS.C.), and total cost.

In Table 4, the proposed OCRP achieves the lowest total cost, while NoRes yields the highest total cost due to the highest on-demand cost. The OCRP algorithm reserves 59 VMs (including both classes I_1 and I_2). Although MaxRes reserves 100 VMs (50 per VM class) to entirely avoid the higher cost in the on-demand plan, it still incurs much higher cost than that of OCRP. Additionally, MaxRes incurs the highest oversubscribed cost since the reserved resources are unnecessarily overprovisioned. EVU incurs the total cost lower than those of MaxRes and NoRes algorithms. Although the oversubscribed cost of OCRP is higher than that of EVU, the on-demand cost of the OCRP algorithm is much lower. Again, it is possible that the on-demand cost can increase due to the price uncertainty. As a result, the diminution of the on-demand cost is more important. The result of this experiment shows the balance between the number of provisioning resources to be acquired in the first and second stages in which OCRP can provide the most optimal tradeoff.

7.2.4 Bender Decomposition

Fig. 9 shows the bound convergence obtained by solving Bender decomposition algorithm. The adjustment of lower and upper bounds is performed in each iteration. At iteration $\nu = 42$, algorithm converges. The optimal solution obtained from the decomposition is the same as one obtained by solving DEF without decomposition. We observe that the subproblems can be solved efficiently due to their smaller number of variables and parallelization.

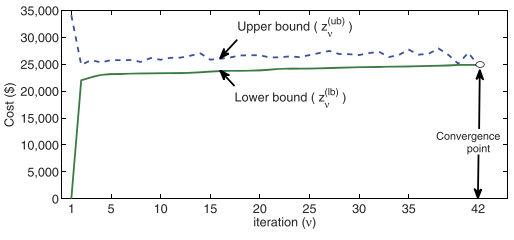


Fig. 9. Convergence of the upper and lower bounds by applying the Benders decomposition.

However, solving the master problem requires substantial amount of time since more Benders cuts have to be added.

7.3 Case Study: Twelve Provisioning Stage Problem

The SAA approach presented in Section 6 is studied. In this 12-PSP, only provider J_2 and J_3 are considered. It is assumed that the resource price is stable and the demand varies within set $\{10, 20, 30, 40, 50\}$ (i.e., $5^{12} = 244{,}140{,}625$ scenarios for the 12 stages). For sampling data, four sample sizes are determined, namely $N \in \{200, 500, 600, 750\}$. Then, the Monte Carlo sampling technique generates demand realizations for five batches per each size, i.e., $M = 5$. Lower bound estimate of each sample size is calculated by solving five SAA problems with respective batches. For upper bound estimate, the solution obtained from each solved SAA problem is fixed to another new SAA problem whose sample size is 750, i.e., $\tilde{N} = 750$. Ten batches of the new SAA problem are constructed and solved, i.e., $\tilde{M} = 10$. Then, the solutions obtained from the ten batches can be calculated as the upper bound estimate.

In Table 5, the estimates of SAA lower and upper bounds are presented. The optimal solution is found in the sample size of 750. From this optimal solution, advance reservations with only the 6-month reservation contract are used in only T_1 and T_7. That is, 10 VMs will be reserved to provider J_2 in stages T_1 and T_7 each and 30 VMs will be reserved to

TABLE 5
Estimation of Lower and Upper Bounds of Provisioning Costs in the 12 Provisioning Stage Problem

Sample size	Lower bound	Upper bound
200	$306,878.48 ± 2,385.37	$308,127.28 ± 664.16
		$308,532.91 ± 706.13
		$307,867.89 ± 848.34
		$308,754.77 ± 1,054.02
		$308,343.21 ± 691.58
500	$308,503.54 ± 1,227.10	$309,279.03 ± 718.57
		$308,692.47 ± 846.25
		$307,604.94 ± 798.90
		$308,813.45 ± 1,128.59
		$308,376.38 ± 863.05
600	$308,860.94 ± 601.41	$308,319.20 ± 435.36
		$308,637.95 ± 994.99
		$307,997.97 ± 1,602.84
		$307,754.95 ± 677.16
		$308,654.15 ± 776.73
750	$307,843.31 ± 813.70	$308,847.21 ± 1,281.17
		$308,780.37 ± 1,038.88
		$308,147.03 ± 1,183.10
		$308,699.35 ± 742.89
		$308,454.52 ± 852.28

provider J_3 in stages T_1 and T_7 each. We conclude that 40 reserved VMs are only needed in stages T_1 and T_7 each. This solution can avoid the higher on-demand cost, since only 10 more VMs could be provisioned with on-demand plan in any provisioning stages.

7.4 Discussion

7.4.1 Experimental Results

1. *Balance of costs.* We observe that the cloud broker with OCRP will minimize on-demand cost rather than the oversubscribed cost. Since resource pricing in the on-demand plan is higher and possibly increased by cloud providers, the reservation plan is more attractive by the cloud broker. However, reserving too many VMs may not be optimal (e.g., as that of MaxRes from Table 4). Therefore, the tradeoff between on-demand and oversubscribed costs needs to be adjusted in which OCRP can optimally perform.
2. *Virtual machine outsourcing.* As presented in Section 7.2, the VM outsourcing from a private cloud to a public cloud provider (or public cloud) shows interesting result. In the experiment, the private cloud fully utilizes its own resources. Then, extra VMs can be spilled over to public clouds. Purchasing and deploying new hardware to a private cloud may not be an optimal solution, since the total cost of ownership (TCO) must be considered. To reduce this TCO, workload outsourcing is the attractive choice which is shown from our evaluation.

7.4.2 Implementation Issues

1. *Multiple provisioning stages planning issue.* As shown in Section 7.3, the OCRP algorithm can be applied to multiple provisioning stages representing long-term planning. Since the optimal solution of the first provisioning stage depends on multiple probability distributions describing the uncertainty occurring in consequent time epochs, multiple stages planning is needed. For example, the workload of some online souvenir shopping websites could be dramatically increased in the high-season consisting of many time periods in a year (e.g., Christmas Day, Valentine's Day, etc.). As a result, the websites should provision resources by considering multiple time epochs (i.e., provisioning stages) in advance, while reservation contracts offered by cloud providers can be taken into account to reduce the provisioning cost.
2. *Use of decomposition method.* The use of decomposition method for OCRP has to be carefully considered, since the formulation of the OCRP algorithm is a pure integer program which is the NP-hard problem [5]. Although the subproblems can be solved in parallel, the master problem with the additional Benders cuts requires considerable computational time. The performance improvement of the decomposition algorithm will be considered in the future work.
3. *Benefit of SAA.* Sample-average approximation method can overcome the provisioning problems with a large set of scenarios which are impossible to solve with deterministic equivalent formulation directly. In contrast, the approximation algorithm with estimation of SAA lower and upper bounds can yield tolerably solutions while the problems can be practically solved in timely manner.
4. *Limitation of stochastic programming.* Stochastic programming does not address the method to obtain appropriate probability distributions describing uncertainty (i.e., distributions of scenarios Ω). However, this limitation can be alleviated by applying variance reduction techniques (e.g., importance sampling [27]) to increase the precision of the estimates of uncertainty.

7.4.3 Future Research Direction

For the future work, scenario reduction techniques [28] will be applied to reduce the number of scenarios. In addition, the optimal pricing scheme for cloud providers with the consideration of competition in the market will be investigated.

8 Conclusion

In this paper, we have proposed an optimal cloud resource provisioning (OCRP) algorithm to provision resources offered by multiple cloud providers. The optimal solution obtained from OCRP is obtained by formulating and solving stochastic integer programming with multistage recourse. We have also applied Benders decomposition approach to divide an OCRP problem into subproblems which can be solved parallelly. Furthermore, we have applied the SAA approach for solving the OCRP problem with a large set of scenarios. The SAA approach can effectively achieve an estimated optimal solution even the problem size is greatly large. The performance evaluation of the OCRP algorithm has been performed by numerical studies and simulations. From the results, the algorithm can optimally adjust the tradeoff between reservation of resources and allocation of on-demand resources. The OCRP algorithm can be used as a resource provisioning tool for the emerging cloud computing market in which the tool can effectively save the total cost.

Acknowledgments

This work was done in the Parallel and Distributed Computing Centre (PDCC) of the School of Computer Engineering, Nanyang Technological University, Singapore. This work was supported by the projects "User and Domain Driven Data Analytics" and "Design and Analysis of Cloud Computing for Data Value Chain: Operation Research Approach," granted by the A*STAR Thematic Strategic Research Programme.

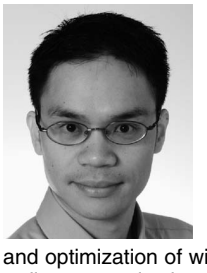
References

[1] I. Foster, Y. Zhao, and S. Lu, "Cloud Computing and Grid Computing 360-Degree Compared," *Proc. Grid Computing Environments Workshop (GCE '08)*, 2008.
[2] Amazon EC2, http://aws.amazon.com/ec2, 2012.
[3] GoGrid, http://www.gogrid.com, 2012.
[4] Amazon EC2 Reserved Instances, http://aws.amazon.com/ec2/reserved-instances, 2012.

[5] F.V. Louveaux, "Stochastic Integer Programming," *Handbooks in OR & MS,* vol. 10, pp. 213-266, 2003.
[6] A.J. Conejo, E. Castillo, and R. García-Bertrand, "Linear Programming: Complicating Variables," *Decomposition Techniques in Mathematical Programming,* chapter 3, pp. 107-139, Springer, 2006.
[7] J. Linderoth, A. Shapiro, and S. Wright, "The Empirical Behavior of Sampling Methods for Stochastic Programming," *Ann. Operational Research,* vol. 142, no. 1, pp. 215-241, 2006.
[8] G. Juve and E. Deelman, "Resource Provisioning Options for Large-Scale Scientific Workflows," *Proc. IEEE Fourth Int'l Conf. e-Science,* 2008.
[9] Z. Huang, C. He, and J. Wu, "On-Demand Service in Grid: Architecture Design, and Implementation," *Proc. 11th Int'l Conf. Parallel and Distributed Systems (ICPADS '05),* 2005.
[10] Y. Jie, Q. Jie, and L. Ying, "A Profile-Based Approach to Just-in-Time Scalability for Cloud Applications," *Proc. IEEE Int'l Conf. Cloud Computing (CLOUD '09),* 2009.
[11] Y. Kee and C. Kesselman, "Grid Resource Abstraction, Virtualization, and Provisioning for Time-Target Applications," *Proc. IEEE Int'l Symp. Cluster Computing and the Grid,* 2008.
[12] A. Filali, A.S. Hafid, and M. Gendreau, "Adaptive Resources Provisioning for Grid Applications and Services," *Proc. IEEE Int'l Conf. Comm.,* 2008.
[13] D. Kusic and N. Kandasamy, "Risk-Aware Limited Lookahead Control for Dynamic Resource Provisioning in Enterprise Computing Systems," *Proc. IEEE Int'l Conf. Autonomic Computing,* 2006.
[14] K. Miyashita, K. Masuda, and F. Higashitani, "Coordinating Service Allocation through Flexible Reservation," *IEEE Trans. Services Computing,* vol. 1, no. 2, pp. 117-128, Apr.-June 2008.
[15] J. Chen, G. Soundararajan, and C. Amza, "Autonomic Provisioning of Backend Databases in Dynamic Content Web Servers," *Proc. IEEE Int'l Conf. Autonomic Computing,* 2006.
[16] L. Grit, D. Irwin, A. Yumerefendi, and J. Chase, "Virtual Machine Hosting for Networked Clusters: Building the Foundations for Autonomic Orchestration," *Proc. IEEE Int'l Workshop Virtualization Technology in Distributed Computing,* 2006.
[17] H.N. Van, F.D. Tran, and J.-M. Menaud, "SLA-Aware Virtual Resource Management for Cloud Infrastructures," *Proc. IEEE Ninth Int'l Conf. Computer and Information Technology,* 2009.
[18] M. Cardosa, M.R. Korupolu, and A. Singh, "Shares and Utilities Based Power Consolidation in Virtualized Server Environments," *Proc. IFIP/IEEE 11th Int'l Conf. Symp. Integrated Network Management (IM '09),* 2009.
[19] F. Hermenier, X. Lorca, and J.-M. Menaud, "Entropy: A Consolidation Manager for Clusters," *Proc. ACM SIGPLAN/SIGOPS Int'l Conf. Virtual Execution Environments (VEE '09),* 2009.
[20] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," *Proc. IFIP/IEEE Int'l Symp. Integrated Network Management (IM '07),* pp. 119-128, May 2007.
[21] P. Jirutitijaroen and C. Singh, "Reliability Constrained Multi-Area Adequacy Planning Using Stochastic Programming with Sample-Average Approximations," *IEEE Trans. Power Systems,* vol. 23, no. 2, pp. 504-513, May 2008.
[22] S. Chaisiri, B.S. Lee, and D. Niyato, "Optimal Virtual Machine Placement across Multiple Cloud Providers," *Proc. IEEE Asia-Pacific Services Computing Conf. (APSCC),* 2009.
[23] GNU Linear Programming Kit (GLPK), http://www.gnu.org/software/glpk, 2012.
[24] W.-K. Mak, D.P. Morton, and R.K. Wood, "Monte Carlo Bounding Techniques for Determining Solution Quality in Stochastic Programs," *Operations Research Letter,* vol. 24, pp. 47-56, 1999.
[25] M.D. McKay, R.J. Beckman, and W.J. Conover, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics,* vol. 21, no. 2, pp. 239-245, 1979.
[26] R. Chheda, D. Shookowsky, S. Stefanovich, and J. Toscano, "Profiling Energy Usage for Efficient Consumption," *Architecture J.,* no. 18, 2008.
[27] G.B. Dantzig and G. Infangerm, "Large-Scale Stochastic Linear Programs: Importance Sampling and Benders Decomposition," *Proc. IMACS World Congress on Computation and Applied Math.,* 1991.
[28] H. Heitsch and W. Römisch, "Scenario Reduction Algorithms in Stochastic Programming," *J. Computational Optimization and Applications,* vol. 24, pp. 187-206, 2003.

Sivadon Chaisiri received the MEng degree from Kasetsart University, Bangkok, Thailand, in 2005. He is currently working toward the PhD degree at Nanyang Technological University, Singapore. His current research interests include cloud computing and distributed systems. He is a student member of the IEEE.

Bu-Sung Lee received the BSc (Hons.) and PhD degrees from the Electrical and Electronics Department, Loughborough University of Technology, United Kingdom, in 1982 and 1987, respectively. He is currently an associate professor with the School of Computer Engineering, Nanyang Technological University, Singapore. He was elected the inaugural president of Singapore Research and Education Networks (SingAREN), 2003-2007, and has been an active member of several national standards organizations, such as a board member of Asia Pacific Advanced Networks (APAN) Ltd. In 2010, he held a joint appointment as director, Service Platform Lab, HP Labs Singapore. His research interests include computer networks protocols, distributed computing, network management, and grid/cloud computing. He is a member of the IEEE.



Dusit Niyato received the BE degree from King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand, in 1999 and the PhD degree in electrical and computer engineering from the University of Manitoba, Winnipeg, Canada, in 2008. He is currently an assistant professor in the Division of Computer Communications, School of Computer Engineering, Nanyang Technological University, Singapore. His current research interests include the design, analysis, and optimization of wireless communication, smart grid systems, green radio communications, and mobile cloud computing. He is a member of the IEEE.