Profit Maximization in a Federated Cloud by Optimal Workload Management and Server Speed Setting

Keqin Li, *Fellow, IEEE*

Abstract—In an environment with multiple heterogeneous multiservers across multiple clouds and data centers, optimal workload management and server speed setting makes strong impact on the aggregated performance, cost, and profit of a cloud service provider. Such a situation provides us a more challenging opportunity to address and discuss profit maximization of a service provider in a wider scale and to generate more significant influence. In this paper, we investigate profit maximization by optimal workload management and server speed setting for multiple heterogeneous multiservers in a federated cloud or a geo-distributed data center. The heterogeneous multiservers have different sizes, speeds, power consumption models, workload, performance, costs, and profit. To conduct rigorous study, each multiserver system is modeled by an M/M/m queueing system, such that the profit of a multiserver system can be characterized analytically. We address two problems, i.e., workload management without server speed setting and workload management with server speed setting. Both problems are formulated as multi-variable optimization problems. We develop numerical algorithms to solve these problems. We also provide numerical data for the purpose of illustration. To the best of the author's knowledge, this is the first paper which analytically discusses profit maximization for multiple heterogeneous multiservers in a federated cloud or a geo-distributed data center using queueing systems.

Index Terms—Cloud service provider, federated cloud, heterogeneous multiserver system, optimal server speed setting, optimal workload management, profit maximization, queueing system

1 Introduction

1.1 Motivation

Cloud computing has the capability and characteristic of providing computing services through an on-demand, pay-per-use, and pay-as-you-go billing method. Cloud computing is a computing business model in which a cloud service provider owns, rents, operates, maintains, upgrades, and manages physical or virtual computing infrastructure and resources, and provides and delivers various services and solutions to consumers and customers, businesses and individuals. A user or subscriber accesses these resources and services over the Internet on a measurable and metered basis. Cloud services are becoming increasingly desirable for companies and industries, because they offer great advantages in terms of cost, performance, accessibility, scalability, elasticity, reliability, and sustainability.

Significant research has been conducted for the technology-related issues of cloud computing. However, it is equally important to understand the economics- and business-related issues of cloud computing. Resource utilization, performance optimization, customer satisfaction, pricing strategies, market competition, revenue maximization, server configuration, cost reduction, and profit maximization for cloud service providers have been placed as urgent research agenda [6], [22]. It has been a significant challenge for a cloud service provider to make the most efficient use of available resources, deliver the best quality of service to the users, provide the highest satisfaction of customers, attract the most consumers, be the most competitive in the market, generate the largest revenue, spend the least cost of service, receive the lowest penalty for low quality service, balance the above contradictory considerations, and ultimately, earn the highest profit.

Cloud federation (also called federated cloud, cloud of clouds, InterCloud, and geo-distributed data center) requires one cloud service provider to lease/rent computing resources to/from another provider [2], [3]. These resources become temporary or permanent extension of a provider's cloud computing environment, depending on the specific federation agreement between providers. Cloud federation offers two solid and substantial benefits to cloud service providers. First, it allows providers to earn revenue from idle or underutilized computing resources. Second, it enables providers to expand their geographic footprints and accommodate sudden spikes in demand without actually building new points-of-presence. Cloud service providers strive to make all aspects of cloud federation, from resource insourcing/outsourcing to billing support systems and customer support, all transparent to consumers. A cloud service provider can also establish extensions of its customer-facing service-level agreements into its partner providers' data centers when federating cloud services with partners [1].

• *The author is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA. E-mail: lik@newpaltz.edu.*

Manuscript received 20 May 2021; revised 8 Sept. 2021; accepted 4 Nov. 2021. Date of publication 9 Nov. 2021; date of current version 8 Sept. 2022.
(Corresponding author: Keqin Li.)
Recommended for acceptance by P. D. Yoo.
Digital Object Identifier no. 10.1109/TSUSC.2021.3126666

A federated cloud is able to make use of computing infrastructures across multiple geographically distributed clouds and data centers. Such cloud federation integrates heterogeneous resources and computing power, and furthermore, provides more powerful and flexible services to various service requests from scientific, business, and industrial application domains. In such an environment with multiple heterogeneous multiservers across multiple clouds and data centers, optimal workload management and server speed setting makes strong impact on the aggregated performance, cost, and profit of a cloud service provider. Such a situation provides us a more challenging opportunity to address and discuss profit maximization of a service provider in a wider scale and to generate more significant influence.

1.2 Related Work

Profit maximization for cloud service providers has been a very active and productive research area in the last few years. Refs. [5], [6], [7], [16], [29] give some recent comprehensive surveys.

Profit maximization for federated clouds and geo-distributed data centers has been investigated extensively by a number of researchers. In [9], the authors presented ways for a provider to enhance its profit in cloud federation, e.g., renting (i.e., outsourcing) resources from other providers, leasing (i.e., insourcing) free resources to other providers, and shutting down unused resources to save power. In [14], the authors developed a customer satisfaction-aware algorithm based on ant-colony optimization for geo-distributed data centers, by formulating profit maximization as an optimization problem under customer satisfaction and data center constraints, and maximizing net profit by dispatching service requests to proper data centers and generating appropriate amount of VMs to meet customer satisfaction. In [21], the authors designed an energy-efficient, profit- and cost-aware request dispatching and resource allocation algorithm to maximize the net profit of a cloud service provider operating geographically distributed data centers in a multi-electricity-market environment, by formulating the net profit maximization issue as a constrained optimization problem, using a unified task model capturing multiple cloud layers (e.g., SaaS, PaaS, IaaS), judiciously distributing service requests to data centers, powering on/off an appropriate number of servers, and allocating server resources to dispatched requests. In [25], the authors described an algorithm for VM provisioning in a federated cloud environment, attempting to improve a cloud provider's profit. In [26], the authors formulated a problem for cloud service providers owning multiple geo-distributed clouds to decide their computing resource prices as a game of resource pricing. In [27], the authors established policies that help in the decision-making process to enhance profit, utilization, and QoS in a cloud federation environment. In [30], the authors constructed a business-oriented federated cloud computing model, where multiple independent infrastructure providers can cooperate seamlessly to provide scalable IT infrastructures and QoS-assured hosting services for real-time online interactive applications with a business layer that can trigger on-demand resource provisioning across multiple infrastructure providers and help to maximize customer satisfaction, business benefits, and resources usage.

In contrast to profit maximization for federated clouds, several researchers have considered profit maximization for competing cloud service providers in a competitive cloud computing environment, typically by using a game theoretic approach. In [8], the authors conducted an in-depth game theoretic study of a competition market with multiple competing cloud providers, where each cloud provider is modeled by an M/M/1 queue with fixed service charge. Furthermore, the market share of each cloud provider is proportional to its attraction, which is inversely proportional to the service time and service price. (In our study, we use M/M/m queueing systems, and service charge depends on the amount of service and the quality of service. Furthermore, the workload of each multiserver is determined by global optimization.) In [13], the authors proposed a price bidding mechanism for multi-attribute cloud-computing resource provision from the perspective of a non-cooperative game, in which the information of each player (one customer and multiple providers) is incomplete to others and each player wishes to maximize his own benefit. In [17], the author studied a competitive cloud computing market, in which each cloud service provider can adjust its server size, server speed, and service charge function, such that its profit is maximized. Whenever a cloud service provider takes an action, the market is re-stabilized, such that all competing cloud service providers have the same expected customer satisfaction, where the satisfaction of a customer includes two aspects, i.e., satisfaction on the price of service and satisfaction on the quality of service. It was shown that such a non-cooperative game has a Nash equilibrium. In [20], the authors focused on request migration strategies among multiple servers for load balancing and considered the problem from a game theoretic perspective and formulated it as a non-cooperative game among the multiple servers. In [28], the authors formulated the competition among cloud providers as a non-cooperative stochastic game, which is modeled as a Markov decision process whose solution is a Markov perfect equilibrium, where the providers propose the price policy simultaneously.

1.3 Our Contributions

In this paper, we investigate profit maximization by optimal workload management and server speed setting for multiple heterogeneous multiservers in a federated cloud or a geo-distributed data center. The heterogeneous multiservers have different sizes, speeds, power consumption models, workload, performance, costs, and profit. To conduct rigorous study, each multiserver system is modeled by an M/M/m queueing system, such that the profit of a multiserver system can be characterized analytically. We address two problems, i.e., workload management without server speed setting and workload management with server speed setting. Both problems are formulated as multi-variable optimization problems. We develop numerical algorithms to solve these problems. We also provide numerical data for the purpose of illustration. To the best of the author's knowledge, this is the first paper which analytically discusses profit maximization for multiple heterogeneous multiservers in a

TABLE 1
Summary of Notations and Definitions

Notation	Definition
Queueing Model	
n	the number of heterogeneous multiserver systems in a federated cloud
S_i	the ith multiserver system, represented by an M/M/m queueing model
m_i	the number of identical servers in S_i (i.e., the size of S_i)
λ_i	the arrival rate of a Poisson stream of service requests to S_i (measured by tasks/second)
λ	$= \lambda_1 + \lambda_2 + \cdots + \lambda_n$
r	task execution requirement (measured by the number of billion processor cycles)
s_i	task execution speed of S_i (measured by GHz = billion processor cycles per second)
x_i	$= r/s_i$, task execution times on servers of S_i (measured by seconds)
μ_i	$= 1/\bar{x}_i = s_i/\bar{r}$, the average service rate of S_i (measured by tasks/second)
ρ_i	$= \lambda_i/m_i\mu_i = \lambda_i\bar{x}_i/m_i = \lambda_i/m_i \cdot \bar{r}/s_i$, the server utilization of S_i
$p_{i,k}$	the probability that there are k service requests (in waiting or being processed) in S_i
$P_{q,i}$	the probability of queueing in S_i
W_i	the waiting time of S_i (measured by seconds)
T_i	the response time of S_i (measured by seconds)
Power Consumption Models	
P_i	$= \lambda_i\bar{r}\xi_i s_i^{\alpha_i - 1} + m_i P_i^*$ for the *idle-speed model*, and $= m_i(\xi_i s_i^{\alpha_i} + P_i^*)$ for the *constant-speed model*
ξ_i, α_i	parameters of the power consumption models
P_i^*	static power consumption of S_i (measured by Watts)
Profit Model	
C_i	the expected charge to a service request processed on S_i (measured by cent)
s_0	a parameter indicating the expectation and satisfaction of a consumer
a	the service charge per unit amount of service (measured by cent per billion processor cycles)
c	the service level agreement
d	the degree of penalty of breaking the service level agreement (measured by cent/second)
β_i	the cost of infrastructure facilities of one server per unit of time in S_i (measured by cent/second)
γ	the cost of energy consumption per Watt and per unit of time (measured by cent/Watt/second)
G_i	the expected net business gain in one unit of time of S_i (measured by cent/second)

federated cloud or a geo-distributed data center using queueing systems.

The remainder of the paper is summarized as follows. In Section 2, we present the preliminaries, including models and problems. In Section 3, we discuss workload management without server speed setting. In Section 4, we discuss workload management with server speed setting. In Section 5, we discuss optimal multiserver selection. We conclude the paper in Section 6.

2 The Preliminaries

In this section, we present our models and problems.

2.1 Models

A cloud computing environment or a data center serves users' service requests by using multiple heterogeneous multiserver systems. A federated cloud or a geo-distributed data center can maintain a collection of n heterogeneous multiserver systems $S_1, S_2, \ldots, S_n$, which are different in size, speed, power consumption model, workload, performance, cost, and profit. Our models include a multiserver queueing model, two power consumption models, and a profit model defined in [4], [17]. Due to space limitation and to avoid duplication, detailed descriptions of these models are omitted in this paper. Instead, we give highlights of these models below. Table 1 gives a summary of all the notations and their definitions used in these models. (Note: $\bar{y}$ represents the expectation of a random variable y.)

The Queueing Model. A multiserver system is abstracted as an M/M/m queueing model, which is extensively used in theory and practice [10], [15]. Each S_i is characterized by its server size m_i (i.e., the number of servers), server speed s_i (i.e., the task execution speed), and arrival rate λ_i (i.e., the number of arrival tasks per second). Tasks are specified by a random variable r, which is the task execution requirement. The server utilization of S_i is $\rho_i = \lambda_i/m_i\mu_i$, where $\mu_i = 1/\bar{x}_i$ is the average service rate of S_i, and $x_i = r/s_i$ is the random task execution time on S_i. The waiting time W_i is a random variable. The response time of S_i is a random variable $T_i = W_i + r/s_i$.

The Power Consumption Models. Let ξ_i and α_i be some constants that determine the dynamic power consumption of S_i, and P_i^* be the static power consumption of S_i. In the *idle-speed model*, we have $P_i = \lambda_i\bar{r}\xi_i s_i^{\alpha_i - 1} + m_i P_i^*$. In the *constant-speed model*, we have $P_i = m_i(\xi_i s_i^{\alpha_i} + P_i^*)$. The difference between the two models is that when a server S_i is idle, it runs at speed zero and does not consume dynamic power $\xi_i s_i^{\alpha_i}$ in the idle-speed model, while still runs at speed s_i and consumes dynamic power in the constant-speed model.

We would like to mention that energy efficiency has been an important component in the optimization of workload forecasting, workflow management, task scheduling, load balancing, resource allocation, and server configuration in general distributed computing systems [11], [12], [18], [19], [23], [24].

The Profit Model. The *service charge function* of S_i specified by s_0, a, c, d for a service request with execution requirement r and response time T_i is

$$C_i(r,T_i)=\begin{cases} a_i r, \text{ if } 0 \le T_i \le (c_i/s_{0,i})r; \\ a_i r - d_i(T_i-(c_i/s_{0,i})r), \\ \qquad \text{if } (c_i/s_{0,i})r < T_i \le (a_i/d_i + c_i/s_{0,i})r; \\ 0, \quad \text{if } T_i > (a_i/d_i + c_i/s_{0,i})r; \end{cases}$$

which is a random variable (see [4], [17] for elaboration and justification). It has been proven in [4] that the expected charge to a service request processed on S_i, i.e., $C_i = \overline{C_i(r,T_i)}$, is

$$C_i = a\bar{r}\left(1 - P_{q,i} \times \frac{1}{(m_i s_i - \lambda_i \bar{r})(c/s_0 - 1/s_i) + 1} \times \frac{1}{(m_i s_i - \lambda_i \bar{r})(a/d + c/s_0 - 1/s_i) + 1}\right),$$

where $P_{q,i} = p_{i,m_i}/(1-\rho_i)$ and $p_{i,m_i} = p_{i,0}(m_i\rho_i)^{m_i}/m_i!$. The above closed-form expression makes it possible to take an analytical approach to profit maximization.

The most important metric in this paper is the expected *net business gain* (i.e., the *profit*) of a multiserver system S_i in one unit of time

$$G_i = \lambda_i C_i - (\beta_i m_i + \gamma P_i),$$

which is actually expressed as the revenue minus the cost [4]. The above equation is

$$G_i = \lambda_i C_i - (\beta_i m_i + \gamma(\lambda_i \bar{r}\xi_i s_i^{\alpha_i - 1} + m_i P_i^*)),$$

for the idle-speed model, and

$$G_i = \lambda_i C_i - (\beta_i m_i + \gamma m_i(\xi_i s_i^{\alpha_i} + P_i^*))$$

for the constant-speed model.

The overall profit of a federated cloud or a geo-distributed data center with n heterogeneous multiserver systems $S_1, S_2, \ldots, S_n$ is $G = G_1 + G_2 + \cdots + G_n$, which is

$$G = \sum_{i=1}^{n}(\lambda_i C_i - (\beta_i m_i + \gamma(\lambda_i \bar{r}\xi_i s_i^{\alpha_i - 1} + m_i P_i^*))),$$

for the idle-speed model, and

$$G = \sum_{i=1}^{n}(\lambda_i C_i - (\beta_i m_i + \gamma m_i(\xi_i s_i^{\alpha_i} + P_i^*))),$$

for the constant-speed model. It is the quantity G that we need to maximize.

2.2 Problems

The two multi-variable optimization problems to be solved in this paper are formally defined as follows.

Problem 1 (*Workload management without server speed setting*): Given certain workload specified by λ and $\bar{r}$, n heterogeneous multiserver systems $S_1, S_2, \ldots, S_n$, where S_i is specified by m_i, s_i, ξ_i, α_i, P_i^*, for all $1 \le i \le n$, a service charge function specified by s_0, a, c, d, cost parameters β_i and γ, find a workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$, such that G is maximized, subject to the constraint that $\lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$.

Problem 2 (*Workload management with server speed setting*): Given certain workload specified by λ and $\bar{r}$, n heterogeneous multiserver systems $S_1, S_2, \ldots, S_n$, where S_i is specified by m_i, ξ_i, α_i, P_i^*, for all $1 \le i \le n$, a service charge function specified by s_0, a, c, d, cost parameters β_i and γ, find a workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and a server speed setting $(s_1, s_2, \ldots, s_n)$, such that G is maximized, subject to the constraint that $\lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$.

The two problems are solved in Sections 3 and 4 respectively.

3 Workload Management Without Server Speed Setting

In this section, we address the problem of workload management without server speed setting.

3.1 Analysis

To solve this multi-variable optimization problem, we view $G(\lambda_1, \lambda_2, \ldots, \lambda_n)$ as a function of $\lambda_1, \lambda_2, \ldots, \lambda_n$. We can maximize $G(\lambda_1, \lambda_2, \ldots, \lambda_n)$ subject to the constraint $F(\lambda_1, \lambda_2, \ldots, \lambda_n) = \lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$ by using the following Lagrange multiplier system (a standard method in multivariable calculus)

$$\nabla G(\lambda_1, \lambda_2, \ldots, \lambda_n) = \phi \nabla F(\lambda_1, \lambda_2, \ldots, \lambda_n),$$

that is

$$\frac{\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n)}{\partial \lambda_i} = \phi \frac{\partial F(\lambda_1, \lambda_2, \ldots, \lambda_n)}{\partial \lambda_i} = \phi,$$

for all $1 \le i \le n$, where ϕ is a Lagrange multiplier. It is clear that for all $1 \le i \le n$, we have

$$\frac{\partial G}{\partial \lambda_i} = C_i + \lambda_i \frac{\partial C_i}{\partial \lambda_i} - \gamma \bar{r}\xi_i s_i^{\alpha_i - 1},$$

for the idle-speed model, and

$$\frac{\partial G}{\partial \lambda_i} = C_i + \lambda_i \frac{\partial C_i}{\partial \lambda_i},$$

for the constant-speed model. For convenience, we rewrite C_i as

$$C_i = a\bar{r}\left(1 - \frac{P_{q,i}}{D_1 D_2}\right),$$

where $D_1 = (m_i s_i - \lambda_i \bar{r})(c/s_0 - 1/s_i) + 1$, and $D_2 = (m_i s_i - \lambda_i \bar{r})(a/d + c/s_0 - 1/s_i) + 1$. Therefore, we obtain

$$\begin{aligned} \frac{\partial C_i}{\partial \lambda_i} &= -a\bar{r}\left(\frac{1}{D_1 D_2}\cdot\frac{\partial P_{q,i}}{\partial \lambda_i} - \frac{P_{q,i}}{D_1^2 D_2}\cdot\frac{\partial D_1}{\partial \lambda_i} - \frac{P_{q,i}}{D_1 D_2^2}\cdot\frac{\partial D_2}{\partial \lambda_i}\right) \\ &= -a\bar{r}\left(\frac{1}{D_1 D_2}\cdot\frac{\partial P_{q,i}}{\partial \lambda_i} + \bar{r}(c/s_0 - 1/s_i)\frac{P_{q,i}}{D_1^2 D_2} \right. \\ &\qquad \left. + \bar{r}(a/d + c/s_0 - 1/s_i)\frac{P_{q,i}}{D_1 D_2^2}\right), \end{aligned}$$

for all $1 \le i \le n$. To further calculate $\partial C_i/\partial \lambda_i$, we need to examine $\partial P_{q,i}/\partial \lambda_i$. Recall that ([15], p. 102)

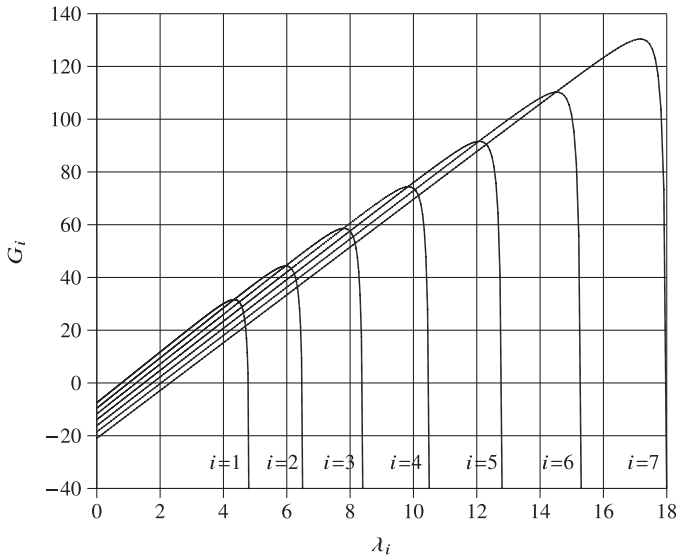

Fig. 1. Profit G_i versus task arrival rate λ_i (idle-speed model).

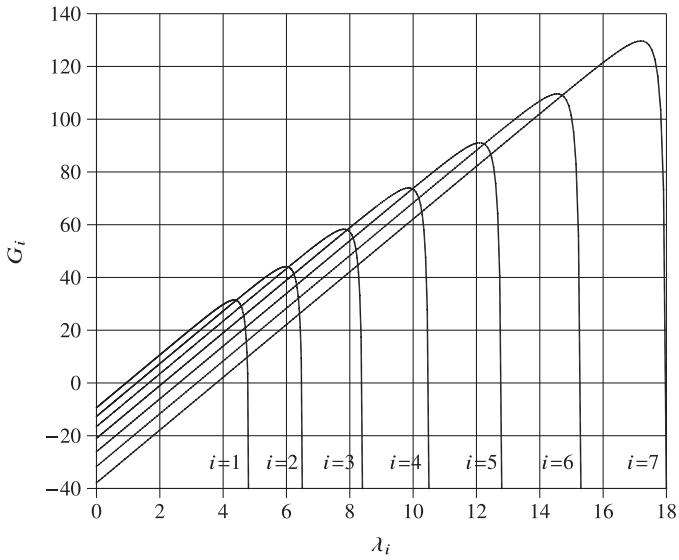

Fig. 2. Profit G_i versus task arrival rate λ_i (constant-speed model).

$$P_{q,i} = \frac{m_i^{m_i}}{m_i!} \cdot p_{i,0} \frac{\rho_i^{m_i}}{1-\rho_i}.$$

Hence, we get

$$\frac{\partial P_{q,i}}{\partial \lambda_i} = \frac{m_i^{m_i}}{m_i!} \left(\frac{\rho_i^{m_i}}{1-\rho_i} \cdot \frac{\partial p_{i,0}}{\partial \lambda_i} + p_{i,0} \cdot \frac{\bar{r}}{m_i s_i} \cdot \frac{m_i \rho_i^{m_i-1}(1-\rho_i) + \rho_i^{m_i}}{(1-\rho_i)^2} \right),$$

where we notice that $\partial \rho_i / \partial \lambda_i = \bar{r}/m_i s_i$, for all $1 \le i \le n$. To further calculate $\partial P_{q,i}/\partial \lambda_i$, we need to examine $\partial p_{i,0}/\partial \lambda_i$. Recall that

$$p_{i,0} = \left(\sum_{k=0}^{m_i-1} \frac{m_i^k}{k!} \rho_i^k + \frac{m_i^{m_i}}{m_i!} \cdot \frac{\rho_i^{m_i}}{1-\rho_i} \right)^{-1}.$$

Thus, we have

$$\frac{\partial p_{i,0}}{\partial \lambda_i} = -p_{i,0}^2 \left(\sum_{k=1}^{m_i-1} \frac{m_i^{k-1}}{(k-1)!} \rho_i^{k-1} + \frac{m_i^{m_i-1}}{m_i!} \cdot \frac{m_i \rho_i^{m_i-1}(1-\rho_i) + \rho_i^{m_i}}{(1-\rho_i)^2} \right) \frac{\bar{r}}{s_i},$$

for all $1 \le i \le n$.

3.2 Algorithms

An effective and efficient method is required to find $\lambda_1, \lambda_2, \ldots, \lambda_n$ and ϕ, which satisfy the equation $\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n)/\partial \lambda_i = \phi$, for all $1 \le i \le n$, and $F(\lambda_1, \lambda_2, \ldots, \lambda_n) = \lambda$.

Consider $n = 7$ heterogeneous multiserver systems $S_1, S_2, \ldots, S_n$, where the parameters of S_i are $m_i = 3 + i$, $s_i = 1.1 + 0.1i$ GHz, $\xi_i = 3.2 + 0.2i$, $\alpha_i = 3.4 - 0.1i$, and $P_i^* = 4.5 + 0.5i$ Watts, for all $1 \le i \le n$. Throughout this paper, we use the following parameter setting: $\bar{r} = 1.0$ billion processor cycles, $s_0 = 1.0$ GHz, $a = 10.0$ cents per billion processor cycles, $c = 3.0$, $d = 1.0$ cent per second, $\beta_i = 1.5$ cents per second, $\gamma = 0.075$ cents per Watt and per second.

In Figs. 1 and 2, we show the profit G_i versus the task arrival rate λ_i, for all $1 \le i \le n$, and the idle-speed model and the constant-speed model respectively. It is observed that as λ_i increases, G_i increases almost linearly. The reason is that the revenue increases linearly with λ_i. However, as λ_i approaches its maximum value, i.e., $m_i s_i/\bar{r}$, G_i drops very quickly. The reason is that the M/M/m queueing system for S_i becomes saturated, and the task response time T_i increases rapidly. This results in free service to virtually all service requests with little revenue and no profit.

In Figs. 3 and 4, we show the value of $\partial G/\partial \lambda_i$ versus the task arrival rate λ_i, for all $1 \le i \le n$, and the idle-speed model and the constant-speed model respectively. It is observed that $\partial G/\partial \lambda_i$ is a decreasing function of λ_i. This is a key fact which results in an efficient method to find $\lambda_1, \lambda_2, \ldots, \lambda_n$ and ϕ. The method is essentially the binary search method for both $\lambda_1, \lambda_2, \ldots, \lambda_n$ and ϕ.

A complete description of the method to find λ_i is given in Algorithm 2. For a given ϕ, we can find λ_i such that $\partial G/\partial \lambda_i = \phi$, for all $1 \le i \le n$, by using binary search (lines 2–9) of λ_i in the range $[0, m_i s_i/\bar{r})$ (line 1). (Note: $|I|$ is the length of an interval I. mid(I) is the middle point of I.)

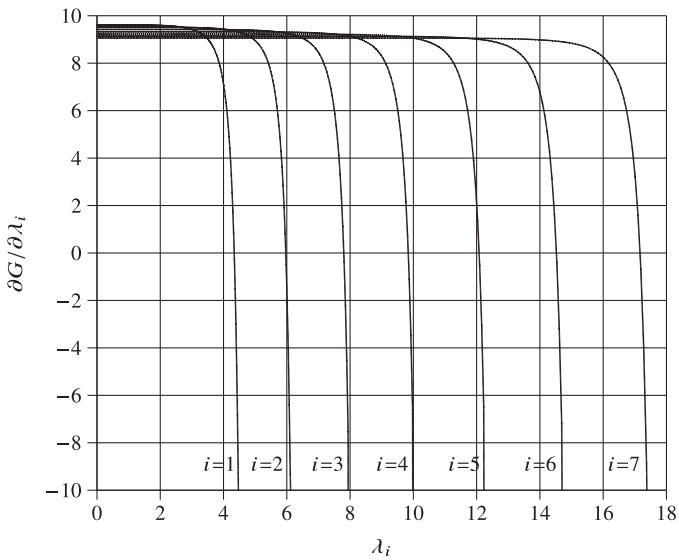

Fig. 3. $\partial G/\partial \lambda_i$ versus task arrival rate λ_i (idle-speed model).

Algorithm 1. Optimal Workload Management Without Server Speed Setting

Input: Workload specified by λ and $\bar{r}$, n heterogeneous multi-server systems $S_1, S_2, \ldots, S_n$, where S_i is specified by m_i, s_i, ξ_i, α_i, P_i^*, for all $1 \le i \le n$, a service charge function specified by s_0, a, c, d, cost parameters β and γ.
Output: A workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$, such that G is maximized, subject to the constraint that $\lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$.

if (the power consumption model is idle-speed model) **then** (1)
  Calculate $b_1, b_2, \ldots, b_n$ and $\lambda_1^*, \lambda_2^*, \ldots, \lambda_n^*$; (2)
  Determine j such that $\lambda_j^* < \lambda \le \lambda_{j+1}^*$, where $1 \le j \le n$; (3)
  Initialize the search interval of ϕ to be $I_\phi = [b_{j+1}, b_j]$; (4)
else //the power consumption model is constant-speed model (5)
  $j \leftarrow n$; (6)
  Initialize the search interval of ϕ to be $I_\phi = (-\infty, a\bar{r}]$; (7)
end if; (8)
while ($|I_\phi| \ge \epsilon$) **do** (9)
  $\phi \leftarrow \text{mid}(I_\phi)$; (10)
  for $i \leftarrow 1$ **to** j **do** //only $S_1, S_2, \ldots, S_j$ are involved (11)
    Obtain λ_i by using Algorithm Find_λ_i; (12)
  end do; (13)
  if ($\lambda_1 + \lambda_2 + \cdots + \lambda_j < \lambda$) **then** (14)
    Continue the search in the left half of I_ϕ; (15)
  else (16)
    Continue the search in the right half of I_ϕ; (17)
  end if (18)
end do; (19)
//final calculation (20)
$\phi \leftarrow \text{mid}(I_\phi)$; (21)
for $i \leftarrow 1$ **to** j **do** (22)
  Obtain λ_i by using algorithm Find_λ_i; (23)
end do; (24)
for $i \leftarrow j+1$ **to** n **do** (25)
  $\lambda_i \leftarrow 0$; (26)
end do; (27)
return $\lambda_1, \lambda_2, \ldots, \lambda_n$. (28)

Our method to solve the problem of workload management without server speed setting (i.e., a method to find ϕ as well as all the λ_i's) is given in Algorithm 1. For a given ϕ, once all the $\lambda_1, \lambda_2, \ldots, \lambda_n$ are available (lines 10–13), we check $\lambda_1 + \lambda_2 + \cdots + \lambda_n$, which is a decreasing function of ϕ. (Due to analytical sophistication, we do not prove this claim here. Instead, we have verified this fact by numerical calculation.) Hence, based on the relation between $\lambda_1 + \lambda_2 + \cdots + \lambda_n$ and λ, we can adjust the search interval of ϕ (lines 14–18). A binary search is completed when the search interval is sufficiently small (line 9), so that sufficiently accurate results can be obtained. All our binary searches are completed when the length of a search interval is less than $\epsilon = 10^{-12}$).

It remains to decide the initial search interval of ϕ (lines 1–8). The situation is fairly simple for the constant-speed model. For all $1 \le i \le n$, ϕ is in the range of $(-\infty, a\bar{r}]$ (line 7, also see Fig. 4), where $\partial G/\partial \lambda_i = a\bar{r}$ when $\lambda_i = 0$, and $\partial G/\partial \lambda_i$ approaches $-\infty$ when λ_i approaches $m_i s_i/\bar{r}$. All the n heterogeneous multiserver systems are involved in workload management (line 6).

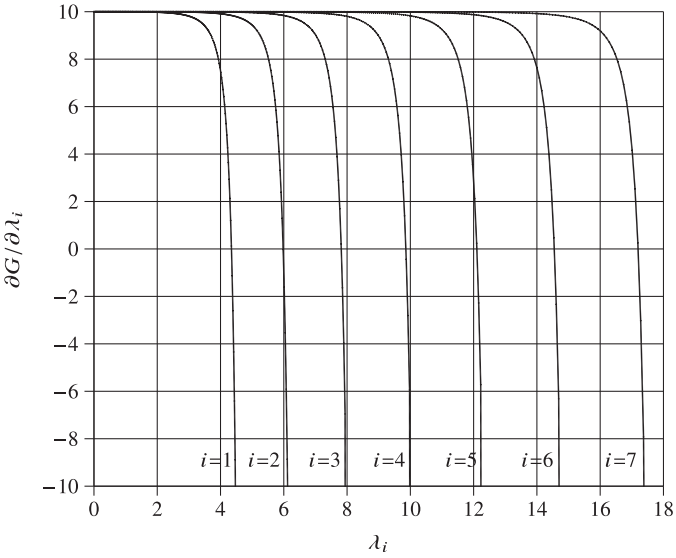

Fig. 4. $\partial G/\partial \lambda_i$ versus task arrival rate λ_i (constant-speed model).

Algorithm 2. Find_λ_i

Input: m_i, s_i, ξ_i, α_i, ϕ.
Output: λ_i such that $\partial G/\partial \lambda_i = \phi$,

Initialize the search interval of λ_i to be $I_\lambda = [0, m_i s_i/\bar{r}]$; (1)
while ($|I_\lambda| \ge \epsilon$) **do** (2)
  $\lambda_i \leftarrow \text{mid}(I_\lambda)$; (3)
  if ($\partial G/\partial \lambda_i < \phi$) **then** (4)
    Continue the search in the left half of I_λ; (5)
  else (6)
    Continue the search in the right half of I_λ; (7)
  end if (8)
end do; (9)
return $\text{mid}(I_\lambda)$. (10)

The situation for the idle-speed model is more complicated. Let $b_i = C_i - \gamma \bar{r} \xi_i s_i^{\alpha_i - 1}$, for all $1 \le i \le n$, which is actually the maximum value of $\partial G/\partial \lambda_i$ when $\lambda_i = 0$. Due to heterogeneity of the multiserver systems, each S_i may have its own b_i. For instance, the 7 heterogeneous multiserver systems have $b_1 = 9.6121560$, $b_2 = 9.5191173$, $b_3 = 9.4222849$, $b_4 = 9.3250000$, $b_5 = 9.2306242$, $b_6 = 9.1423265$, $b_7 = 9.0629077$. In general, we can assume that the indices of the n systems are arranged in such a way $b_1 > b_2 > b_3 > \cdots > b_n$ (see Fig. 3). For convenience, we also define $b_{n+1} = -\infty$. We understand that to solve our optimization problem, it is essentially to find a horizontal line in Fig. 3 (which represents ϕ), such that the line intersects with the n curves and the n intersections give the values of the λ_i's. The line should be appropriately chosen by moving up and down to represent increment/decrement of ϕ, such that n intersections result in $\lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$. However, the different b_i's imply that when λ is too small, which requires ϕ to be big enough, it is impossible to have $\partial G/\partial \lambda_i = \phi$, for all $1 \le i \le n$ and some ϕ.

To deal with the complication, for all $1 \le i \le n$, we define

$$\lambda_i^* = \sum_{j=1}^{i-1} \lambda_j',$$

TABLE 2
Numerical Data for Optimal Workload Management Without Server Speed Setting (Idle-Speed Model)

λ	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
5.0	2.6935022	2.3064978	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
10.0	3.0110558	4.0751213	2.9138228	0.0000000	0.0000000	0.0000000	0.0000000
15.0	3.1940697	4.4586719	5.6676608	1.6795976	0.0000000	0.0000000	0.0000000
20.0	3.2323399	4.5270485	5.8271255	6.4134861	0.0000000	0.0000000	0.0000000
25.0	3.3175774	4.6697662	6.1115813	7.4612515	3.4398236	0.0000000	0.0000000
30.0	3.3416159	4.7080080	6.1798369	7.6228718	8.1476675	0.0000000	0.0000000
35.0	3.4050821	4.8054681	6.3428038	7.9518918	9.4310652	3.0636890	0.0000000
40.0	3.4076601	4.8093293	6.3489799	7.9631843	9.4595398	8.0113066	0.0000000
45.0	3.4690324	4.8992641	6.4878269	8.2007322	9.9668894	11.5621986	0.4140563
50.0	3.4690485	4.8992872	6.4878614	8.2007879	9.9669928	11.5624668	5.4135554
55.0	3.4718213	4.9032662	6.4938020	8.2103479	9.9846777	11.6077112	10.3283737
60.0	3.5245382	4.9776779	6.6021837	8.3778591	10.2720006	12.2180736	14.0276670
65.0	3.7457208	5.2696487	6.9902555	8.9036633	11.0073747	13.3002796	15.7830575
70.0	4.1635581	5.7692792	7.5799074	9.5940383	11.8109374	14.2302790	16.8520005
75.0	4.6900796	6.3626601	8.2368169	10.3122046	12.5885971	15.0658360	17.7438057

where λ'_j is chosen such that $\partial G/\partial \lambda'_j = b_i$, for all $1 \le j \le i-1$. Clearly, $\lambda^*_1 = 0$. Furthermore, let

$$\lambda^*_{n+1} = \sum_{i=1}^{n} \frac{m_i s_i}{\bar{r}},$$

which is the maximum workload that the n multiserver systems can handle collectively. For the 7 heterogeneous multiserver systems, we have $\lambda^*_1 = 0$, $\lambda^*_2 = 2.6730206$, $\lambda^*_3 = 7.0736576$, $\lambda^*_4 = 13.3203419$, $\lambda^*_5 = 21.5594176$, $\lambda^*_6 = 31.9362715$, $\lambda^*_7 = 44.5859437$, $\lambda^*_8 = 76.3000000$. One important observation is that S_i can be involved in workload management only when $\lambda > \lambda^*_i$, which makes $\partial G/\partial \lambda'_i = \phi$ possible, where $\phi < b_i$. Therefore, when $\lambda^*_j < \lambda \le \lambda^*_{j+1}$, where $1 \le j \le n$, only $S_1, S_2, \ldots, S_j$ are involved in workload management (lines 11–13, 22–24), and $\lambda_{j+1} = \cdots = \lambda_n = 0$ (lines 25–27). When $\lambda^*_j < \lambda \le \lambda^*_{j+1}$, where $1 \le j \le n$, the initial search interval of ϕ is $[b_{j+1}, b_j]$ (line 4).

3.3 Numerical Data

In Tables 2 and 3, we display the optimal workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ obtained by our algorithm for the two power consumption models. It can be observed from Table 2 (and the discussion in Section 3.2) that for the idle-speed model, when λ is low, only small multiservers are involved in workload management, and large multiservers do not participate due to insufficient workload and revenue. As λ increases, more and more multiservers join in workload management. For both power consumption models, if $S_1, S_2, \ldots, S_j$ are involved in workload management, we will have $\lambda_1 < \lambda_2 < \cdots < \lambda_j$, except when S_j just starts to join in and λ is still not sufficiently large. This is intuitively reasonable and acceptable, since the S_i's have increasing sizes, i.e., $m_1 < m_2 < \cdots < m_n$.

In Figs. 5 and 6, we show the maximized profit G for the two power consumption models. It is observed that G looks similar to the G_i's in Fig. 1. As λ increases, G increases almost linearly. However, as λ approaches its maximum value, i.e., λ^*_{n+1}, G drops very quickly.

4 Workload Management With Server Speed Setting

In this section, we address the problem of workload management with server speed setting.

TABLE 3
Numerical Data for Optimal Workload Management Without Server Speed Setting (Constant-Speed Model)

λ	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7
5.0	0.0348715	0.1160103	0.2709638	0.5156470	0.8612196	1.3157801	1.8855077
10.0	0.1336940	0.3436525	0.6786734	1.1485882	1.7594818	2.5156175	3.4202926
15.0	0.2809787	0.6298071	1.1382753	1.8106511	2.6494901	3.6567332	4.8340645
20.0	0.4663449	0.9553130	1.6287556	2.4869625	3.5301726	4.7588412	6.1736102
25.0	0.6838608	1.3113762	2.1417865	3.1726024	4.4024510	5.8307095	7.4572136
30.0	0.9309428	1.6943478	2.6741696	3.8659229	5.2670054	6.8758989	8.6917127
35.0	1.2076023	2.1035869	3.2251922	4.5664695	6.1238285	7.8949723	9.8783484
40.0	1.5157650	2.5401236	3.7951580	5.2739298	6.9721242	8.8868505	11.0160489
45.0	1.8584714	3.0055238	4.3843268	5.9875223	7.8104591	9.8499623	12.1037342
50.0	2.2388711	3.5008631	4.9922024	6.7058082	8.6371858	10.7832604	13.1418091
55.0	2.6590734	4.0259488	5.6172698	7.4269211	9.4510288	11.6869545	14.1328037
60.0	3.1191050	4.5789772	6.2571855	8.1490439	10.2515902	12.5628150	15.0812832
65.0	3.6162890	5.1566480	6.9092108	8.8708501	11.0395845	13.4140700	15.9933476
70.0	4.1448226	5.7543081	7.5705077	9.5917371	11.8169178	14.2453130	16.8763937
75.0	4.6898507	6.3624797	8.2367065	10.3121811	12.5886723	15.0660167	17.7440930

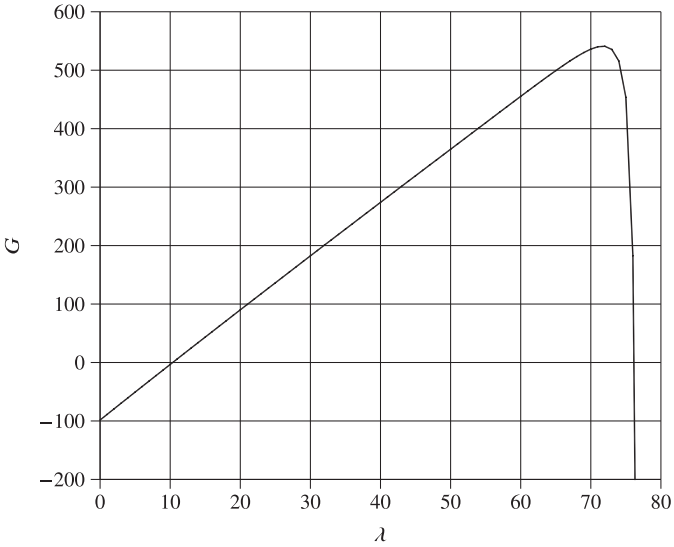

Fig. 5. Profit G versus task arrival rate λ (idle-speed model).

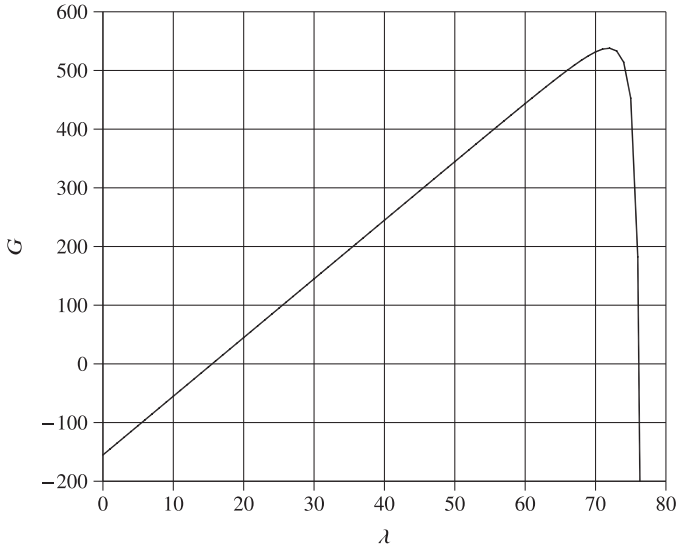

Fig. 6. Profit G versus task arrival rate λ (constant-speed model).

4.1 Analysis

To solve this multi-variable optimization problem, we view $G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)$ as a function of $\lambda_1, \lambda_2, \ldots, \lambda_n$ and $s_1, s_2, \ldots, s_n$. We can maximize $G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)$ subject to the constraint $F(\lambda_1, \lambda_2, \ldots, \lambda_n) = \lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$ by using the following Lagrange multiplier system:

$$\nabla G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n) = \phi \nabla F(\lambda_1, \lambda_2, \ldots, \lambda_n),$$

that is

$$\frac{\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)}{\partial \lambda_i} = \phi \frac{\partial F(\lambda_1, \lambda_2, \ldots, \lambda_n)}{\partial \lambda_i} = \phi,$$

for all $1 \le i \le n$, where ϕ is a Lagrange multiplier, and

$$\frac{\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)}{\partial s_i} = 0,$$

for all $1 \le i \le n$. It is easily notice that $\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)/\partial \lambda_i$ is essentially the same as $\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n)/\partial \lambda_i$ obtained in the last section. It is clear that for all $1 \le i \le n$, we have

$$\frac{\partial G}{\partial s_i} = \lambda_i \frac{\partial C_i}{\partial s_i} - \gamma \lambda_i \bar{r} \xi_i (\alpha_i - 1) s_i^{\alpha_i - 2},$$

for the idle-speed model, and

$$\frac{\partial G}{\partial s_i} = \lambda_i \frac{\partial C_i}{\partial s_i} - \gamma m_i \xi_i \alpha_i s_i^{\alpha_i - 1},$$

for the constant-speed model. Furthermore, we have

$$\begin{aligned}\frac{\partial C_i}{\partial s_i} &= -a\bar{r}\left(\frac{1}{D_1 D_2} \cdot \frac{\partial P_{q,i}}{\partial s_i} - \frac{P_{q,i}}{D_1^2 D_2} \cdot \frac{\partial D_1}{\partial s_i} - \frac{P_{q,i}}{D_1 D_2^2} \cdot \frac{\partial D_2}{\partial s_i}\right) \\ &= -a\bar{r}\left(\frac{1}{D_1 D_2} \cdot \frac{\partial P_{q,i}}{\partial s_i} - (m_i c/s_0 - \lambda_i \bar{r}/s_i^2) \frac{P_{q,i}}{D_1^2 D_2}\right. \\ &\quad \left. - (m_i(a/d + c/s_0) - \lambda_i \bar{r}/s_i^2) \frac{P_{q,i}}{D_1 D_2^2}\right),\end{aligned}$$

for all $1 \le i \le n$. To further calculate $\partial C_i/\partial s_i$, we need to examine $\partial P_{q,i}/\partial s_i$, which is

$$\begin{aligned}\frac{\partial P_{q,i}}{\partial s_i} =& \frac{m_i^{m_i}}{m_i!}\left(\frac{\rho_i^{m_i}}{1-\rho_i} \cdot \frac{\partial p_{i,0}}{\partial s_i}\right. \\ & \left. - p_{i,0} \cdot \frac{\rho_i}{s_i} \cdot \frac{m_i \rho_i^{m_i - 1}(1-\rho_i) + \rho_i^{m_i}}{(1-\rho_i)^2}\right),\end{aligned}$$

where we notice that $\partial \rho_i/\partial s_i = -\lambda_i \bar{r}/m_i s_i^2 = -\rho_i/s_i$, for all $1 \le i \le n$. To further calculate $\partial P_{q,i}/\partial s_i$, we need to examine $\partial p_{i,0}/\partial s_i$, which is

$$\begin{aligned}\frac{\partial p_{i,0}}{\partial s_i} =& p_{i,0}^2\left(\sum_{k=1}^{m_i - 1} \frac{m_i^k}{(k-1)!} \rho_i^{k-1}\right. \\ & \left. + \frac{m_i^{m_i}}{m_i!} \cdot \frac{m_i \rho_i^{m_i - 1}(1-\rho_i) + \rho_i^{m_i}}{(1-\rho_i)^2}\right) \frac{\rho_i}{s_i},\end{aligned}$$

for all $1 \le i \le n$.

4.2 Algorithms

An effective and efficient method is required to find $\lambda_1, \lambda_2, \ldots, \lambda_n$, $s_1, s_2, \ldots, s_n$, and ϕ which satisfy the equations

$$\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)/\partial \lambda_i = \phi,$$

and

$$\partial G(\lambda_1, \lambda_2, \ldots, \lambda_n, s_1, s_2, \ldots, s_n)/\partial s_i = 0,$$

for all $1 \le i \le n$, and $F(\lambda_1, \lambda_2, \ldots, \lambda_n) = \lambda$. It is clear that solving the system of $(2n+1)$ nonlinear equations directly is very complicated.

In Figs. 7 and 8, using S_4 as an example, we show the profit G_i versus server speed s_i, for $\lambda_i = 8, 0, 9.0, \ldots, 14.0$, and the idle-speed model and the constant-speed model respectively. It is observed that as s_i increases from $\lambda_i \bar{r}/m_i$, G_i increases rapidly. This is because slight increment of s_i significantly increases the processing power of S_i and reduces the task response time, which results in increased revenue and profit. However, as s_i further increases, the increased speed and power do not bring more revenue, but increased cost of energy consumption, which results in reduced profit. Therefore, there is an optimal choice of s_i.

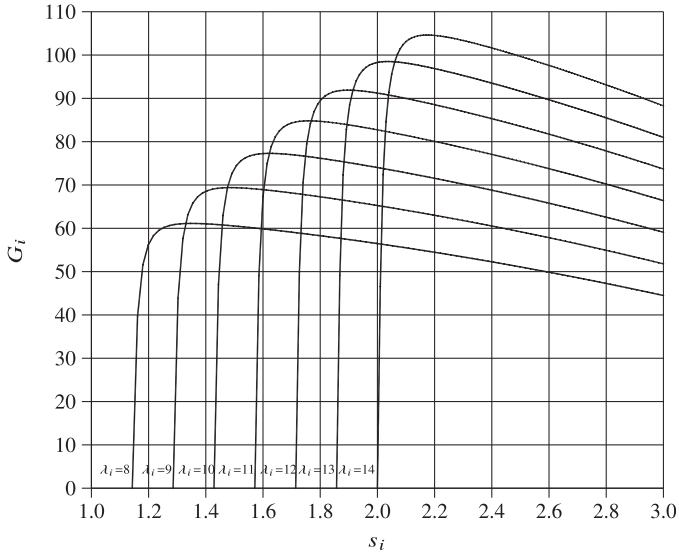

Fig. 7. Profit G_i versus server speed s_i (idle-speed model).

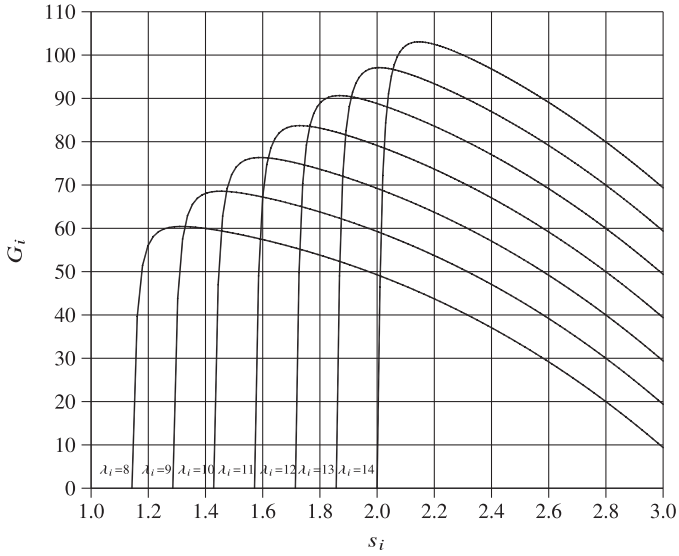

Fig. 8. Profit G_i versus server speed s_i (constant-speed model).

In Figs. 9 and 10, using S_4 as an example, we show the value of $\partial G/\partial s_i$ versus server speed s_i, for $\lambda_i = 8, 0, 9.0, \ldots, 14.0$, and the idle-speed model and the constant-speed model respectively. It is observed that $\partial G/\partial s_i$ is a decreasing function of s_i. Hence, the optimal value of s_i can be found by using the binary search method. A complete description of the method to find s_i is given in Algorithm 4.

Algorithm 3. Optimal Workload Management With Server Speed Setting

Input: Workload specified by λ and $\bar{r}$, n heterogeneous multi-server systems $S_1, S_2, \ldots, S_n$, where S_i is specified by m_i, ξ_i, α_i, P_i^*, for all $1 \leq i \leq n$, a service charge function specified by s_0, a, c, d, cost parameters β and γ.
Output: A workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and a server speed setting $(s_1, s_2, \ldots, s_n)$, such that G is maximized, subject to the constraint that $\lambda_1 + \lambda_2 + \cdots + \lambda_n = \lambda$.

```
Initialize s_1, s_2, ..., s_n with reasonable values;                    (1)
do                                                                       (2)
  Find λ_1, λ_2, ..., λ_n using Algorithm 1 based on s_1, s_2,
  ..., s_n;                                                              (3)
  for i ← 1 to n do                                                      (4)
    Find s'_i such that ∂G/∂s'_i = 0 using Algorithm Find_s_i;           (5)
  end do;                                                                (6)
  error ← max{|s'_1 − s_1|, |s'_2 − s_2|, ..., |s'_n − s_n|};           (7)
  for i ← 1 to n do                                                      (8)
    s_i ← s'_i;                                                          (9)
  end do;                                                                (10)
while (error > ε);                                                       (11)
return λ_1, λ_2, ..., λ_n and s_1, s_2, ..., s_n.                       (12)
```

Our method to solve the problem of workload management with server speed setting is described in Algorithm 3. The method is essentially an iterative method, in which, an optimal workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and an optimal server speed setting $(s_1, s_2, \ldots, s_n)$ are obtained separately, but in an interleaved manner. First of all, there is a reasonable initial server speed setting $(s_1, s_2, \ldots, s_n)$ (line 1). Then, there are iterations (lines 2–11). In each repetition, we first calculate an optimal workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ by using Algorithm 1 for the current server speed setting $(s_1, s_2, \ldots, s_n)$ (line 3). Based on the current workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$, we further calculate an optimal server speed setting $(s'_1, s'_2, \ldots, s'_n)$, by using Algorithm Find_s_i (lines 4–6), which is presented in Algorithm 4. The two server speed settings $(s_1, s_2, \ldots, s_n)$ and $(s'_1, s'_2, \ldots, s'_n)$ are compared (line 7) to decide whether enough accuracy has been reached (line 11) and whether the iteration should be repeated (lines 8–10).

4.3 Numerical Data

In Tables 4 and 5, we display the optimal workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_n)$ and an optimal server speed setting $(s_1, s_2, \ldots, s_n)$ obtained by our algorithm for the two power consumption models. It can be observed that $\lambda_1 < \lambda_2 < \cdots < \lambda_n$, which is essentially due to the sizes of the S_i's. When λ is small, we have $s_1 > s_2 > \cdots > s_n$. When λ is large, we have $s_1 < s_2 < \cdots < s_n$. This is essentially due to the parameter setting of the S_i's. Intuitively and informally, when λ_i is low, a large S_i tends to choose slower s_i to reduce the cost of energy consumption, since there is no enough revenue. When λ_i is high, a large S_i tends to choose faster s_i to increase the revenue, since there is enough workload.

Algorithm 4. Find_s_i

Input: m_i, λ_i, ξ_i, α_i, ϕ.
Output: s_i such that $\partial G/\partial s_i = 0$.

```
Initialize the search interval of s_i to be I_s = [λ_i r̄/m_i, ub],
where ub is reasonably large;                                            (1)
while (|I_s| ≥ ε) do                                                     (2)
  s_i ← mid(I_s);                                                        (3)
  if (∂G/∂s_i < 0) then                                                  (4)
    Continue the search in the left half of I_s;                         (5)
  else                                                                   (6)
    Continue the search in the right half of I_s;                        (7)
  end if                                                                 (8)
end do;                                                                  (9)
return mid(I_s).                                                         (10)
```

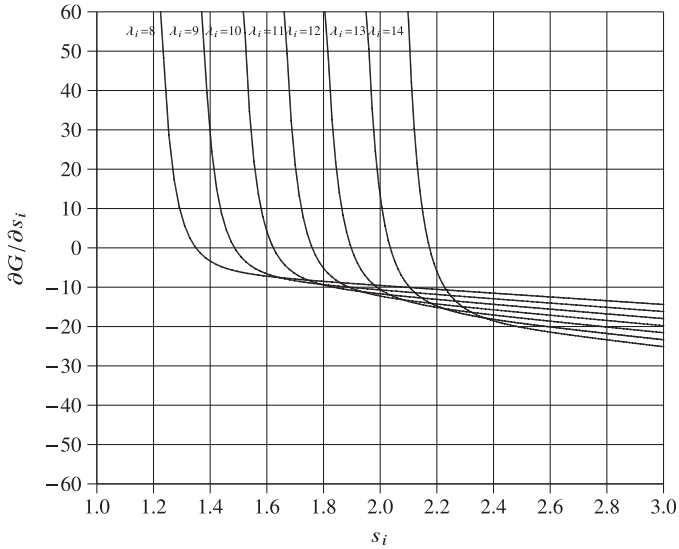

Fig. 9. $\partial G/\partial s_i$ versus server speed s_i (idle-speed model).

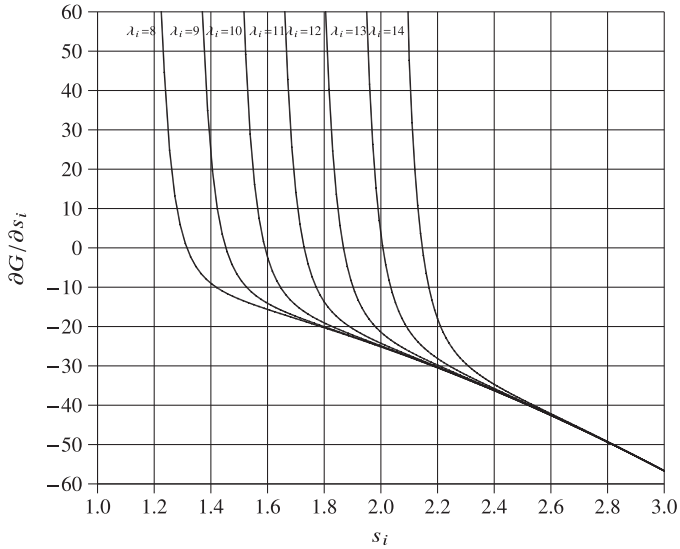

Fig. 10. $\partial G/\partial s_i$ versus server speed s_i (constant-speed model).

In Figs. 11 and 12, we show the maximized profit G for the two power consumption models. It is observed that as λ increases, G increases almost linearly in the range $0 \leq \lambda \leq 80$. There are two significant differences between Figs. 11–12 and 5–6. First, when λ is small, the s_i's can be set at low levels, thus reducing the cost of energy consumption and increasing the profit. Second, when λ is large, the s_i's can be set at high levels, thus preventing system saturation and avoiding decreased revenue and profit. Of course, when λ is sufficiently large (around 170 for the idle-speed model and 160 for the constant-speed model), the increased s_i's eventually raise the cost of energy consumption, do not increase the revenue, and reduce the profit. To summarize, the profit earned from optimal workload management with server speed setting is noticeably higher than that from optimal workload management without server speed setting.

TABLE 4
Numerical Data for Optimal Workload Management With Server Speed Setting (Idle-Speed Model)

λ	λ_1, s_1	λ_2, s_2	λ_3, s_3	λ_4, s_4	λ_5, s_5	λ_6, s_6	λ_7, s_7
10.0	0.7674925	1.0275285	1.2683179	1.4842801	1.6717334	1.8282215	1.9524261
	0.5183247	0.4950887	0.4726212	0.4506799	0.4291499	0.4080280	0.3874424
15.0	1.1650594	1.5349823	1.8834126	2.2047114	2.4947088	2.7499231	2.9672023
	0.6273584	0.6038234	0.5810681	0.5587254	0.5365487	0.5143612	0.4920336
20.0	1.5578455	2.0333584	2.4879744	2.9165365	3.3151307	3.6803330	4.0088215
	0.7253249	0.7026750	0.6809353	0.6596947	0.6386709	0.6176481	0.5964463
25.0	1.9451101	2.5240111	3.0846344	3.6226228	4.1347648	4.6183000	5.0705567
	0.8173251	0.7962568	0.7762892	0.7570029	0.7381160	0.7194165	0.7007295
30.0	2.3263370	3.0074395	3.6744713	4.3241349	4.9542927	5.5633529	6.1499717
	0.9056718	0.8866926	0.8690540	0.8523532	0.8363331	0.8208142	0.8056617
35.0	2.7013739	3.4840361	4.2581527	5.0217236	5.7740266	6.5151063	7.2455808
	0.9915613	0.9750771	0.9602211	0.9466266	0.9340821	0.9224649	0.9117119
40.0	3.0703065	3.9541882	4.8361827	5.7158145	6.5941023	7.4731756	8.3562303
	1.0756634	1.0620152	1.0503295	1.0402927	1.0317576	1.0246818	1.0191044
45.0	3.4333578	4.4182862	5.4089832	6.4067179	7.4145638	8.4371715	9.4809195
	1.1583772	1.1478593	1.1396842	1.1336062	1.1295606	1.1276080	1.1279196
50.0	3.7908203	4.8767144	5.9769246	7.0946781	8.2354077	9.4067106	10.6187441
	1.2399536	1.2328248	1.2284635	1.2267063	1.2275880	1.2312946	1.2381598
55.0	4.1430138	5.3298407	6.5403378	7.7798974	9.0566070	10.3814257	11.7688776
	1.3205588	1.3170492	1.3167752	1.3196686	1.3258804	1.3357444	1.3497865
60.0	4.4902603	5.7780106	7.0995206	8.4625488	9.8781236	11.3609721	12.9305640
	1.4003079	1.4006243	1.4046862	1.4125331	1.4244482	1.4409348	1.4627414
65.0	4.8328706	6.2215437	7.6547418	9.1427834	10.6999161	12.3450309	14.1031135
	1.4792848	1.4836145	1.4922398	1.5053197	1.5232855	1.5468314	1.5769588
70.0	5.1711374	6.6607335	8.2062439	9.8207349	11.5219430	13.3333100	15.2858973
	1.5575530	1.5660667	1.5794647	1.5980369	1.6223782	1.6533949	1.6923709
75.0	5.5053319	7.0958478	8.7542462	10.4965231	12.3441653	14.3255425	16.4783432
	1.6351626	1.6480167	1.6663810	1.6906873	1.7217087	1.7605849	1.8089120
80.0	5.8357038	7.5271304	9.2989473	11.1702556	13.1665466	15.3214861	17.6799302
	1.7121544	1.7294932	1.7530035	1.7832697	1.8212582	1.8683622	1.9265198

TABLE 5
Numerical Data for Optimal Workload Management With Server Speed Setting (Constant-Speed Model)

λ	λ_1, s_1	λ_2, s_2	λ_3, s_3	λ_4, s_4	λ_5, s_5	λ_6, s_6	λ_7, s_7
10.0	0.6780067	0.9627875	1.2316513	1.4780667	1.6985100	1.8917753	2.0592024
	0.4273204	0.4271278	0.4201445	0.4094959	0.3967579	0.3829323	0.3688379
15.0	1.0784340	1.4739675	1.8506227	2.2018754	2.5230946	2.8106152	3.0613907
	0.5393895	0.5366076	0.5279219	0.5158239	0.5014856	0.4855286	0.4683240
20.0	1.4722907	1.9721592	2.4540958	2.9123284	3.3426120	3.7413512	4.1051627
	0.6404900	0.6368432	0.6281662	0.6165528	0.6029764	0.5879264	0.5716535
25.0	1.8609901	2.4626720	3.0495005	3.6167084	4.1609392	4.6794551	5.1697347
	0.7357905	0.7323131	0.7245882	0.7144363	0.7027124	0.6898499	0.6760715
30.0	2.2438101	2.9461993	3.6383527	4.3167163	4.9791771	5.6243727	6.2513718
	0.8273529	0.8248035	0.8187223	0.8107406	0.8016468	0.7918620	0.7816288
35.0	2.6204136	3.4230370	4.2212893	5.0130365	5.7977308	6.5758399	7.3486529
	0.9162525	0.9152277	0.9113470	0.9061150	0.9002920	0.8943219	0.8885046
40.0	2.9908146	3.8935013	4.7987501	5.7060558	6.6167456	7.5335505	8.4605821
	1.0031198	1.0041068	1.0028953	1.0009102	0.9989167	0.9974115	0.9967865
45.0	3.3552222	4.3579465	5.3711100	6.3960445	7.4362533	8.4971536	9.5862700
	1.0883534	1.0917605	1.0936225	1.0953224	1.0976584	1.1012077	1.1064872
50.0	3.7139376	4.8167401	5.9387081	7.0832152	8.2562345	9.4662831	10.7248814
	1.1722204	1.1783966	1.1836869	1.1894638	1.1965831	1.2057292	1.2175760
55.0	4.0672951	5.2702427	6.5018548	7.7677463	9.0766472	10.4405793	11.8756345
	1.2549087	1.2641569	1.2731910	1.2833995	1.2957183	1.3109641	1.3299997
60.0	4.4156312	5.7187963	7.0608332	8.4497929	9.8974414	11.4197013	13.0378038
	1.3365566	1.3491431	1.3622043	1.3771675	1.3950701	1.4168849	1.4436953
65.0	4.7592688	6.1627185	7.6159010	9.1294924	10.7185656	12.4033320	14.2107216
	1.4172692	1.4334307	1.4507756	1.4707904	1.4946334	1.5234573	1.5585968
70.0	5.0985100	6.6023014	8.1672925	9.8069679	11.5399709	13.3911807	15.3937766
	1.4971296	1.5170781	1.5389406	1.5642810	1.5943973	1.6306444	1.6746393
75.0	5.4336338	7.0378114	8.7152202	10.4823303	12.3616118	14.3829817	16.5864108
	1.5762048	1.6001319	1.6267267	1.6576471	1.6943481	1.7384096	1.7917607
80.0	5.7648954	7.4694913	9.2598772	11.1556805	13.1834471	15.3784937	17.7881148
	1.6545504	1.6826306	1.7141556	1.7508929	1.7944714	1.8467176	1.9099030

5 Optimal Multiserver Selection

Notice that in our problem of workload management with server speed setting, we essentially assume that all the n multiserver systems are involved in service. In other words, there is minimum cost of $\beta_i m_i + \gamma m_i P_i^*$ for S_i, even though its workload is $\lambda_i = 0$ and its speed is $s_i = 0$. Is it possible to exclude some multiservers from consideration, i.e., only a subset of $\{S_1, S_2, \ldots, S_n\}$ are considered for optimal workload management with server speed setting, such that higher G can be obtained? Notice that "exclusion of S_i" means not only $\lambda_i = 0$, but also zero cost of S_i. Because of the cost reduction and the possibility to achieve the same revenue by using fewer multiserver systems, it is possible that G can be increased.

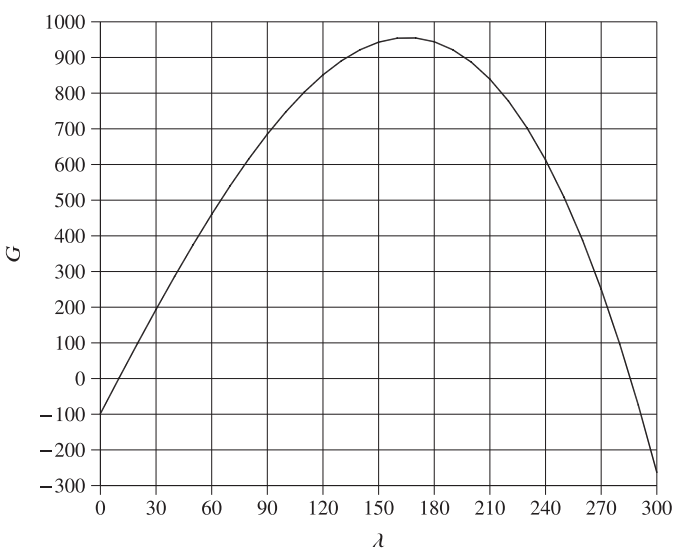

Fig. 11. Profit G versus task arrival rate λ (optimal speed setting, idle-speed model).

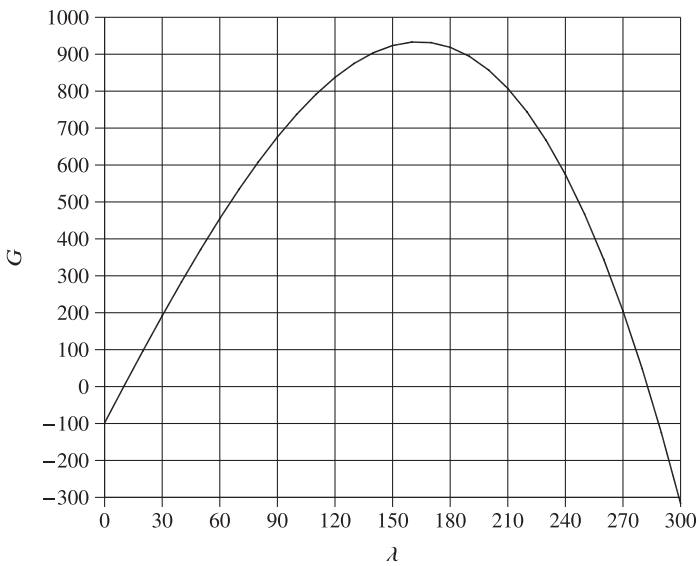

Fig. 12. Profit G versus task arrival rate λ (optimal speed setting, constant-speed model).

TABLE 6
Numerical Data for Optimal Multiserver Selection (Idle-Speed Model)

j	λ_1, s_1	λ_2, s_2	λ_3, s_3	λ_4, s_4	λ_5, s_5	λ_6, s_6	λ_7, s_7	G
4	7.6718405	9.9430994	12.3756057	15.0094545				330.9316344
	2.1444240	2.1898814	2.2459368	2.3142428				
5	5.6003279	7.2197450	8.9105732	10.6896801	12.5796737			347.2629128
	1.6572739	1.6713982	1.6912185	1.7172092	1.7501973			
6	4.3035109	5.5368119	6.7983417	8.0945575	9.4348661	10.8319118		343.9396579
	1.3573828	1.3556105	1.3573036	1.3624419	1.3712353	1.3840929		
7	3.4333578	4.4182862	5.4089832	6.4067179	7.4145638	8.4371715	9.4809195	331.1374459
	1.1583772	1.1478593	1.1396842	1.1336062	1.1295606	1.1276080	1.1279196	

TABLE 7
Numerical Data for Optimal Multiserver Selection (Constant-Speed Model)

j	λ_1, s_1	λ_2, s_2	λ_3, s_3	λ_4, s_4	λ_5, s_5	λ_6, s_6	λ_7, s_7	G
4	7.6343345	9.9257748	12.3854093	15.0544814				325.1070293
	2.0998775	2.1550478	2.2186775	2.2934363				
5	5.5484643	7.1873199	8.9036862	10.7151046	12.6454250			342.5320465
	1.6033425	1.6286851	1.6569610	1.6898644	1.7289085			
6	4.2380161	5.4898973	6.7753428	8.1011335	9.4774475	10.9181627		339.8916461
	1.2949053	1.3057525	1.3167197	1.3292115	1.3442110	1.3626081		
7	3.3552222	4.3579465	5.3711100	6.3960445	7.4362533	8.4971536	9.5862700	327.5381422
	1.0883534	1.0917605	1.0936225	1.0953224	1.0976584	1.1012077	1.1064872	

In Tables 6 and 7, for $\lambda = 45$, we consider servers $S_1, S_2, \ldots, S_j$, where $j = 4, 5, 6, 7$. For each j, we find the the optimal workload distribution $(\lambda_1, \lambda_2, \ldots, \lambda_j)$ and the optimal server speed setting $(s_1, s_2, \ldots, s_j)$, as well as the profit G. It is noticed that by using $j = 6$ multiserver systems, we can achieve higher G than using $j = 7$ multiserver systems. Furthermore, G is even higher when $j = 5$. However, G is less by using $j = 4$ multiserver systems than using $j = 5$ multiserver systems. Thus, there is an optimal number of multiserver systems. Even though the optimal number of multiserver systems is known, there is still a problem of choosing the right multiserver systems. Due to the involvement of discrete and combinatorial optimization (i.e., the selection of an optimal subset of the S_i's), such a problem of optimal multiserver selection plus their optimal workload management and server speed setting is extremely challenging, and deserves further investigation. It is very likely that the problem is NP-hard and requires heuristic algorithms to solve.

6 Concluding Remarks

We have investigated profit maximization by optimal workload management and server speed setting for multiple heterogeneous multiservers in a federated cloud or a geo-distributed data center. The heterogeneous multiservers differ in size, speed, power consumption model, workload, performance, cost, and profit. Each multiserver system is treated as an M/M/m queueing system, such that the profit of a multiserver system can be characterized analytically. We have addressed two problems, i.e., workload management without server speed setting and workload management with server speed setting. Both problems are formulated as multi-variable optimization problems. We have developed numerical algorithms to solve these problems. We have also provided numerical data for the purpose of illustration.

In addition to the optimal multiserver selection problem mentioned in Section 5, we point out two more topics for future research. One possible direction is to consider more sophisticated queueing models, e.g., M/G/m. Another possible direction is to consider competitive cloud service providers [17], each is equipped with a federated cloud.

Acknowledgments

The author would like to express his gratitude to the three anonymous reviewers for their comments and suggestions that have help to improve the presentation of the manuscript.

References

[1] Accessed: Jul. 26, 2021. [Online]. Available: http://searchtelecom.techtarget.com/definition/cloud-federation
[2] D. Ardagna *et al.*, "MODAClouds: A model-driven approach for the design and execution of applications on multiple clouds," in *Proc. 4th Int. Workshop Model. Softw. Eng.*, 2012, pp. 50–56.
[3] R. Buyya, R. Ranjan, and R. N. Calheiros, "InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services," in *Proc. 10th Int. Conf. Algorithms Archit. Parallel Process.*, 2010, pp. 13–31.
[4] J. Cao, K. Hwang, K. Li, and A. Zomaya, "Optimal multiserver configuration for profit maximization in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1087–1096, Jun. 2013.
[5] S. S. Chauhan, E. S. Pilli, R. C. Joshi, G. Singh, and M. C. Govil, "Brokering in interconnected cloud computing environments: A survey," *J. Parallel Distrib. Comput.*, vol. 133, pp. 193–209, 2019.
[6] P. Cong, G. Xu, T. Wei, and K. Li, "A survey of profit optimization techniques for cloud providers," *ACM Comput. Surv.*, vol. 53, no. 2, 2020, Art. no. 26.
[7] A. Elhabbash, F. Samreen, J. Hadley, and Y. Elkhatib, "Cloud brokerage: A systematic survey," *ACM Comput. Surv.*, vol. 51, no. 6, 2019, Art. no. 119.

[8] Y. Feng, B. Li, and B. Li, "Price competition in an oligopoly market with multiple IaaS cloud providers," *IEEE Trans. Comput.*, vol. 63, no. 1, pp. 59–73, Jan. 2014.
[9] I. Goiri, J. Guitart, and J. Torres, "Characterizing cloud federation for enhancing providers' profit," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, 2010, pp. 123–130.
[10] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*, Cambridge, U.K.: Cambridge Univ. Press, 2013.
[11] Z. He, K. Li, K. Li, W. Zhou, and J. Liu, "Server configuration optimization in mobile edge computing: A cost-performance tradeoff perspective," *Softw. Pract. Experience*, vol. 51, no. 9, pp. 1868–1895, 2021.
[12] N. R. Herbst, N. Huber, S. Kounev, E. Amrehn, "Self-adaptive workload classification and forecasting for proactive resource provisioning," *Concurrency Comput., Pract. Experience*, vol. 26, no. 12, pp. 2053–2078, 2014.
[13] J. Hu, K. Li, C. Liu, and K. Li, "A game-based price bidding algorithm for multi-attribute cloud resource provision," *IEEE Trans. Services Comput.*, vol. 14, no. 4, pp. 1111–1122, Jul./Aug. 2021.
[14] C. Jing, Y. Zhu, and M. Li, "Customer satisfaction-aware scheduling for utility maximization on geo-distributed cloud data centers," in *Proc. IEEE 10th Int. Conf. High Perform. Comput. Commun. IEEE Int. Conf. Embedded Ubiquitous Comput.*, 2013, pp. 218–225.
[15] L. Kleinrock, *Queueing Systems, vol. 1. Theory*, New York, NY, USA: Wiley, 1975.
[16] D. Kumar, G. Baranwal, Z. Raza, and D. P. Vidyarthi, "A survey on spot pricing in cloud computing," *J. Netw. Syst. Manage.*, vol. 26, no. 4, pp. 809–856, 2018.
[17] K. Li, "On the profits of competing cloud service providers: A game theoretic approach," *J. Comput. Syst. Sci.*, vol. 117, pp. 130–153, 2021.
[18] M. Lin, Z. Liu, A. Wierman, and L. L. H. Andrew, "Online algorithms for geographical load balancing," in *Proc. Int. Green Comput. Conf.*, 2012, pp. 1–10.
[19] P. Lindberg *et al.*, "Comparison and analysis of greedy energy-efficient scheduling algorithms for computational grids," in *Energy-Efficient Distributed Computing Systems*, A. Y. Zomaya and Y. C. Lee, Eds. Hoboken, NJ, USA: Wiley, 2012, pp. 189–214.
[20] C. Liu, K. Li, and K. Li, "A game approach to multi-servers load balancing with load-dependent server availability consideration," *IEEE Trans. Cloud Comput.*, vol. 9, no. 1, pp. 1–13, First Quarter 2021.
[21] S. Liu, S. Ren, G. Quan, M. Zhao, and S. Ren, "Profit aware load balancing for distributed cloud data centers," in *Proc. IEEE 27th Int. Symp. Parallel Distrib. Process.*, 2013, pp. 611–622.
[22] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, "Cloud computing – The business perspective," *Decis. Support Syst.*, vol. 51, no. 1, pp. 176–189, 2011.
[23] N. Min-Allah, S. U. Khan, and W. Yongji, "Optimal task execution times for periodic tasks using nonlinear constrained optimization," *J. Supercomputing*, vol. 59, pp. 1120–1138, 2012.
[24] N. Min-Allah, M. B. Qureshi, S. Alrashed, and O. F. Rana, "Cost efficient resource allocation for real-time tasks in embedded systems," *Sustain. Cities Soc.*, vol. 48, 2019, Art. no. 101523.
[25] K. S. Patel and A. K. Sarje, "VM provisioning policies to improve the profit of cloud infrastructure service providers," in *Proc. 3rd Int. Conf. Comput. Commun. Netw. Technol.*, 2012, pp. 1–5.
[26] H. Roh, C. Jung, W. Lee, and D.-Z. Du, "Resource pricing game in geo-distributed clouds," in *Proc. 32nd IEEE Conf. Comput. Commun.*, 2013, pp. 1519–1527.
[27] A. N. Toosi, R. N. Calheiros, R. K. Thulasiram, and R. Buyya, "Resource provisioning policies to increase IaaS provider's profit in a federated cloud environment," in *Proc. IEEE 13th Int. Conf. High Perform. Comput. Commun.*, 2011, pp. 279–287.
[28] T. Truong-Huu and C.-K. Tham, "A novel model for competition and cooperation among cloud providers," *IEEE Trans. Cloud Comput.*, vol. 2, no. 3, pp. 251–265, Third Quarter 2014.
[29] C. Wu, R. Buyya, and K. Ramamohanarao, "Cloud pricing models: Taxonomy, survey and interdisciplinary challenges," *ACM Comput. Surv.*, vol. 52, no. 6, 2019, Art. no. 108.
[30] X. Yang, B. Nasser, M. Surridge, and S. Middleton, "A business-oriented cloud federation model for real-time applications," *Future Gener. Comput. Syst.*, vol. 28, no. 8, pp. 1158–1167, 2012.

Keqin Li (Fellow, IEEE) is a SUNY distinguished professor of computer science with the State University of New York. He is also a national distinguished professor with Hunan University, China. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high-performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, intelligent and soft computing. He has authored or coauthored more than 800 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He holds more than 60 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top 10 most influential scientists in parallel and distributed computing based on a composite indicator of Scopus citation database. He has chaired many international conferences. He is currently an associate editor of the *ACM Computing Surveys* and the *CCF Transactions on High Performance Computing*. He has served on the editorial boards of the *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Services Computing*, and *IEEE Transactions on Sustainable Computing*.