IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 19, 2018, accepted December 16, 2018, date of publication December 21, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2889220

Optimal Power and Performance Management for Heterogeneous and Arbitrary Cloud Servers

KEQIN LI, (Fellow, IEEE)
College of Information Science and Engineering, Hunan University, Changsha 410082, China
Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

e-mail: lik@newpaltz.edu

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1003401 and in part by the Key Program of the National Natural Science Foundation of China under Grant 61432005.

ABSTRACT We investigate optimal power and performance management for heterogeneous and arbitrary cloud servers in a data center. In particular, we study the problems of power-constrained performance optimization and performance-constrained power optimization in a data center with multiple heterogeneous and arbitrary servers. These problems are essential to find optimal server speeds, such that: 1) the average task response time is minimized, and that the total power consumption does not exceed certain power constraint or 2) the total power consumption is minimized, and that the average task response time does not exceed certain performance constraint. Each server is treated as a G/G/1 queuing system, whose task interarrival times and task execution requirements can have arbitrary probability distributions. Furthermore, these servers are entirely heterogeneous in terms of task interarrival time, task execution requirement, and power consumption models. The main contributions of this paper are summarized as follows: 1) we formulate the average task response time as well as the total power consumption in a data center with multiple heterogeneous and arbitrary servers as the functions of server speeds; 2) we define our optimization problems by finding optimal server speeds, since the server speeds determine both the average task response time and total power consumption; 3) we develop algorithms to find the optimal solutions and demonstrate numerical data; and 4) we also develop several closed-form heuristic solutions and compare their quality with that of the optimal solution. Our approach provides an analytical way of studying the power-performance tradeoff at the data center level.

INDEX TERMS Arbitrary cloud server, average task response time, data center, heterogeneous server, power consumption.

I. INTRODUCTION

A. MOTIVATION

The Internet has created myriad new opportunities for modern society. There are about 2.5 billion people online around the world. In every minute, there are 204 million email messages exchanged, 5 million searches made on Google, 1.8 million "likes" generated on Facebook, 350,000 tweets sent on Twitter, 272,000 merchandise sold on Amazon, and 15,000 tracks downloaded via iTunes. All the above online activities are delivered through data centers, and the more we send emails, watch online videos, use social media, and conduct business online, the more demands on data centers will grow. Cloud computing is an effective way to reduce the costs associated with running traditional private data centers owned by individual companies, through large-scale, high-volume, and low-cost centralization of computing and communication resources from service providers. These cloud service providers have the necessary technical and financial capabilities, and are able to operate and maintain dynamically scalable virtual systems capable of serving a large number of consumers and customers from diversified businesses simultaneously.

The Internet of Things (IoT) has been defined in Recommendation ITU-T Y.2060 (06/2012) as a global infrastructure for the information society, enabling advanced services by interconnecting physical and virtual things based on existing and evolving interoperable information and communication technologies [5]. The IoT is the network of physical objects (e.g., goods, products, vehicles, buildings) embedded with electronics, sensors, software, and network connectivity, which enable objects to collect and process data. The IoT allows objects to be sensed and controlled remotely through

existing network infrastructure, creating opportunities for tight integration of the physical world into computer and communication systems. Each thing is uniquely identifiable through its embedded devices and is able to interoperate within the existing Internet infrastructure [3]. It is estimated that the IoT will consist of 50 billion objects by 2020 [10] and contribute 19 trillion USD in the global economy [4]. It is conceivable that cloud computing is one of the major enabling technologies for the IoT. The huge volume of data generated by the IoT require diversified services from data centers, which are well suited for large-scale transmission, analysis, and storage of data, that can be easily collected from, but not as easily processed by, IoT devices, e.g., security cameras, temperature thermostats, power monitors, etc.

The data center industry represents a significant economic burden due to its energy consumption. If the worldwide Internet were a country, it would be the 12th largest consumer of electricity in the world, somewhere between Spain and Italy. The continued expansion of the data center industry means that the energy consumption of data centers and the associated emissions of greenhouse gases and other air pollutants will continue to grow [22]. Motivated by cost reduction in owning and operating data centers, and pressure from environmental organizations, the largest consumer-facing companies like Google, Facebook, eBay, Microsoft, and Apple have been highly energy efficient. However, 11.3 (92%) of the 12.3 million servers are installed in small and medium server rooms, enterprise/corporate data centers, and multi-tenant data centers, which are much less energy efficient. A typical data center wastes large amounts of energy powering equipment doing little or no work. The average server operates at only 12–18% of capacity. Increasing energy efficiency in these data centers is a pressing issue, since they occupy 95% of electricity share. Also, since the average power usage effectiveness (PUE, i.e., the ratio of the energy used by all facilities in a data center to the energy consumed by computing equipment) is 2.9, reduction of every watt used by IT equipment results in reduction of almost 2 additional watts used by cooling, power distribution, and lighting equipment [22].

One effective way of power management is dynamic voltage scaling, i.e., a power management technique in computer architecture, where the voltage used in a component is increased or decreased, depending upon circumstances [2]. Low voltage modes are used in conjunction with lowered clock frequencies to minimize power consumption associated with components such as CPUs; only when significant computational power is needed will the voltage and frequency be raised. Dynamic voltage scaling is widely used as an effective strategy to manage switching (i.e., dynamic) power consumption. However, the speed at which a digital circuit can switch states is proportional to the voltage differential in that circuit. Reducing the voltage means that a circuit switches slower, reducing the maximum frequency at which that circuit can run. This, in turn, reduces the rate at which program instructions can be issued, which may increase run time of an application. While the quality of service is a major concern of cloud computing consumers, how to manage energy efficiency together with quality of service, i.e., a combined and balanced consideration of power and performance, becomes a significant and challenging issue in data centers.

B. RELATED WORK

Managing an energy efficient data center for cloud computing has been a hot research topic in the last few years. There have been several surveys available in the literature. Al-Dulaimy *et al.* [6] surveyed previous studies and researches that aimed to improve power efficiency of virtualized data centers. Beloglazov *et al.* [7] discussed causes and problems of high power/energy consumption, and presented a taxonomy of energy efficient design of computing systems, covering the hardware, operating system, virtualization, and data center levels. Garg and Buyya [11] discussed various elements of clouds which contribute to the total energy consumption and how it is addressed in the literature. Kong and Liu [15] investigated the green-energy-aware power management problem for data centers and surveyed and classified works that explicitly consider renewable energy and/or carbon emission. Mittal [20] highlighted the need of achieving energy efficiency in data centers and surveyed several recent architectural techniques designed for power management of data centers. Many authors examined various ways of making computing and information systems greener and environmentally sustainable, and presented a comprehensive coverage of key topics of importance and practical relevance, i.e., green technologies, design, standards, maturity models, strategies and adoption [21]. Orgerie *et al.* [23] surveyed techniques and solutions that aim to improve the energy efficiency of computing and network resources. Rahman *et al.* [24] summarized the motivations, current state of the art, approaches, and techniques proposed for power management methodologies based on geographic load balancing.

Numerous researchers have investigated power and performance management in cloud servers. Cao *et al.* [8] addressed optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers as optimization problems, i.e., power constrained performance optimization and performance constrained power optimization. Huang *et al.* [12] minimized power consumption under performance constraints through load distribution for heterogeneous embedded nodes with dedicated/general tasks and different queueing disciplines. Lefévre and Orgerie [16] explored the energy issue by analyzing how much energy virtualized environments cost, and provided an energy-efficient framework dedicated to cloud architectures. Li [17] considered the problem of optimal power allocation among multiple heterogeneous servers in a data center, i.e., minimizing the average task response time of multiple heterogeneous computer systems with energy constraint. Li [18] investigated the technique of using workload dependent dynamic power management (i.e., variable power and speed of processor cores according to the current

workload) to improve system performance and to reduce energy consumption. Malik *et al.* [19] emphasized that the operational cost of data centers is dominated by the cost on energy consumption, and modeled a data center as a cyber physical system to capture its thermal properties. Tian *et al.* [26] optimized the performance and power consumption tradeoff for multiple heterogeneous servers with continuous and discrete speed scaling. Westphall *et al.* [27] proposed two hybrid strategies to optimize the use of green cloud computing resources.

Although the above studies all considered power and performance management for cloud servers from different perspectives with different models, none has considered optimal power and performance management for heterogeneous and arbitrary cloud servers in a data center, which is the main focus of this paper.

C. NEW CONTRIBUTIONS

In this paper, we investigate optimal power and performance management for heterogeneous and arbitrary cloud servers in a data center. In particular, we study the problems of power constrained performance optimization and performance constrained power optimization in a data center with multiple heterogeneous and arbitrary servers. Essentially, the purpose of these problems is to find optimal server speeds, such that (1) the average task response time is minimized, and that the total power consumption does not exceed certain power constraint; (2) or, the total power consumption is minimized, and that the average task response time does not exceed certain performance constraint. Notice that from a user's point of view, the average task response time of all servers is an important performance measure in a data center, and from a service provider's point of view, the total power consumption of all servers is an important cost measure in a data center. Our approach to optimal power and performance management is different from other approaches, e.g., controlling the arrival rate of tasks.

It is worth to mention that in our model, each server is treated as a G/G/1 queuing system, whose task interarrival times and task execution requirements can have arbitrary probability distributions. Furthermore, these servers are entirely heterogeneous in terms of task interarrival time, task execution requirement, and power consumption model. Hence, we deal with any number of heterogeneous and arbitrary cloud servers in a data center.

The main contributions of the paper are summarized as follows.

- We formulate the average task response time as well as the total power consumption in a data center with multiple heterogeneous and arbitrary servers as functions of server speeds.
- We define our optimization problems by finding optimal server speeds, since the server speeds determine both average task response time and total power consumption.
- We develop algorithms to find the optimal solutions and demonstrate numerical data.
- We also develop several closed-form heuristic solutions and compare their quality with that of the optimal solution.

Our approach provides an analytical way of studying the power-performance tradeoff at the data center level. To the best of the author's knowledge, such combined analytical study of data center power and performance optimization has not been conducted before for heterogeneous and arbitrary cloud servers.

The rest of the paper is organized as follows. In Sections 2 and 3, we present our server model and power consumption models. In Section 4, we consider the problem of power constrained performance optimization. In Section 5, we develop heuristic methods. In Section 6, we consider the problem of performance constrained power optimization. In Section 7, we demonstrate numerical data. In Section 8, we conclude the paper.

II. THE SERVER MODEL

In this section, we present a G/G/1 queuing model for arbitrary cloud servers in a data center. Throughout the paper, we use $\bar{y}$ to denote the expectation of a random variable y, and σ_y^2 to denote the variance of y, and $C_y = \sigma_y/\bar{y}$ to denote the coefficient of variation of y.

We consider a group of n heterogeneous servers $1, 2, \ldots, n$ in a data center or a cloud computing environment, each having its own arrival stream of tasks, power supply, and execution speed. There is no load distribution and balancing mechanism. A task submitted to a server must be processed on that server, i.e., task mitigation, migration, or rejection is not allowed. System performance optimization is achieved by an optimal power allocation among the servers, i.e., an optimal speed setting of the servers. Furthermore, such performance optimization is accomplished with a power consumption constraint. We would like to emphasize that the capability for the servers to dynamically adjust their speeds is critical in our study.

Each server is modeled as a general G/G/1 queuing system. Assume that there is an arbitrary stream of arrival tasks to server i, where $1 \leq i \leq n$. The interarrival time t_i is any random variable with mean $\bar{t}_i$ and variance $\sigma_{t_i}^2$, which can be collected from observing and recording the task stream in a real server. Notice that t_i can have an arbitrary probability distribution function (pdf). The arrival rate is $\lambda_i = 1/\bar{t}_i$ (measured by the number of tasks per second). Let r_i represent the random execution requirement (measured by the number of giga instructions) of a task submitted to server i. Again, r_i can have an arbitrary probability distribution with mean $\bar{r}_i$ and variance $\sigma_{r_i}^2$, which can be obtained from real tasks. We use s_i to denote the execution speed of server i (measured in the number of giga instructions executed per second). The random execution time of a task on server i is $x_i = r_i/s_i$

(measured in second) with mean $\bar{x}_i = \bar{r}_i/s_i$ and variance $\sigma_{x_i}^2 = \sigma_{r_i}^2/s_i^2$ and coefficient of variation $C_{x_i} = \sigma_{x_i}/\bar{x}_i$.

Let $\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_n$ be the total arrival rate. The average task response time in the data center with n servers is

$$T(s_1, s_2, \ldots, s_n) = \frac{1}{\lambda}\sum_{i=1}^{n}\lambda_i\left(\frac{\bar{r}_i}{s_i} + (\bar{r}_i^2 + \sigma_{r_i}^2)\left(\frac{\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2}{2s_i(\bar{t}_i s_i - \bar{r}_i)(\bar{t}_i^2 s_i^2 + \sigma_{r_i}^2)}\right)\right),$$

where we view T as a function of server speeds $s_1, s_2, \ldots, s_n$. For clarity of presentation, the derivation of the above result is given in Appendix A.

III. POWER CONSUMPTION MODELS

In this section, we describe two types of server speed and power consumption models.

Power dissipation and circuit delay in digital CMOS circuits can be accurately modeled by simple equations, even for complex microprocessor circuits. CMOS circuits have dynamic, static, and short-circuit power dissipation; however, the dominant component in a well-designed circuit is dynamic power consumption P (i.e., the switching component of power), which is approximately $P = aCV^2f$ (measured in Watt), where a is an activity factor, C is the loading capacitance, V is the supply voltage, and f is the clock frequency [9]. In the ideal case, the supply voltage and the clock frequency are related in such a way that $V \propto f^\phi$ for some constant $\phi > 0$ [28]. The processor execution speed s is usually linearly proportional to the clock frequency, namely, $s \propto f$. For ease of discussion, we will assume that $V = bf^\phi$ and $s = cf$, where b and c are some constants. Hence, we know that power consumption is $P = aCV^2f = ab^2Cf^{2\phi+1} = (ab^2C/c^{2\phi+1})s^{2\phi+1} = \xi s^\alpha$, where $\xi = ab^2C/c^{2\phi+1}$ and $\alpha = 2\phi+1$. For instance, by setting $\alpha = 2.0$ and $\xi = 9.4192$, the value of P calculated by the equation $P = \xi s^\alpha$ is reasonably close to (with relative error less than 6.5%) that in [13] for the Intel Pentium M processor (see [13, Fig. 1.1 and Table 1.6]).

Since the servers considered in this paper are heterogeneous in the sense that each has its own ξ and α values, we assume that a server i with speed s_i consumes power $\xi_i s_i^{\alpha_i}$. Notice that a server still consumes some amount of power even when it is idle. We assume that an idle server i consumes certain base power P_i^*, which includes static power dissipation, short-circuit power dissipation, and other leakage and wasted power [1]. We will consider two types of server speed and power consumption models.

- In the *idle-speed model*, a server runs at zero speed when there is no task to perform. Since the power for speed s_i is $\xi_i s_i^{\alpha_i}$, the power supplied to server i is $P_i = \rho_i \xi_i s_i^{\alpha_i} = \lambda_i \bar{r}_i \xi_i s_i^{\alpha_i - 1}$. By including P_i^* in P_i, we get $P_i = \rho_i \xi_i s_i^{\alpha_i} + P_i^* = \lambda_i \bar{r}_i \xi_i s_i^{\alpha_i - 1} + P_i^*$.
- In the *constant-speed model*, server i still runs at the speed s_i and consumes power $\xi_i s_i^{\alpha_i}$ even if there is no task to perform (i.e., the server is not fully utilized). Hence, the power allocated to server i is $P_i = \xi_i s_i^{\alpha_i} + P_i^*$, which is independent of ρ_i.

The total power consumption (viewed as a function of server speeds $s_1, s_2, \ldots, s_n$) is

$$P(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n}(\lambda_i \bar{r}_i \xi_i s_i^{\alpha_i - 1} + P_i^*),$$

for the idle-speed model, and

$$P(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n}(\xi_i s_i^{\alpha_i} + P_i^*),$$

for the constant-speed model.

IV. POWER CONSTRAINED PERFORMANCE OPTIMIZATION

In this section, we consider power constrained performance optimization.

A. PROBLEM DEFINITION

Our optimization problem is defined as follows. Given the means $\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_n$ and the variances $\sigma_{t_1}^2, \sigma_{t_2}^2, \ldots, \sigma_{t_n}^2$ of task interarrival times, the means $\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_n$ and the variances $\sigma_{r_1}^2, \sigma_{r_2}^2, \ldots, \sigma_{r_n}^2$ of task execution requirements, parameters of the power consumption models, i.e., $\xi_1, \xi_2, \ldots, \xi_n$, and $\alpha_1, \alpha_2, \ldots, \alpha_n$, base power consumptions $P_1^*, P_2^*, \ldots, P_n^*$, and total available power $\tilde{P}$, our optimization problem is to find optimal server speeds $s_1, s_2, \ldots, s_n$, such that (1) the average task response time $T(s_1, s_2, \ldots, s_n)$ is minimized, and (2) the total power consumption $P(s_1, s_2, \ldots, s_n)$ does not exceed $\tilde{P}$.

It should be notice that the objective of the above optimization problem is to minimize the average task response time of all the servers in a data center. These servers are entirely heterogeneous in terms of mean and variance of task interarrival time, task arrival rate, mean and variance of task execution requirement, power consumption model, base power consumption, server speed, server utilization, task execution time, average task waiting time, and average task response time.

Notice that since $s_i > \lambda_i \bar{r}_i$, we need

$$\tilde{P} > \sum_{i=1}^{n}(\xi_i(\lambda_i \bar{r}_i)^{\alpha_i} + P_i^*),$$

for both idle-speed model and constant-speed model.

To meet the requirement of minimum server speeds, we must have

$$\tilde{P} > \sum_{i=1}^{n}\left(\xi_i\left(\frac{\bar{r}_i}{\bar{t}_i}\right)^{\alpha_i} + P_i^*\right),$$

for both idle-speed model and constant-speed model.

B. THE ALGORITHM

We can minimize $T(s_1, s_2, \ldots, s_n)$ subject to the constraint $P(s_1, s_2, \ldots, s_n) = \tilde{P}$ by using the following Lagrange multiplier system,

$$\nabla T(s_1, s_2, \ldots, s_n) = \phi \nabla P(s_1, s_2, \ldots, s_n),$$

where ϕ is a Lagrange multiplier (see [25, Sec. 12.8]). Notice that

$$\begin{aligned}
&\frac{\partial T(s_1, s_2, \ldots, s_n)}{\partial s_i} \\
&= \frac{\lambda_i}{\lambda} \cdot \frac{\partial T_i}{\partial s_i} \\
&= \frac{\lambda_i}{\lambda} \cdot \frac{\partial}{\partial s_i}\left(\frac{\bar{r}_i}{s_i} + \left(\frac{\bar{r}_i^2 + \sigma_{r_i}^2}{2}\right)\right. \\
&\qquad \left.\times\left(\frac{\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2}{s_i(\bar{t}_i s_i - \bar{r}_i)(\bar{t}_i^2 s_i^2 + \sigma_{r_i}^2)}\right)\right) \\
&= \frac{\lambda_i}{\lambda} \cdot \frac{\partial}{\partial s_i}\left(\frac{\bar{r}_i}{s_i} + \left(\frac{\bar{r}_i^2 + \sigma_{r_i}^2}{2}\right)\right. \\
&\qquad \left.\times\left(\frac{\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2}{\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i}\right)\right) \\
&= \frac{\lambda_i}{\lambda}\left(-\frac{\bar{r}_i}{s_i^2} + \left(\frac{\bar{r}_i^2 + \sigma_{r_i}^2}{2}\right)\right. \\
&\qquad \times\left(\frac{2\sigma_{t_i}^2 s_i}{\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i}\right. - \\
&\qquad \left.\left.\frac{(\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2)(4\bar{t}_i^3 s_i^3 - 3\bar{r}_i \bar{t}_i^2 s_i^2 + 2\bar{t}_i \sigma_{r_i}^2 s_i - \bar{r}_i \sigma_{r_i}^2)}{(\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i)^2}\right)\right).
\end{aligned}$$

Also, we have

$$\frac{\partial P(s_1, s_2, \ldots, s_n)}{\partial s_i} = (\alpha_i - 1)\lambda_i \bar{r}_i \xi_i s_i^{\alpha_i - 2},$$

for the idle-speed model, and

$$\frac{\partial P(s_1, s_2, \ldots, s_n)}{\partial s_i} = \alpha_i \xi_i s_i^{\alpha_i - 1},$$

for the constant-speed model. Hence, we get

$$\begin{aligned}
&\frac{1}{(\alpha_i - 1)\lambda \bar{r}_i \xi_i s_i^{\alpha_i - 2}}\left(-\frac{\bar{r}_i}{s_i^2} + \left(\frac{\bar{r}_i^2 + \sigma_{r_i}^2}{2}\right)\right. \\
&\quad \times\left(\frac{2\sigma_{t_i}^2 s_i}{\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i}\right. \\
&\quad \left.\left.- \frac{(\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2)(4\bar{t}_i^3 s_i^3 - 3\bar{r}_i \bar{t}_i^2 s_i^2 + 2\bar{t}_i \sigma_{r_i}^2 s_i - \bar{r}_i \sigma_{r_i}^2)}{(\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i)^2}\right)\right) \\
&= \phi,
\end{aligned}$$

for the idle-speed model, and

$$\begin{aligned}
&\frac{\lambda_i}{\lambda \alpha_i \xi_i s_i^{\alpha_i - 1}} \\
&\quad \times\left(-\frac{\bar{r}_i}{s_i^2} + \left(\frac{\bar{r}_i^2 + \sigma_{r_i}^2}{2}\right)\left(\frac{2\sigma_{t_i}^2 s_i}{\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i}\right.\right. \\
&\quad \left.\left.- \frac{(\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2)(4\bar{t}_i^3 s_i^3 - 3\bar{r}_i \bar{t}_i^2 s_i^2 + 2\bar{t}_i \sigma_{r_i}^2 s_i - \bar{r}_i \sigma_{r_i}^2)}{(\bar{t}_i^3 s_i^4 - \bar{r}_i \bar{t}_i^2 s_i^3 + \bar{t}_i \sigma_{r_i}^2 s_i^2 - \bar{r}_i \sigma_{r_i}^2 s_i)^2}\right)\right) \\
&= \phi,
\end{aligned}$$

for the constant-speed model.

It is unlikely that the above equations accommodate a closed-form solution. We use the following strategy to find a numerical solution $(\phi, s_1, s_2, \ldots, s_n)$.

A complete description of the algorithm to optimize T is given in Algorithm 1. A key observation is that the left-hand sides of the last two equations are increasing functions of s_i due to the convexity of T_i as a function of s_i. This leads to the following method to find a numerical solution $(\phi, s_1, s_2, \ldots, s_n)$. First, given a ϕ (line 3), which is negative (line 1), since $\partial T_i / \partial s_i < 0$, we can find s_i for all $1 \le i \le n$ (lines 4–6). Second, the obtained s_i's are used to verify the constraint $P(s_1, s_2, \ldots, s_n) = \tilde{P}$ (lines 7–12). Third, ϕ can be obtained by using the classical bisection method (lines 1–13), where we notice that P is an increasing function of $s_1, s_2, \ldots, s_n$.

Algorithm 1 Optimizing T

Input: Parameters $\bar{t}_i$, $\sigma_{t_i}^2$, $\bar{r}_i$, $\sigma_{r_i}^2$, ξ_i, α_i, P_i^*, for all $1 \le i \le n$, and $\tilde{P}$.
Output: Optimal $s_1, s_2, \ldots, s_n$, such that $T(s_1, s_2, \ldots, s_n)$ is minimized and $P(s_1, s_2, \ldots, s_n) \le \tilde{P}$.

```
Initialize the search interval of φ to be [−100, 0];                (1)
while (the length of the search interval is ≥ ε) do                 (2)
  φ ← the middle point of the search interval;                      (3)
  for (i ← 1; i ≤ n; i++) do                                        (4)
    Calculate s_i using Algorithm 2;                                (5)
  end do;                                                           (6)
  Calculate P(s_1, s_2, ..., s_n);                                  (7)
  if (P(s_1, s_2, ..., s_n) < P̃) then                               (8)
      Set the search interval to the right half;                    (9)
  else                                                              (10)
      Set the search interval to the left half;                     (11)
  end if                                                            (12)
end do                                                              (13)
```

A complete description of the method to find s_i is given in Algorithm 2. The value of s_i can also be found by using the bisection method (lines 1–11) in such a way that

$$(\partial T(s_1, s_2, \ldots, s_n)/\partial s_i)/(\partial P(s_1, s_2, \ldots, s_n)/\partial s_i) = \phi,$$

where we notice that s_i is an increasing function of ϕ.

It is well known that the bisection method is extremely fast and efficient. Let I denote the maximum length of all initial search intervals in this paper. Then, the time complexity of Algorithm 2 is $O(\log(I/\epsilon))$. (We set $\epsilon = 10^{-10}$ in this paper.) Due to the use of Algorithm 2 as a sub-algorithm, the time complexity of Algorithm 1 is $O(n(\log(I/\epsilon))^2)$.

V. HEURISTIC METHODS

In this section, we develop several heuristic methods with closed-form solutions, so that the optimal server speed setting can be compared with the server speed settings obtained by using these heuristic methods.

Algorithm 2 Finding s_i

Input: Parameters $\bar{t}_i$, $\sigma^2_{t_i}$, $\bar{r}_i$, $\sigma^2_{r_i}$, ξ_i, α_i, and λ.
Output: s_i such that $(\partial T/\partial s_i)/(\partial P/\partial s_i) = \phi$.

```
Initialize the search interval of s_i to be [0, 100];                (1)
while (the length of the search interval is ≥ ε) do                  (2)
    s_i ← the middle point of the search interval;                   (3)
    Calculate ∂T(s_1, s_2, ..., s_n)/∂s_i;                           (4)
    Calculate ∂P(s_1, s_2, ..., s_n)/∂s_i;                           (5)
    if ((∂T/∂s_i)/(∂P/∂s_i) < φ) then                                (6)
        Set the search interval to the right half;                   (7)
    else                                                             (8)
        Set the search interval to the left half;                    (9)
    end if                                                           (10)
end do                                                               (11)
```

There are a number of heuristic methods to be considered.

- The Workload Proportional Method — In the *workload proportional* (WP) method, the dynamic power allocated to a server is proportional to its workload $w_i = \lambda_i \bar{r}_i$. In the idle-speed model, we have

$$\lambda_i \bar{r}_i \xi_i s_i^{\alpha_i - 1} = w_i \xi_i s_i^{\alpha_i - 1} = \left(\frac{w_i}{w_1 + w_2 + \cdots + w_n}\right)\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right),$$

  which gives

$$s_i = \left(\frac{1}{\xi_i}\left(\frac{1}{w_1 + w_2 + \cdots + w_n}\right)\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right)\right)^{1/(\alpha_i - 1)},$$

  for all $1 \le i \le n$. In the constant-speed model, we have

$$\xi_i s_i^{\alpha_i} = \left(\frac{w_i}{w_1 + w_2 + \cdots + w_n}\right)\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right),$$

  which gives

$$s_i = \left(\frac{1}{\xi_i}\left(\frac{w_i}{w_1 + w_2 + \cdots + w_n}\right)\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right)\right)^{1/\alpha_i},$$

  for all $1 \le i \le n$.
- The Equal Speed Method — In the *equal speed* (ES) method, all servers have the same speed s. For the idle-speed model, we have

$$P(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} (\lambda_i \bar{r}_i \xi_i s_i^{\alpha_i - 1} + P_i^*) = \sum_{i=1}^{n} (\lambda_i \bar{r}_i \xi_i s^{\alpha_i - 1} + P_i^*) = \tilde{P}.$$

  Therefore, s satisfies the following equation,

$$\sum_{i=1}^{n} \lambda_i \bar{r}_i \xi_i s^{\alpha_i - 1} = \tilde{P} - \sum_{i=1}^{n} P_i^*.$$

  If $\alpha_1 = \alpha_2 = \cdots = \alpha_n = \alpha$, we get

$$s = \left(\left(\sum_{i=1}^{n} \lambda_i \bar{r}_i \xi_i\right)^{-1}\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right)\right)^{1/(\alpha - 1)}.$$

  For the constant-speed model, we have

$$P(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} (\xi_i s_i^{\alpha_i} + P_i^*) = \sum_{i=1}^{n} (\xi_i s^{\alpha_i} + P_i^*) = \tilde{P}.$$

  Therefore, s satisfies the following equation,

$$\sum_{i=1}^{n} \xi_i s^{\alpha_i} = \tilde{P} - \sum_{i=1}^{n} P_i^*.$$

  If $\alpha_1 = \alpha_2 = \cdots = \alpha_n = \alpha$, we get

$$s = \left(\left(\sum_{i=1}^{n} \xi_i\right)^{-1}\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right)\right)^{1/\alpha}.$$

- The Equal Utilization Method — In the *equal utilization* (EU) method, all servers have the same utilization ρ, i.e., $\rho_i = w_i/s_i = \rho$, and $s_i = w_i/\rho$, for all $1 \le i \le n$. For the idle-speed model, we have

$$P(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} (w_i \xi_i s_i^{\alpha_i - 1} + P_i^*) = \sum_{i=1}^{n} \left(\xi_i \frac{w_i^{\alpha_i}}{\rho^{\alpha_i - 1}} + P_i^*\right) = \tilde{P}.$$

  Therefore, ρ satisfies the following equation,

$$\sum_{i=1}^{n} \xi_i \frac{w_i^{\alpha_i}}{\rho^{\alpha_i - 1}} = \tilde{P} - \sum_{i=1}^{n} P_i^*.$$

  If $\alpha_1 = \alpha_2 = \cdots = \alpha_n = \alpha$, we get

$$\rho = \left(\left(\sum_{i=1}^{n} \xi_i w_i^{\alpha}\right)\left(\tilde{P} - \sum_{i=1}^{n} P_i^*\right)^{-1}\right)^{1/(\alpha - 1)}.$$

  For the constant-speed model, we have

$$P(s_1, s_2, \ldots, s_n) = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} (\xi_i s_i^{\alpha_i} + P_i^*) = \sum_{i=1}^{n} \left(\xi_i \left(\frac{w_i}{\rho}\right)^{\alpha_i} + P_i^*\right) = \tilde{P}.$$

  Therefore, ρ satisfies the following equation,

$$\sum_{i=1}^{n} \xi_i \frac{w_i^{\alpha_i}}{\rho^{\alpha_i}} = \tilde{P} - \sum_{i=1}^{n} P_i^*.$$

Algorithm 3 Optimizing P

Input: Parameters $\bar{t}_i, \sigma^2_{t_i}, \bar{r}_i, \sigma^2_{r_i}, \xi_i, \alpha_i, P^*_i$, for all $1 \le i \le n$, and $\tilde{T}$.
Output: Optimal $s_1, s_2, \ldots, s_n$, such that $P(s_1, s_2, \ldots, s_n)$ is minimized and $T(s_1, s_2, \ldots, s_n) \le \tilde{T}$.

```
Initialize the search interval of P;                    (1)
while (the length of the search interval is ≥ ε) do     (2)
    P ← the middle point of the search interval;        (3)
    Call Alg. 1 to find the optimal T with P̃ = P;       (4)
    if (T > T̃) then                                     (5)
        Set the search interval to the right half;      (6)
    else                                                (7)
        Set the search interval to the left half;       (8)
    end if                                              (9)
end do                                                  (10)
```

If $\alpha_1 = \alpha_2 = \cdots = \alpha_n = \alpha$, we get

$$\rho = \left(\left(\sum_{i=1}^{n} \xi_i w_i^{\alpha}\right)\left(\tilde{P} - \sum_{i=1}^{n} P^*_i\right)^{-1}\right)^{1/\alpha}.$$

- The Equal Time Method — In the *equal time* (ET) method, all servers have the same average task response time T, i.e., $T_1 = T_2 = \cdots = T_n = T$. Therefore, s_i satisfies the following equation,

$$\frac{\bar{r}_i}{s_i} + (\bar{r}_i^2 + \sigma^2_{r_i})\left(\frac{\sigma^2_{t_i} s_i^2 + \sigma^2_{r_i}}{2 s_i (\bar{t}_i s_i - \bar{r}_i)(\bar{t}_i^2 s_i^2 + \sigma^2_{r_i})}\right) = T.$$

We observe that the left-hand side of the above equation is a decreasing functions of s_i. Given a T, we can find s_i for all $1 \le i \le n$ by using the bisection method. The obtained s_i's are used to verify the constraint $P(s_1, s_2, \ldots, s_n) = \tilde{P}$. The value of T can also be found by using the bisection method in such a way that $P(s_1, s_2, \ldots, s_n) = \tilde{P}$.

VI. PERFORMANCE CONSTRAINED POWER OPTIMIZATION

In this section, we consider performance constrained power optimization, which is actually a dual form of power constrained performance optimization.

A. PROBLEM DEFINITION

Given the means $\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_n$ and the variances $\sigma^2_{t_1}, \sigma^2_{t_2}, \ldots, \sigma^2_{t_n}$ of task interarrival times, the means $\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_n$ and the variances $\sigma^2_{r_1}, \sigma^2_{r_2}, \ldots, \sigma^2_{r_n}$ of task execution requirements, parameters of the power consumption models, i.e., $\xi_1, \xi_2, \ldots, \xi_n$, and $\alpha_1, \alpha_2, \ldots, \alpha_n$, base power consumptions $P^*_1, P^*_2, \ldots, P^*_n$, and a time constraint $\tilde{T}$, our dual optimization problem is to find optimal server speeds $s_1, s_2, \ldots, s_n$, such that (1) the total power consumption $P(s_1, s_2, \ldots, s_n)$ is minimized, and (2) the average task response time $T(s_1, s_2, \ldots, s_n)$ does not exceed $\tilde{T}$.

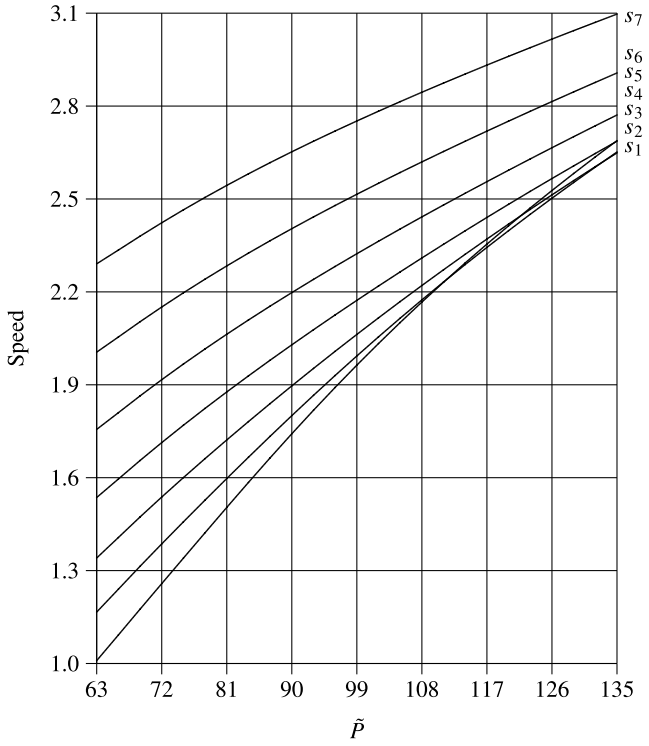

FIGURE 1. Optimal server speeds (idle-speed model).

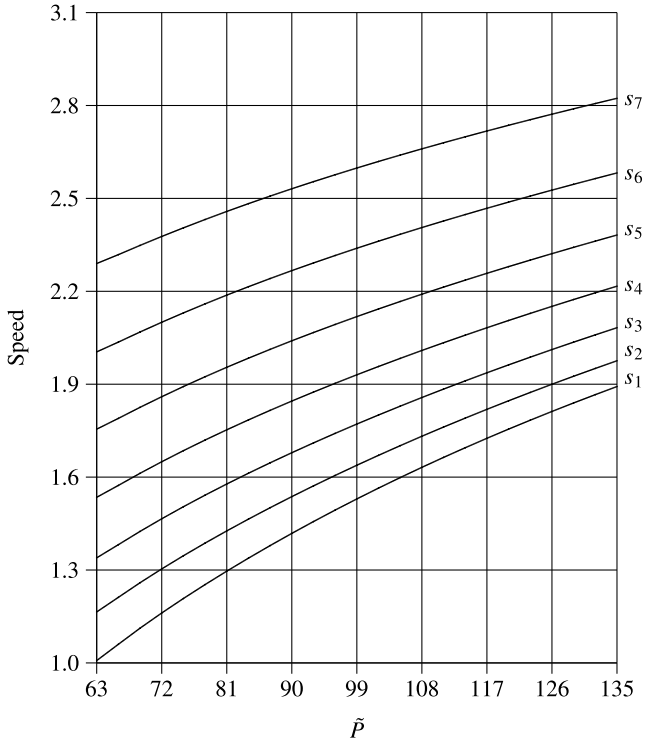

FIGURE 2. Optimal server speeds (constant-speed model).

B. THE ALGORITHM

It is clear that the above optimization problem can be solved by bisection search of $\tilde{P}$ that yields $T(s_1, s_2, \ldots, s_n) = \tilde{T}$ and

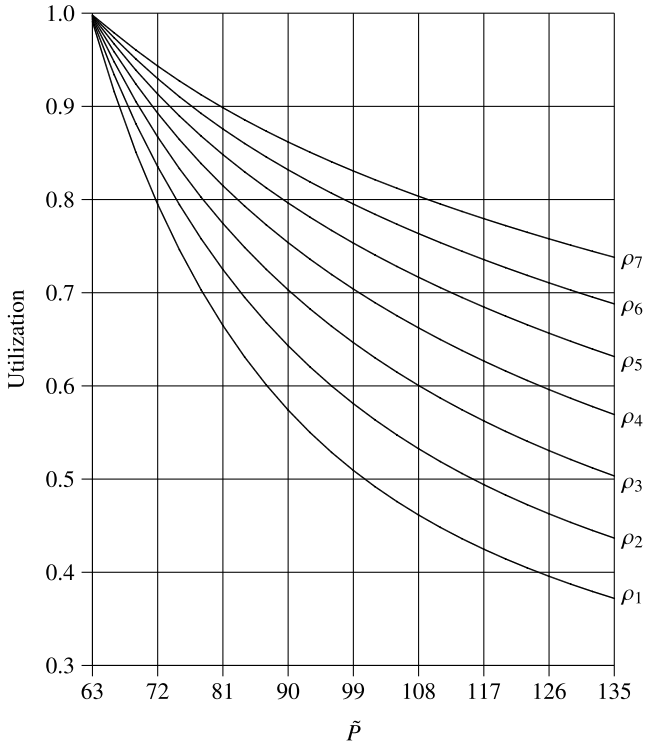

FIGURE 3. Server utilization (idle-speed model).

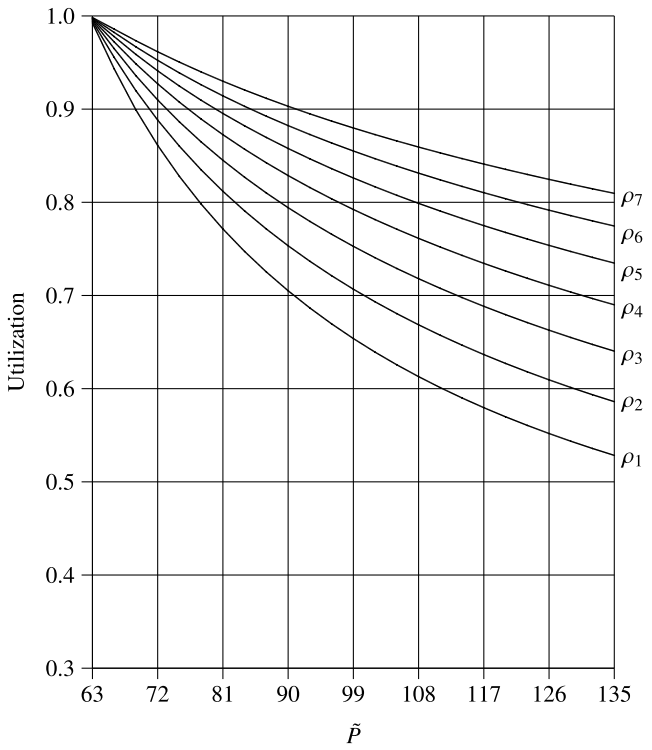

FIGURE 4. Server utilization (constant-speed model).

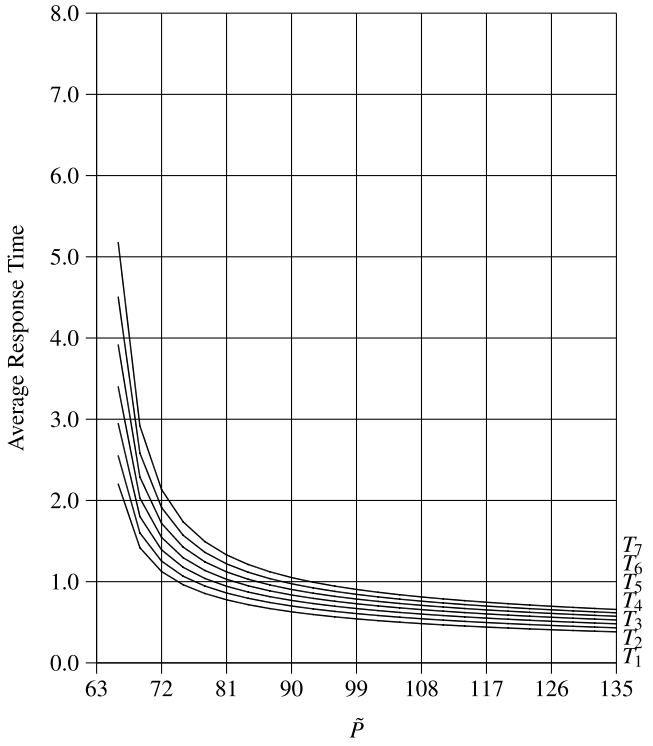

FIGURE 5. Average task response times (idle-speed model).

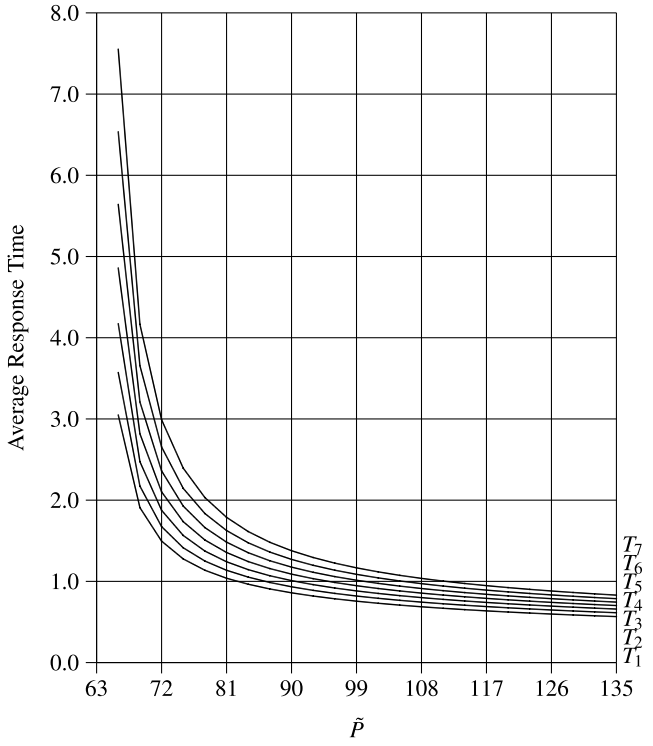

FIGURE 6. Average task response times (constant-speed model).

the solution to the dual optimization problem, based on the observation that T is a decreasing function of $\tilde{P}$. A complete description of the method is given in Algorithm 3. The initial search interval of P is

$$\left[\sum_{i=1}^{n}(\xi_i(\lambda_i\bar{r}_i)^{\alpha_i}+P_i^*),\ 1000\right].$$

TABLE 1. Performance comparison (power constrained, idle-speed model).

$\tilde{P}$	WP	ES	EU	ET	OPT
63.0	—	—	63.9278478	58.6240391	31.7770505
66.0	—	—	4.3257014	3.9726990	3.6439684
69.0	—	—	2.5096058	2.3078020	2.1502993
72.0	—	—	1.8727847	1.7240902	1.6228733
75.0	—	—	1.5448081	1.4234505	1.3489804
78.0	—	—	1.3429553	1.2383629	1.1789228
81.0	—	—	1.2050346	1.1118244	1.0617014
84.0	—	—	1.1040565	1.0191073	0.9751537
87.0	—	—	1.0264056	0.9477439	0.9080790
90.0	—	—	0.9644639	0.8907618	0.8541947
93.0	—	4.1037799	0.9136319	0.8439540	0.8096966
96.0	—	1.3718077	0.8709660	0.8046285	0.7721409
99.0	—	1.0316946	0.8344926	0.7709809	0.7398827
102.0	—	0.8886355	0.8028376	0.7417548	0.7117714
105.0	—	0.8056014	0.7750137	0.7160476	0.6869768
108.0	2.6236078	0.7493019	0.7502927	0.6931930	0.6648831
111.0	1.2735480	0.7075043	0.7281249	0.6726878	0.6450234
114.0	0.9807738	0.6745976	0.7080872	0.6541446	0.6270364
117.0	0.8470258	0.6476168	0.6898483	0.6372596	0.6106378
120.0	0.7676590	0.6248320	0.6731447	0.6217913	0.5956005
123.0	0.7136339	0.6051579	0.6577642	0.6075447	0.5817406
126.0	0.6736253	0.5878735	0.6435335	0.5943607	0.5689072
129.0	0.6422738	0.5724782	0.6303099	0.5821078	0.5569756
132.0	0.6166995	0.5586115	0.6179743	0.5706767	0.5458414
135.0	0.5952075	0.5460058	0.6064268	0.5599752	0.5354164

Due to the use of Algorithm 1 as a sub-algorithm, the time complexity of Algorithm 3 is $O(n(\log(I/\epsilon))^3)$.

C. HEURISTIC METHODS

For the EU, ET, and the optimal methods, $\tilde{P}$ can be arbitrarily close to its lower bound, i.e.,

$$\sum_{i=1}^{n}(\xi_i(\lambda_i\bar{r}_i)^{\alpha_i}+P_i^*).$$

For the WP method, we notice that in the idle-speed model,

$$s_i=\left(\frac{1}{\xi_i}\left(\frac{1}{w_1+w_2+\cdots+w_n}\right)\left(\tilde{P}-\sum_{i=1}^{n}P_i^*\right)\right)^{1/(\alpha_i-1)}>\bar{r}_i/\bar{t}_i,$$

which gives

$$\tilde{P}>\xi_i(w_1+w_2+\cdots+w_n)\left(\frac{\bar{r}_i}{\bar{t}_i}\right)^{\alpha_i-1}+\sum_{i=1}^{n}P_i^*,$$

for all $1\le i\le n$. In the constant-speed model, we have

$$s_i=\left(\frac{1}{\xi_i}\left(\frac{w_i}{w_1+w_2+\cdots+w_n}\right)\left(\tilde{P}-\sum_{i=1}^{n}P_i^*\right)\right)^{1/\alpha_i}>\bar{r}_i/\bar{t}_i,$$

which gives

$$\tilde{P}>\xi_i\left(\frac{w_1+w_2+\cdots+w_n}{w_i}\right)\left(\frac{\bar{r}_i}{\bar{t}_i}\right)^{\alpha_i}+\sum_{i=1}^{n}P_i^*,$$

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

For the ES method, we notice that in the idle-speed model, if $\alpha_1=\alpha_2=\cdots=\alpha_n=\alpha$, we have

$$s=\left(\left(\sum_{i=1}^{n}\lambda_i\bar{r}_i\xi_i\right)^{-1}\left(\tilde{P}-\sum_{i=1}^{n}P_i^*\right)\right)^{1/(\alpha-1)}>\bar{r}_i/\bar{t}_i,$$

which gives

$$\tilde{P}>\left(\sum_{i-1}^{n}\lambda_i\bar{r}_i\xi_i\right)\left(\frac{\bar{r}_i}{\bar{t}_i}\right)^{\alpha-1}+\sum_{i=1}^{n}P_i^*,$$

for all $1\le i\le n$. In the constant-speed model, if $\alpha_1=\alpha_2=\cdots=\alpha_n=\alpha$, we have

$$s=\left(\left(\sum_{i=1}^{n}\xi_i\right)^{-1}\left(\tilde{P}-\sum_{i=1}^{n}P_i^*\right)\right)^{1/\alpha}>\bar{r}_i/\bar{t}_i,$$

which gives

$$\tilde{P}>\left(\sum_{i-1}^{n}\xi_i\right)\left(\frac{\bar{r}_i}{\bar{t}_i}\right)^{\alpha}+\sum_{i=1}^{n}P_i^*,$$

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

VII. NUMERICAL DATA

In this section, we demonstrate numerical data for the performance of our optimization algorithms and heuristic algorithms using synthetic parameters. Our computing environment is an Intel® Xeon® CPU E5620 2.40GHz with the Linux OS version RHEL 6.8. All the data in this section are generated by a computation program written in C++ supported by the g++ 4.4.7 compiler.

TABLE 2. Performance comparison (power constrained, constant-speed model).

$\tilde{P}$	WP	ES	EU	ET	OPT
63.0	—	—	95.6449097	87.7073810	38.8419287
66.0	—	—	6.2507746	5.7383891	5.2330295
69.0	—	—	3.5351405	3.2486967	3.0047632
72.0	—	—	2.5878994	2.3804502	2.2247615
75.0	—	—	2.1034564	1.9364987	1.8241353
78.0	—	—	1.8077740	1.6655727	1.5784593
81.0	—	—	1.6076024	1.4821748	1.4113259
84.0	—	—	1.4624906	1.3492207	1.2895662
87.0	—	—	1.3520448	1.2480169	1.1964423
90.0	—	—	1.2648639	1.1681152	1.1225866
93.0	—	—	1.1940720	1.1032160	1.0623424
96.0	—	—	1.1352742	1.0492939	1.0120886
99.0	—	—	1.0855287	1.0036552	0.9693976
102.0	—	—	1.0427912	0.9644287	0.9325793
105.0	—	—	1.0055959	0.9302731	0.9004197
108.0	3.8205788	—	0.9728640	0.9002017	0.8720237
111.0	1.7990116	—	0.9437836	0.8734718	0.8467158
114.0	1.3632272	—	0.9177314	0.8495135	0.8239768
117.0	1.1658457	—	0.8942207	0.8278819	0.8034002
120.0	1.0499084	—	0.8728660	0.8082245	0.7846631
123.0	0.9718638	17.8197736	0.8533574	0.7902582	0.7675057
126.0	0.9147314	2.3847530	0.8354433	0.7737529	0.7517163
129.0	0.8704788	1.5657610	0.8189168	0.7585194	0.7371205
132.0	0.8347918	1.2658779	0.8036062	0.7444009	0.7235734
135.0	0.8051341	1.1069346	0.7893680	0.7312661	0.7109532

TABLE 3. Performance comparison (performance constrained, idle-speed model).

$\tilde{T}$	WP	ES	EU	ET	OPT
0.6	134.2884172	123.8572803	136.7557103	124.6890845	119.0960334
0.8	118.6171222	105.2560434	102.2893780	96.3869942	93.7293995
1.0	113.6950720	99.5110907	88.2023718	84.7342090	83.0519679
1.2	111.5143183	97.1135698	81.1305927	78.8054486	77.5558477
1.4	110.3179802	95.8621472	77.0184842	75.3153006	74.3081303
1.6	109.5698391	95.1073420	74.3694322	73.0442714	72.1948190
1.8	109.0599992	94.6063611	72.5340769	71.4581675	70.7211354
2.0	108.6910505	94.2509408	71.1927346	70.2914921	69.6394129
2.2	108.4120169	93.9862289	70.1719501	69.3989553	68.8136873
2.4	108.1937554	93.7816742	69.3702100	68.6948910	68.1637125
2.6	108.0184412	93.6189836	68.7244317	68.1257184	67.6393109
2.8	107.8745766	93.4865626	68.1934682	67.6562918	67.2075984
3.0	107.7544202	93.3767189	67.7493907	67.2626381	66.8461733
3.2	107.6525726	93.2841537	67.3726020	66.9278635	66.5392715
3.4	107.5651547	93.2051006	67.0489590	66.6397293	66.2754899
3.6	107.4893087	93.1368109	66.7680065	66.3891576	66.0463831
3.8	107.4228836	93.0772321	66.5218527	66.1692755	65.8455663
4.0	107.3642295	93.0248013	66.3044317	65.9747860	65.6681257
4.2	107.3120600	92.9783075	66.1110058	65.8015416	65.5102191
4.4	107.2653574	92.9367983	65.9378217	65.6462496	65.3688000
4.6	107.2233063	92.8995142	65.7818681	65.5062627	65.2414221
4.8	107.1852453	92.8658423	65.6407015	65.3794289	65.1260991
5.0	107.1506328	92.8352828	65.5123186	65.2639809	65.0212014
5.2	107.1190210	92.8074237	65.3950616	65.1584538	64.9253799
5.4	107.0900361	92.7819226	65.2875469	65.0616235	64.8375080

Let us consider a group of $n = 7$ heterogeneous servers with the following parameters: $\bar{t}_i = 1.05-0.05i$, $\sigma_{t_i} = 0.21-0.01i$, $\bar{r}_i = 0.9+0.1i$, $\sigma_{r_i} = 0.45+0.05i$, $\xi_i = 0.9+0.1i$, $\alpha_i = 3$, $P_i^* = 2$, for all $1 \le i \le n$. The above parameters imply that

$$\sum_{i=1}^{n}\left(\xi_i\left(\frac{\bar{r}_i}{\bar{t}_i}\right)^{\alpha_i}+P_i^*\right)=62.8126110.$$

TABLE 4. Performance comparison (performance constrained, constant-speed model).

$\tilde{T}$	WP	ES	EU	ET	OPT
0.6	182.3802468	194.9225752	208.6688826	181.7643823	173.5982278
0.8	135.5743537	150.9752017	132.7398966	121.3408969	117.5244558
1.0	121.7966005	138.2462142	105.4872686	99.2617300	96.8030697
1.2	116.3353676	133.0576756	92.7253686	88.7227517	86.8709812
1.4	113.6108945	130.3788854	85.6037734	82.7486582	81.2445081
1.6	112.0230823	128.7730042	81.1366422	78.9552291	77.6813375
1.8	110.9958269	127.7112594	78.0983930	76.3504388	75.2426645
2.0	110.2810242	126.9600027	75.9077862	74.4580871	73.4768115
2.2	109.7565927	126.4015572	74.2577946	73.0240075	72.1427027
2.4	109.3561520	125.9706539	72.9723232	71.9011284	71.1010121
2.6	109.0407352	125.6283330	71.9436530	70.9988027	70.2660228
2.8	108.7860493	125.3499613	71.1024048	70.2582660	69.5822833
3.0	108.5761993	125.1192268	70.4019733	69.6398208	69.0124152
3.2	108.4003646	124.9249107	69.8099385	69.1157163	68.5303454
3.4	108.2509322	124.7590497	69.3030749	68.6659862	68.1173534
3.6	108.1223947	124.6158379	68.8643213	68.2759069	67.7596621
3.8	108.0106714	124.4909443	68.4808703	67.9343883	67.4469136
4.0	107.9126763	124.3810738	68.1429248	67.6329211	67.1711745
4.2	107.8260327	124.2836750	67.8428654	67.3648692	66.9262679
4.4	107.7488813	124.1967425	67.5746789	67.1249823	66.7073147
4.6	107.6797470	124.1186781	67.3335573	66.9090528	66.5104103
4.8	107.6174445	124.0481928	67.1156109	66.7136700	66.3323940
5.0	107.5610108	123.9842355	66.9176596	66.5360399	66.1706798
5.2	107.5096555	123.9259407	66.7370785	66.3738524	66.0231321
5.4	107.4627238	123.8725891	66.5716819	66.2251809	65.8879721

TABLE 5. Accuracy of the G/G/1 approximation.

ρ	Simulation	99% C.I.	Approximation	Relative Error
0.1000000	1.4056398	0.1393483%	1.4435529	2.6972120%
0.1500000	1.4141122	0.1399932%	1.4695010	3.9168597%
0.2000000	1.4320511	0.1411626%	1.4991060	4.6824357%
0.2500000	1.4509859	0.1422068%	1.5332366	5.6686030%
0.3000000	1.4799714	0.1441451%	1.5730007	6.2858827%
0.3500000	1.5167226	0.1458743%	1.6198362	6.7984467%
0.4000000	1.5677397	0.1488191%	1.6756467	6.8829667%
0.4500000	1.6284936	0.1522949%	1.7430114	7.0321330%
0.5000000	1.7063720	0.1563612%	1.8255226	6.9826843%
0.5500000	1.8059174	0.1607947%	1.9283472	6.7793659%
0.6000000	1.9365131	0.1663225%	2.0592108	6.3360111%
0.6500000	2.1099013	0.1731604%	2.2302259	5.7028543%
0.7000000	2.3413061	0.1814181%	2.4615484	5.1356942%
0.7500000	2.6675611	0.1898358%	2.7894226	4.5682757%
0.8000000	3.1908007	0.2018568%	3.2862949	2.9927991%
0.8500000	4.0303375	0.2102933%	4.1211549	2.2533437%
0.9000000	5.8513945	0.2286247%	5.8009004	-0.8629412%
0.9500000	11.0209946	0.2382927%	10.8598948	-1.4617541%

(Notice that these synthetic parameters are for illustrative purpose only. As mentioned earlier, our optimization algorithms are applicable to any data centers with any number of arbitrary servers.)

For power constrained performance optimization, we give the optimal speed setting, including the optimal server speeds $s_1, s_2, \ldots, s_n$, the server utilization $\rho_1, \rho_2, \ldots, \rho_n$, and the average task response times $T_1, T_2, \ldots, T_n$, in Figures 1–6 for the two power consumption models, where $\tilde{P} = 63, 66, 69, \ldots, 135$. It is clear that the servers $1, 2, \ldots, n$ have increased arrival rate ($\lambda_1 < \lambda_2 < \cdots < \lambda_n$), increased execution requirement ($\bar{r}_1 < \bar{r}_2 < \cdots < \bar{r}_n$), and increased power consumption ($\xi_1 < \xi_2 < \cdots < \xi_n$). Thus, the servers $1, 2, \ldots, n$ have increased server speed ($s_1 < s_2 < \cdots < s_n$), increased server utilization ($\rho_1 < \rho_2 < \cdots < \rho_n$), and increased average response time ($T_1 < T_2 < \cdots < T_n$). (The only exception is that for the idle-speed model, there might be $s_{i_1} > s_{i_2}$ for $i_1 < i_2$, when $\tilde{P}$ is large.) As $\tilde{P}$ increases, all the s_i's increase, and the servers $1, 2, \ldots, n$ have reduced percentage of increment; all the ρ_i's decrease, and the servers

$1, 2, \ldots, n$ have reduced percentage of decrement; all the T_i's decrease, and the servers $1, 2, \ldots, n$ have reduced percentage of decrement.

In Tables 1–2, we compare the performance of the four heuristic methods with that of the optimal solution. It is noticed that if $\tilde{P}$ is not sufficient, it is impossible to implement the WP and ES methods (indicated by "—" in the tables). If $\tilde{P}$ is sufficiently large, all the four heuristic methods have performance comparable to that of the optimal solution. ET has the best performance among the four heuristic methods, since the optimal speed setting tends to make all servers to have roughly the same average task response time.

For performance constrained power optimization, we compare the performance of the four heuristic methods with that of the optimal solution in Tables 3–4, where $\tilde{T} = 0.6, 0.8, 1.0, \ldots, 5.4$. It is noticed that it is always possible to implement the four heuristic methods. As $\tilde{T}$ increases, all methods have reduced power consumption, and EU and ET have more significant reduction than WP and ES, since the optimal speed setting tends to make all servers to have roughly the same utilization and roughly the same average task response time. Again, ET has the best performance among the four heuristic methods.

VIII. CONCLUDING REMARKS

We have investigated optimal power and performance management in a data center with multiple heterogeneous and arbitrary cloud servers. The tradeoff between power and performance is tackled by studying the problems of power constrained performance optimization and performance constrained power optimization. These problems have significant practical importance and implication in data centers supporting cloud computing. Our problems are formulated as multi-variable optimizations by modeling each server as a G/G/1 queuing system, the most general class of queuing models. We are able to find optimal server speed settings numerically. We also find that some simple heuristic solutions such as EU and ET generate near-optimal solutions.

APPENDIX A
DERIVATION OF THE AVERAGE TASK RESPONSE TIME

The average waiting time of tasks in server i is approximately ([14, p. 34, and Appendix B])

$$W_i = \frac{1 + C_{x_i}^2}{(1/\rho_i)^2 + C_{x_i}^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{x_i}^2}{2\bar{t}_i(1-\rho_i)},$$

where

$$\rho_i = \lambda_i \bar{x}_i = \frac{\bar{x}_i}{\bar{t}_i} = \frac{\bar{r}_i}{\bar{t}_i s_i}$$

is the utilization of server i. Since $\rho_i < 1$, we must have $s_i > \bar{r}_i/\bar{t}_i$. Notice that $\rho_i = \lambda_i \bar{x}_i = \lambda_i \bar{r}_i/s_i = w_i/s_i$, where $w_i = \lambda_i \bar{r}_i$ is the expected amount of work received by server i in a unit of time. Since $\rho_i < 1$, we must have $s_i > w_i$. Notice that

$$\begin{aligned}
&\frac{1 + C_{x_i}^2}{(1/\rho_i)^2 + C_{x_i}^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{x_i}^2}{2\bar{t}_i(1-\rho_i)} \\
&= \frac{1 + \sigma_{x_i}^2/\bar{x}_i^2}{(\bar{t}_i/\bar{x}_i)^2 + \sigma_{x_i}^2/\bar{x}_i^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{x_i}^2}{2\bar{t}_i(1-\rho_i)} \\
&= \frac{\bar{x}_i^2 + \sigma_{x_i}^2}{\bar{t}_i^2 + \sigma_{x_i}^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{x_i}^2}{2\bar{t}_i(1-\rho_i)}.
\end{aligned}$$

The above equation for W_i includes some classic results as special cases. For instance, for an M/G/1 queue, we have $\sigma_{t_i}^2 = \bar{t}_i^2$ and

$$W_i = \frac{\lambda_i \overline{x_i^2}}{2(1-\rho_i)},$$

where $\overline{x_i^2} = \bar{x}_i^2 + \sigma_{x_i}^2$. This is exactly the well-known Pollaczek-Khinchin mean value formula ([14, p. 16]).

The average response time of tasks in server i is

$$\begin{aligned}
T_i &= \bar{x}_i + W_i \\
&= \bar{x}_i + \frac{1 + C_{x_i}^2}{(1/\rho_i)^2 + C_{x_i}^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{x_i}^2}{2\bar{t}_i(1-\rho_i)} \\
&= \bar{x}_i + \frac{\bar{x}_i^2 + \sigma_{x_i}^2}{\bar{t}_i^2 + \sigma_{x_i}^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{x_i}^2}{2(\bar{t}_i - \bar{x}_i)} \\
&= \frac{\bar{r}_i}{s_i} + \frac{\bar{r}_i^2/s_i^2 + \sigma_{r_i}^2/s_i^2}{\bar{t}_i^2 + \sigma_{r_i}^2/s_i^2} \cdot \frac{\sigma_{t_i}^2 + \sigma_{r_i}^2/s_i^2}{2(\bar{t}_i - \bar{r}_i/s_i)} \\
&= \frac{\bar{r}_i}{s_i} + \frac{\bar{r}_i^2 + \sigma_{r_i}^2}{\bar{t}_i^2 s_i^2 + \sigma_{r_i}^2} \cdot \frac{\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2}{2s_i(\bar{t}_i s_i - \bar{r}_i)} \\
&= \frac{\bar{r}_i}{s_i} + (\bar{r}_i^2 + \sigma_{r_i}^2)\left(\frac{\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2}{2s_i(\bar{t}_i s_i - \bar{r}_i)(\bar{t}_i^2 s_i^2 + \sigma_{r_i}^2)}\right),
\end{aligned}$$

which is viewed as a function of s_i, where $s_i > \bar{r}_i/\bar{t}_i$.

Let $\lambda = \lambda_1 + \lambda_2 + \cdots + \lambda_n$ be the total arrival rate. The average task response time in the data center with n servers is

$$\begin{aligned}
&T(s_1, s_2, \ldots, s_n) \\
&= \sum_{i=1}^{n} \left(\frac{\lambda_i}{\lambda}\right) T_i \\
&= \frac{1}{\lambda} \sum_{i=1}^{n} \lambda_i \left(\frac{\bar{r}_i}{s_i} + (\bar{r}_i^2 + \sigma_{r_i}^2)\left(\frac{\sigma_{t_i}^2 s_i^2 + \sigma_{r_i}^2}{2s_i(\bar{t}_i s_i - \bar{r}_i)(\bar{t}_i^2 s_i^2 + \sigma_{r_i}^2)}\right)\right),
\end{aligned}$$

where we view T as a function of server speeds $s_1, s_2, \ldots, s_n$.

APPENDIX B
ACCURACY OF THE G/G/1 APPROXIMATION

Our study has employed approximations of the average waiting time and the average response time. Some experiments have been conducted to examine the accuracy of the approximations.

Let us consider a server i. Assume that the interarrival time t_i has a hyper-Erlang distribution with probability density function (pdf)

$$f(t) = \sum_{j=1}^{k_a} w_{a,j} \left(\frac{\lambda_j e^{-\lambda_j t} (\lambda_j t)^{\gamma_{a,j}-1}}{(\gamma_{a,j}-1)!} \right),$$

where $w_{a,1}+w_{a,2}+\cdots+w_{a,k_a} = 1$. (Notice that hyper-Erlang distributions include hyperexponential distributions, exponential distributions, chi-square distributions, and Erlang distributions as special cases.) Similarly, assume that the execution time x_i also has a hyper-Erlang distribution with pdf

$$f(x) = \sum_{j=1}^{k_b} w_{b,j} \left(\frac{\mu_j e^{-\mu_j x} (\mu_j x)^{\gamma_{b,j}-1}}{(\gamma_{b,j}-1)!} \right),$$

where $w_{b,1} + w_{b,2} + \cdots + w_{b,k_b} = 1$. Then, we have

$$\bar{t}_i = \sum_{j=1}^{k_a} w_{a,j} \cdot \frac{\gamma_{a,j}}{\lambda_j},$$

and

$$\bar{x}_i = \sum_{j=1}^{k_b} w_{b,j} \cdot \frac{\gamma_{b,j}}{\mu_j}.$$

Let $r = \bar{t}_i/\bar{x}_i$. For arbitrary server utilization ρ, we adjust λ_j as $\lambda_j \leftarrow \rho r \lambda_j$, for all $1 \le j \le k_a$. This results in the actual server utilization to be ρ.

For interarrival time, we set $k_a = 3$, $w_{a,1} = 0.3$, $w_{a,2} = 0.3, w_{a,3} = 0.4, \gamma_{a,1} = 2, \gamma_{a,2} = 3, \gamma_{a,3} = 4, \lambda_1 = 1, \lambda_2 = 2, \lambda_3 = 3$. For execution time, we set $k_b = 2$, $w_{b,1} = 0.4$, $w_{b,2} = 0.6$, $\gamma_{b,1} = 3$, $\gamma_{b,2} = 4$, $\mu_1 = 2$, $\mu_2 = 3$. We generate 1,000,000 random tasks, simulate a G/G/1 server, record the response time of each task, and report the average response time. In Table 5, we show our experimental results. For $\rho = 0.10, 0.15, 0.20, \ldots, 0.95$, we show the simulation results of the average response time and the 99% confidence interval (C.I.). We also show the theoretical approximation and its relative error, i.e.,

$$(\text{approximation} - \text{simulation})/\text{simulation} \times 100\%.$$

It is observed that the 99% C.I. is very small (less than 0.24%). In other words, the simulation results are very reliable and robust, and very close to the real values of the average response time. Furthermore, the theoretical approximation is very accurate with relative error no more than 7%, i.e., the theoretical approximation can be used in real applications with high accuracy.

ACKNOWLEDGMENTS

The author would like to thank the anonymous reviewers for their comments and suggestions on improving the manuscript.

REFERENCES

[1] *CMOS*. Accessed: Nov. 15, 2018. [Online]. Available: http://en.wikipedia.org/wiki/CMOS

[2] *Dynamic Voltage Scaling*. Accessed: Nov. 15, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Dynamic_voltage_scaling

[3] *Internet of Things*. Accessed: Nov. 15, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_Things

[4] *Cisco CEO Pegs Internet of Things as $19 Trillion Market*. Accessed: Nov. 15, 2018. [Online]. Available: http://www.bloomberg.com/news/articles/2014-01-08/cisco-ceo-pegs-internet-of-things-as-19-trillion-market

[5] *Internet of Things Global Standards Initiative*. Accessed: Nov. 15, 2018. [Online]. Available: http://www.itu.int/en/ITU-T/gsi/iot/Pages/default.aspx

[6] A. Al-Dulaimy, W. Itani, A. Zekri, and R. Zantout, "Power management in virtualized data centers: State of the art," *J. Cloud Comput., Adv., Syst. Appl.*, vol. 5, p. 6, Apr. 2016.

[7] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," *Adv. Comput.*, vol. 82, pp. 47–111, Jan. 2011.

[8] J. Cao, K. Li, and I. Stojmenovic, "Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers," *IEEE Trans. Comput.*, vol. 63, no. 1, pp. 45–58, Jan. 2014.

[9] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, Apr. 1992.

[10] D. Evans, "The Internet of Things: How the next evolution of the Internet is changing everything," Cisco, San Jose, CA, USA, White Paper, 2011. Accessed: Nov. 15, 2018. [Online]. Available: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[11] S. Kumar and R. Buyya, "Green cloud computing and environmental sustainability," in *Harnessing Green IT: Principles and Practices*, S. Murugesan and G. R. Gangadharan, Eds. Chichester, U.K.: Wiley, 2012.

[12] J. Huang, R. Li, J. An, D. Ntalasha, F. Yang, and K. Li, "Energy-efficient resource utilization for heterogeneous embedded computing systems," *IEEE Trans. Comput.*, vol. 66, no. 9, pp. 1518–1531, Sep. 2017.

[13] "Enhanced Intel SpeedStep technology for the Intel pentium M processor," Intel, Santa Clar, CA, USA, White Paper, Mar. 2004. [Online]. Available: http://download.intel.com/design/network/papers/30117401.pdf

[14] L. Kleinrock, *Queueing Systems: Computer Applications*, vol. 2. New York, NY, USA: Wiley, 1976.

[15] F. Kong and X. Liu, "A survey on green-energy-aware power management for datacenters," *ACM Comput. Surv.*, vol. 47, no. 2, 2014, Art. no. 30.

[16] L. Lefèvre and A.-C. Orgerie, "Designing and evaluating an energy efficient cloud," *J. Supercomput.*, vol. 51, no. 3, pp. 352–373, 2010.

[17] K. Li, "Optimal power allocation among multiple heterogeneous servers in a data center," *Sustain. Comput., Inform. Syst.*, vol. 2, pp. 13–22, Mar. 2012.

[18] K. Li, "Improving multicore server performance and reducing energy consumption by workload dependent dynamic power management," *IEEE Trans. Cloud Comput.*, vol. 4, no. 2, pp. 122–137, Apr./Jun. 2016.

[19] S. U. R. Malik, K. Bilal, S. U. Khan, B. Veeravalli, K. Li, and A. Y. Zomaya, "Modeling and analysis of the thermal properties exhibited by cyberphysical data centers," *IEEE Syst. J.*, vol. 11, no. 1, pp. 163–172, Mar. 2017.

[20] S. Mittal. "Power management techniques for data centers: A survey." Accessed: May 7, 2014. [Online]. Available: http://arxiv.org/abs/1404.6681v2

[21] S. Murugesan and G. R. Gangadharan, Eds., *Harnessing Green IT: Principles and Practices*, Hoboken, NJ, USA: Wiley, 2012.

[22] "Scaling up energy efficiency across the data center industry: Evaluating key drivers and barriers," Natural Resources Defense Council, New York, NY, USA, Issue Paper IP:14-08-A, Aug. 2014. [Online]. Available: https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IP.pdf

[23] A.-C. Orgerie, M. D. de Assuncao, and L. Lefevre, "A survey on techniques for improving the energy efficiency of large-scale distributed systems," *ACM Comput. Surv.*, vol. 46, no. 4, 2014, Art. no. 47.

[24] A. Rahman, X. Liu, and F. Kong, "A survey on geographic load balancing based data center power management in the smart grid environment," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 214–233, 1st Quart., 2014.

[25] J. Stewart, *Multivariable Calculus*, 2nd ed. Pacific Grove, CA, USA: Brooks/Cole Publishing Company, 1991.

[26] Y. Tian, C. Lin, and K. Li, "Managing performance and power consumption tradeoff for multiple heterogeneous servers in cloud computing," *Cluster Comput.*, vol. 17, no. 3, pp. 943–955, 2014.

[27] C. B. Westphall, C. M. Westphall, S. R. Villarreal, G. A. Geronimo, and J. Werner. (May 2014). *Green Clouds Through Servers, Virtual Machines and Network Infrastructure Management*. [Online]. Available: http://www.academia.edu/18007775/Green_Clouds_through_Servers_Virtual_Machines_and_Network_Infrastructure_Management

[28] B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "Theoretical and practical limits of dynamic voltage scaling," in *Proc. 41st Design Autom. Conf.*, 2004, pp. 868–873.

KEQIN LI (F'15) is a SUNY Distinguished Professor of computer science with the State University of New York and a Distinguished Professor with Hunan University. He has published over 620 journal articles, book chapters, and refereed conference papers. His current research interests include cloud computing, fog computing, 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, and intelligent and soft computing. He is an IEEE Fellow. He received several best paper awards. He currently serves or has served on the editorial boards of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, the IEEE TRANSACTIONS ON SERVICES COMPUTING, and the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.