Energy-Efficient Stochastic Task Scheduling on Heterogeneous Computing Systems

Kenli Li, Xiaoyong Tang, and Keqin Li, *Senior Member, IEEE*

Abstract—In the past few years, with the rapid development of heterogeneous computing systems (HCS), the issue of energy consumption has attracted a great deal of attention. How to reduce energy consumption is currently a critical issue in designing HCS. In response to this challenge, many energy-aware scheduling algorithms have been developed primarily using the dynamic voltage-frequency scaling (DVFS) capability which has been incorporated into recent commodity processors. However, these techniques are unsatisfactory in minimizing both schedule length and energy consumption. Furthermore, most algorithms schedule tasks according to their average-case execution times and do not consider task execution times with probability distributions in the real-world. In realizing this, we study the problem of scheduling a bag-of-tasks (BoT) application, made of a collection of independent stochastic tasks with normal distributions of task execution times, on a heterogeneous platform with deadline and energy consumption budget constraints. We build execution time and energy consumption models for stochastic tasks on a single processor. We derive the expected value and variance of schedule length on HCS by Clark's equations. We formulate our stochastic task scheduling problem as a linear programming problem, in which we maximize the weighted probability of combined schedule length and energy consumption metric under deadline and energy consumption budget constraints. We propose a heuristic energy-aware stochastic task scheduling algorithm called ESTS to solve this problem. Our algorithm can achieve high scheduling performance for BoT applications with low time complexity $O(n(M + \log n))$, where n is the number of tasks and M is the total number of processor frequencies. Our extensive simulations for performance evaluation based on randomly generated stochastic applications and real-world applications clearly demonstrate that our proposed heuristic algorithm can improve the weighted probability that both the deadline and the energy consumption budget constraints can be met, and has the capability of balancing between schedule length and energy consumption.

Index Terms—Bag-of-tasks, dynamic voltage-frequency scaling, energy consumption, heterogeneous computing system, schedule length, stochastic task scheduling, probability

1 Introduction

1.1 Motivation

During the last few years, the use of high performance heterogeneous computing systems (HCS) to run scientific and commercial applications such as data mining, fractal calculations and simulations, computational biology, streaming services of IPTV, seismic data process, and weather prediction, has increased rapidly. Many current supercomputing systems in the world's top 500 supercomputers are heterogeneous computing systems. For example, the fastest ''Tianhe-2'' has 16,000 nodes, each with two Intel Xeon IvyBridge processors and three Xeon Phi processors for a total of 3,120,000 computing cores. However, the high costs of energy, system performance, reliability, and various environmental issues have forced the high performance computing sector to reconsider some of its old practices with an aim to create more sustainable HCS. Among these issues, energy consumption is currently a critical issue in designing high performance HCS. The ''Tianhe-2'' has power consumption of 17.808 MW, and the average power consumption of the top 10 systems is 6.675 MW as of June 17, 2013 [1]. With an approximate price of $100/\mathrm{MW} \cdot \mathrm{Hour}$, their energy costs are \$1,780.80 and \$667.50 per hour, which are beyond the acceptable budget of many HCS operators. The magnitude of such consumption will be even greater if the energy consumption of the associated cooling systems is also considered.

To reduce energy consumption in HCS, various techniques including *dynamic voltage-frequency scaling* (DVFS), resource hibernation, and memory optimizations have been investigated and developed [2]. Among these techniques, DVFS is perhaps the most appealing method for reducing energy consumption [3], [4], [5], [6], [7], [29]. Most modern high performance microprocessors such as Intel Core Duo, AMD Athlon, and Transmeta's Crusoe processors have DVFS that allows a user to control energy consumption at runtime. DVFS aims to reduce dynamic energy consumption of a processor by scaling down its operational frequency and circuit supply voltage, based on the fact that energy consumption in a CMOS circuit has a direct relationship

- *K. Li is with the College of Information Science and Engineering, National Supercomputing Center in Changsha, Hunan University, Changsha 410082, China. E-mail: lkl510@263.net.*
- *X. Tang is with the College of Information Science and Engineering, National Supercomputing Center in Changsha, Hunan University, Changsha 410082, China, and also with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210046, China. E-mail: tang_313@163.com.*
- *K. Li is with the College of Information Science and Engineering, National Supercomputing Center in Changsha, Hunan University, Changsha 410082, China, and also with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA. E-mail: lik@newpaltz.edu.*

Manuscript received 21 July 2013; revised 7 Oct. 2013; accepted 10 Oct. 2013. Date of publication 20 Oct. 2013; date of current version 15 Oct. 2014. Recommended for acceptance by D. Wang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2013.270

with its frequency and the square of the supplied voltage [5], [6], [7], [8], [9], [10], [11], [36]. Thus, energy consumption can be saved by switching among a processor's voltages and frequencies during task execution.

In examining the existing research on energy-efficient computing, we realize that there is lack of investigation on two important issues in real applications. The first issue is a combined performance metric of schedule length and energy consumption in HCS. Task allocation on DVFS-enabled heterogeneous processors is conventionally addressed as an energy-aware task scheduling problem, which consists of two interdependent subproblems. The first subproblem is scheduling, i.e., to determine a heterogeneous processor to execute a task. The goal is to minimize schedule length, i.e., the execution time of a scheduled application. The second subproblem is slack reclamation, i.e., to exploit proper supply voltage and frequency on a per-task basis [5], [9], [13], [14]. The goal is to minimize energy consumption. However, existing scheduling techniques suffer from the fact that combining the optimum of each subproblem does not lead to the overall optimum of a combined performance metric for energy consumption and schedule length.

The second issue is energy-efficient scheduling of stochastic tasks in HCS. Most of the energy-aware and non-energy-aware scheduling researches on HCS assume that task execution time are fixed and deterministic which are known in advance, or a scheduler schedules tasks according to their average-case execution time [5], [7], [8], [9], [13], [14], [15], [18], [19], [37]. However, in real-world problems, it usually does not suffice to find good schedules for fixed and deterministic execution time or average-case execution time, since tasks usually contain conditional instructions and/or operations that could have different execution time for different inputs [20], [21], [22], [23], [24]. A natural and effective method to tackle this problem is to consider stochastic task scheduling, that is, to interpret task execution time as random variables and to measure the performance of an algorithm by its expected objective-value. Using a stochastic scheduling approach, we can develop algorithms that provide the capability to balance between schedule length and energy consumption, and satisfy time and energy consumption budget constraints with high probability.

The motivation and contribution of the present paper is to propose an energy-efficient stochastic task scheduling algorithm in HCS and to evaluate its performance using a combined performance metric of schedule length and energy consumption.

1.2 Related Research

See Section 1 of the supplementary material which is available in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2013.270.

1.3 Our Contributions

The present paper makes fundamental contributions in four aspects.

- First, we construct an energy-aware stochastic task scheduling architecture in heterogeneous computing environments, which incorporates HCS, BoT applications, stochastic tasks, an energy model, and time and energy budget constraints into consideration.
- Second, we propose a recursive method to compute the expected value and variance of schedule length by using Clark's equations, and we formulate our energy-aware stochastic task scheduling problem as a linear programming problem.
- Third, we propose a heuristic energy-aware stochastic task scheduling algorithm called ESTS, which is time efficient and practical. ESTS can be implemented in polynomial time, and our experiments are always finished in less than ten minutes.
- Fourth, our heuristic algorithm ESTS can improve the weighted probability that both the deadline and the energy consumption budget constraints can be met, and has the capability of balancing between schedule length and energy consumption.

2 System Models

This section describes the target heterogeneous computing systems, the task model, the energy model, and the stochastic task scheduling architecture used in our study.

2.1 Heterogeneous Computing Systems

A heterogeneous computing system (HCS) is a system that distributes data, processing, and program execution among different processors, such that each is best suited for specific computational tasks. An HCS can be a supercomputing system with heterogeneous cores, or a cluster of heterogeneous processors. A processor could be a general-purpose processor (GPP), a special-purpose processor (e.g., digital signal processor (DSP) or graphics processing unit (GPU)), or a co-processor. All these processors are connected by a special network, such as InfiniBand or Myrinet. In this paper, an HCS is modeled as a finite set P of p processors, and p_i represents the ith processor. Each heterogeneous processor $p_i \in P$ is DVFS-enabled, which allows a user to modify the processor's operational frequency and voltage, so that it is possible to dynamically adapt the processor's operating point to satisfy performance, power, and energy requirements. A typical DVFS-enabled processor p_i can operate with a set of discrete frequency-voltage pairs $(f_{i,k}, v_{i,k})$, where $f_{i,1} < f_{i,2} < \cdots < f_{i,M_i}$ and $v_{i,1} < v_{i,2} < \cdots < v_{i,M_i}$, and M_i is the frequency/voltage level of processor p_i. For example, Intel Core i7-2600K can operate with 5 frequencies ranging from 2.6 GHz to 3.5 GHz, and voltages ranging from 0.8 V to 1.375 V. For demonstration purpose, we illustrate two heterogeneous processors. One has 3 frequency levels and the other has 2 frequency levels. The parameters are listed in Table 1, where $P_{s,i}$ is static power dissipation and α_i is switched capacitance. These parameters will be explained in detail in Section 2.3.

2.2 Task Model

We consider scheduling BoT applications or parameter-sweep applications, which are applications whose tasks $v_1, v_2, \ldots, v_n$ are assumed to be independent and atomic [15], [16], [17], [19]. By independent we mean that there is no communication or dependency among tasks. Examples

TABLE 1
Parameters of Two Heterogeneous Processors

Processor	$P_{s,i}$	α_i	$(f_{i,k}, v_{i,k})$ 1	2	3
p_1	65.2	2.15×10^{-8}	$(1.8\times 10^9, 0.89)$	$(2.4\times 10^9, 1.12)$	$(3.0\times 10^9, 1.34)$
p_2	37.2	4.08×10^{-8}	$(2.\ \times 10^9, 0.98)$	$(3.0\times 10^9, 1.42)$	

of BoT applications include Monte Carlo simulations, tomographic reconstructions, massive searches (such as key breaking), and data mining algorithms. They are typical and common tasks in fields such as astronomy, bioinformatics, high energy physics, and many others. We use d to represent the deadline, and Eb to denote the energy consumption budget for a BoT application.

Let $[w_{j,i,k}]$ be an $n \times p \times M_{\max}$ matrix, where $w_{j,i,k}$ gives the estimated time to execute task v_j on processor p_i at frequency $f_{i,k}$. Here, $M_{\max}$ is the maximal operation level of a processor. While determining the exact execution time of a task on a processor remains a challenge, and the actual task execution time is random [20], [21], [30], we assume that task execution times are known in advance only as probability distributions. Tasks for solving real-world problems usually contain conditional instructions and/or operations that could have different execution times for different inputs. Therefore, task execution times are random variables and approximately follow normal distributions [22], [23], [25], [31]. We use the notation $w_{j,i,k} \sim N(\mu_{j,i,k}, \sigma^2_{j,i,k})$ to represent the fact that task v_j's execution time has a normal distribution, where $N(\mu, \sigma^2)$ is the normal distribution with expected value μ and variance σ^2.

The estimation of probability distribution function (PDF) of a task's execution time involves two steps, i.e., profiling its execution time and deriving the PDF of its execution time. Recently, a number of measurement based on building a historic table, using code profiling, or statistical prediction techniques, have been proposed to predict task execution times [15], [20], [21], [32], [29]. In this paper, we take the statistical prediction technique, and apply an effective histogram technique, which is devised and used in [29] to get normal distributions of task execution times. We believe that these techniques can be used to determine the PDF of task execution times in heterogeneous computing systems. Table 2 lists three tasks' execution times on two heterogeneous DVFS-enabled processors specified in Table 1.

2.3 Energy Model

The total energy consumption of a heterogeneous computing system depends on its processors, memory, disks, networks, fans, cooling system, and other components. Although there are much research work focusing on reducing system energy consumption, such as [3], [4], [13], we only consider processor energy consumption in the energy model of this paper, since processors use a significant portion of energy in the system [5], [7], [8], [11].

Power consumption of a CMOS-based processor consists of two parts, i.e., dynamic part that is the switching power of a CMOS circuit, and static part that is the leakage power of a CMOS circuit. When a processor p_i is in the execution model, the power consumption P_i of p_i is defined as [5], [11]:

$$P_i = P_{s,i} + P_{d,i}, \tag{1}$$

where $P_{s,i}$ is the static power dissipation, which is a constant, and $P_{d,i}$ is the dynamic power dissipation. When p_i is in the idle model, we have $P_i = P_{s,i}$. When p_i is operated at voltage level $v_{i,k}$ and frequency level $f_{i,k}$, the dynamic power can be estimated as [8], [9]

$$P_{d,i,k} = \alpha_i v^2_{i,k} f_{i,k}. \tag{2}$$

Thus, the power consumption of p_i is $P_{i,k} = P_{s,i} + P_{d,i,k}$. (The frequency-voltage pairs of two example heterogeneous processors are shown in Table 1.) The energy consumption $EC^j_{i,k}$ for executing task v_j on processor p_i at frequency $f_{i,k}$ is given by

$$\begin{aligned} EC^j_{i,k} &= P_{i,k} w_{j,i,k} \\ &= P_{s,i} w_{j,i,k} + P_{d,i,k} w_{j,i,k} \\ &= P_{s,i} w_{j,i,k} + \alpha_i v^2_{i,k} f_{i,k} w_{j,i,k}. \end{aligned} \tag{3}$$

2.4 Scheduling Architecture

See Section 2 of the supplementary material which is available online.

3 Problem Description

In this section, we first provide a motivational example to illustrate the basic concept and the complexity of scheduling stochastic tasks on heterogeneous computing systems with deadline and energy consumption budget constraints. Then, we introduce the method of computing the expected value and variance of schedule length and energy consumption on HCS. Finally, we formulate the problem of scheduling stochastic tasks on HCS to satisfy time and energy budget constraints with high probability as a linear programming problem.

3.1 A Motivational Example

See Section 3 of the supplementary material which is available online.

3.2 Execution Time on a Single Processor

An important fact about normal random variables is that if X_j is normally distributed with expected value μ_j and variance σ^2_j, for $j = 1, 2, \ldots, r$, then $X = \Sigma^r_{j=1} X_j$ is normally distributed with parameters $\Sigma^r_{j=1}\mu_j$ and $\Sigma^r_{j=1}\sigma^2_j$. In other words, $X \sim N(\sum^r_{j=1}\mu_j, \sum^r_{j=1}\sigma^2_j)$. This attribute can be

TABLE 2
Task Execution Time Matrix $[w_{j,i,k}]$

Task	p_1			p_2	
	$w_{j,1,1}$	$w_{j,1,2}$	$w_{j,1,3}$	$w_{j,2,1}$	$w_{j,2,2}$
v_1	$N(11.2, 2.4^2)$	$N(7.2, 2.0^2)$	$N(6.1, 3.2^2)$	$N(9.1, 1.2^2)$	$N(7.1, 3.3^2)$
v_2	$N(36.9, 4.5^2)$	$N(28.3, 5.1^2)$	$N(19.2, 3.0^2)$	$N(24.7, 7.1^2)$	$N(18.8, 0.9^2)$
v_3	$N(15.6, 3.1^2)$	$N(10.5, 1.8^2)$	$N(7.3, 0.9^2)$	$N(12.1, 5.4^2)$	$N(8.4, 2.1^2)$

applied to calculate the maximum execution time $C^i_{\max}$ of a single processor p_i. When tasks $v_1, v_2, \ldots, v_r$ are scheduled on p_i and executed with different frequencies $f_{i,k_1}, f_{i,k_2}, \ldots, f_{i,k_r}$, $C^i_{\max}$ is

$$C^i_{\max} = \sum_{j=1}^{r} w_{j,i,k_j}. \tag{4}$$

$C^i_{\max}$ follows a normal distribution with expected value and variance of

$$\begin{cases} E\big[C^i_{\max}\big] = \sum\limits_{j=1}^{r} \mu_{j,i,k_j}; \\ Var\big[C^i_{\max}\big] = \sum\limits_{j=1}^{r} \sigma^2_{j,i,k_j}. \end{cases} \tag{5}$$

3.3 Execution Time on an HCS

One of the primary objectives of scheduling BoT applications on heterogeneous computing systems is to minimize the schedule length, which is the maximum of all processors' maximum execution times $C^i_{\max}$ defined as

$$makespan = \mathrm{MAX}^p_{i=1}\big\{C^i_{\max}\big\}. \tag{6}$$

From the mathematical theory, we know that the above schedule length does not follow a normal distribution, even though all processors' maximum execution times $C^i_{\max}$ follow normal distributions. However, in pioneering work [31], [35], [36], Clark developed a method to estimate the expected value and variance of the maximum of random variables that are normally distributed. Hence, Clark's equations can be applied recursively to find the desired expected value and variance of makespan. Note that

$$\begin{aligned} makespan &= \mathrm{MAX}^p_{i=1}\big\{C^i_{\max}\big\} \\ &= \mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}, \ldots, C^p_{\max}\big\} \\ &= \mathrm{MAX}\big\{\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}, C^3_{\max}, \ldots, C^p_{\max}\big\}. \end{aligned}$$

As the tasks are independent, the random variables $C^1_{\max}$, $C^2_{\max}, \ldots, C^p_{\max}$ are all independent of each other, and the correlation coefficient between any pair among them is zero, i.e., $\rho_{x,y} = \rho(C^x_{\max}, C^y_{\max}) = 0$, for all $1 \le x \ne y \le p$. Computing the expected value and variance of $\mathrm{MAX}\{C^1_{\max}, C^2_{\max}\}$ with $\rho_{1,2} = 0$ by using Clark's equations can be done as follows:

$$\begin{aligned} &E\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big] \\ &\quad = E\big[C^1_{\max}\big]\Phi(\xi_{1,2}) + E\big[C^2_{\max}\big]\Phi(-\xi_{1,2}) + \varepsilon_{1,2}\psi(\xi_{1,2}) \\ &\quad = E\big[C^2_{\max}\big] + \big(E\big[C^1_{\max}\big] - E\big[C^2_{\max}\big]\big)\Phi(\xi_{1,2}) \\ &\qquad + \varepsilon_{1,2}\psi(\xi_{1,2}), \end{aligned} \tag{7}$$

where $\varepsilon_{1,2} = \sqrt{Var[C^1_{\max}] + Var[C^2_{\max}] - 2\rho_{1,2}\chi_{1,2}} = \sqrt{Var[C^1_{\max}] + Var[C^2_{\max}]}$, $\rho_{1,2} = 0$, $\xi_{1,2} = (E[C^1_{\max}] - E[C^2_{\max}])/\varepsilon_{1,2}$, $\chi_{1,2} = \sqrt{Var[C^1_{\max}]Var[C^2_{\max}]}$, $\psi(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2}$, and

$$\Phi(x) = \int_{-\infty}^{x} \psi(t)dt = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{1}{2}t^2} dt.$$

The variance of $\mathrm{MAX}\{C^1_{\max}, C^2_{\max}\}$ is presented by the second Clark's equation. That is,

$$\begin{aligned} &Var\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big] \\ &\quad = \big(E^2\big[C^1_{\max}\big] + Var\big[C^1_{\max}\big]\big)\Phi(\xi_{1,2}) \\ &\qquad + \big(E^2\big[C^2_{\max}\big] + Var\big[C^2_{\max}\big]\big) \\ &\Phi(-\xi_{1,2}) + \big(E\big[C^1_{\max}\big] + E\big[C^2_{\max}\big]\big)\varepsilon_{1,2}\psi(\xi_{1,2}) \\ &\qquad - E^2\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big]. \end{aligned} \tag{8}$$

Next, let us consider $\mathrm{MAX}\{C^1_{\max}, C^2_{\max}, C^3_{\max}\} = \mathrm{MAX}\{\mathrm{MAX}\{C^1_{\max}, C^2_{\max}\}, C^3_{\max}\}$. The correlation coefficient $\rho_{1,2,3}$ becomes the correlation coefficient between $\mathrm{MAX}\{C^1_{\max}, C^2_{\max}\}$ and $C^3_{\max}$. It is clear that $\rho_{1,2,3} = 0$. The expected value of $\mathrm{MAX}\{C^1_{\max}, C^2_{\max}, C^3_{\max}\}$ is

$$\begin{aligned} &E\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}, C^3_{\max}\big\}\big] \\ &\quad = E\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big]\Phi(\xi_{1,2,3}) \\ &\qquad + E\big[C^3_{\max}\big]\Phi(-\xi_{1,2,3}) + \varepsilon_{1,2,3}\psi(\xi_{1,2,3}), \end{aligned}$$

where $\varepsilon_{1,2,3} = \sqrt{Var[\mathrm{MAX}\{C^1_{\max}, C^2_{\max}\}] + Var[C^3_{\max}]}$, and $\xi_{1,2,3} = (E[\mathrm{MAX}\{C^1_{\max}, C^2_{\max}\}] - E[C^3_{\max}])/\varepsilon_{1,2,3}$. The variance of $\mathrm{MAX}\{C^1_{\max}, C^2_{\max}, C^3_{\max}\}$ is

$$\begin{aligned} &Var\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}, C^3_{\max}\big\}\big] \\ &\quad = \big(E^2\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big] \\ &\qquad + Var\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big]\big)\Phi(\xi_{1,2,3}) \\ &\qquad + \big(E^2\big[C^3_{\max}\big] + Var\big[C^3_{\max}\big]\big)\Phi(-\xi_{1,2,3}) \\ &\qquad + \big(E\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}\big\}\big] \\ &\qquad\quad + E\big[C^3_{\max}\big]\big)\varepsilon_{1,2,3}\psi(\xi_{1,2,3}) \\ &\qquad - E^2\big[\mathrm{MAX}\big\{C^1_{\max}, C^2_{\max}, C^3_{\max}\big\}\big]. \end{aligned}$$

In a similar way, Clark's equations can be used recursively for $p-1$ steps to determine the expected value and variance of $makespan$.

3.4 Problem Statement

In this paper, we set $X^j_{i,k} = 1$ if task v_j is scheduled on processor p_i at frequency level k, and set $X^j_{i,k} = 0$ otherwise.

According to Section 3.2, processor p_i's maximum execution time $C^i_{\max}$ has a normal distribution, i.e.,

$$C^i_{\max} \sim N\left(\sum_{j=1}^{n}\sum_{k=1}^{M_i} X^j_{i,k}\mu_{j,i,k}, \sum_{j=1}^{n}\sum_{k=1}^{M_i} X^j_{i,k}\sigma^2_{j,i,k}\right). \quad (9)$$

The probability that a given deadline d will be met is

$$\begin{aligned} &Pr_{makespan} \\ &= P[makespan \le d] \\ &= P\left[\mathrm{MAX}\left\{C^1_{\max}, C^2_{\max}, \ldots, C^p_{\max}\right\} \le d\right] \\ &= P\left[C^1_{\max} \le d\right] \times P\left[C^2_{\max} \le d\right] \times \cdots \times P\left[C^p_{\max} \le d\right] \\ &= F_1(d) \times F_2(d) \times \cdots \times F_p(d) \\ &= \prod_{i=1}^{p} F_i(d), \end{aligned} \quad (10)$$

where $F_i(x)$ is the cumulative density function (CDF) of $C^i_{\max}$.

The total energy consumption TE on an HCS is all processors' static energy consumption and dynamic energy consumed by task execution. Thus, we have

$$TE = makespan \sum_{i=1}^{p} P_{s,i} + \sum_{j=1}^{n}\sum_{i=1}^{p}\sum_{k=1}^{M_i} X^j_{i,k} P_{d,i,k} w_{j,i,k}.$$

If n is sufficiently large such that all processors' processing times are very close, we have

$$\begin{aligned} TE &\approx \sum_{j=1}^{n}\sum_{i=1}^{p}\sum_{k=1}^{M_i} X^j_{i,k} P_{i,k} w_{j,i,k} \\ &= \sum_{i=1}^{p} C^i_{\max} P_{s,i} + \sum_{j=1}^{n}\sum_{i=1}^{p}\sum_{k=1}^{M_i} X^j_{i,k}\alpha_i v^2_{i,k} f_{i,k} w_{j,i,k}. \end{aligned}$$

As task execution times are independent, the expected value and variance of TE are computed as

$$\begin{cases} E[TE] = \sum_{j=1}^{n}\sum_{i=1}^{p}\sum_{k=1}^{M_i} X^j_{i,k} P_{i,k}\mu_{j,i,k}; \\ Var[TE] = \sum_{j=1}^{n}\sum_{i=1}^{p}\sum_{k=1}^{M_i} X^j_{i,k} P_{i,k}\sigma^2_{j,i,k}. \end{cases} \quad (11)$$

If there is an energy consumption budget of Eb for this workload, then the probability that the energy consumption budget constraint is satisfied is

$$\begin{aligned} Pr_{TE} &= P[TE \le Eb] \\ &= \int_{-\infty}^{Eb} \frac{1}{\sqrt{2\pi Var[TE]}} \exp\left(-\frac{(t - E[TE])^2}{2Var[TE]}\right) dt. \end{aligned} \quad (12)$$

Thus, the weighted probability that both the deadline and the energy consumption budget constraints are met can be defined as

$$Pr_{System} = \theta \times Pr_{makespan} + (1 - \theta) \times Pr_{TE}, \quad (13)$$

where θ is a weighting parameter and calculated along with the calculation of task execution time PDF. In this study, we first use statistical prediction and histogram technique to get the time and energy consumption of workload. Then, we compute the weighting parameter θ according to the trade-off between schedule length probability and energy consumption probability. The scheduling objective of our problem is to maximize Pr_{System}, which can be formulated as a linear programming problem:

$$\begin{cases} \text{Maximize } Pr_{System}, \text{ such that} \\ X^j_{i,k} = 0 \text{ or } 1, \text{ and } \sum_{i=1}^{p}\sum_{k=1}^{M_i} X^j_{i,k} = 1, \ \forall j. \end{cases} \quad (14)$$

4 The Scheduling Algorithm

In this section, we describe our proposed energy-aware stochastic task scheduling algorithm ESTS for BoT applications, which aims to schedule stochastic tasks on DVFS-enabled heterogeneous processors with high weighted probability that both the deadline and the energy consumption budget constraints are satisfied.

4.1 Approximate Weight of Task Execution Time

In stochastic task scheduling, the WSEPT (*weighted shortest expected processing time*) policy schedules task according to the following order:

$$\frac{w_1}{E[w(v_1)]} \ge \frac{w_2}{E[w(v_2)]} \ge \cdots \ge \frac{w_n}{E[w(v_n)]},$$

where $w(v_j)$ denotes average execution time of task v_j on an HCS defined as

$$w(v_j) = \sum_{i=1}^{p}\sum_{k=1}^{M_i} w_{j,i,k} \Big/ \sum_{i=1}^{p} M_i.$$

If all weights are equal, it becomes the SEPT (*shortest expected processing time*) policy, where we always execute the task having the shortest expected processing time first [25], [26], [28]. However, the real execution time of a stochastic task does not equal to its expected value in most cases, which leads to lower efficiency of WSEPT, SEPT, and LEPT scheduling policies. As pointed out in [20], [25], [26], [35], the real execution time of a stochastic task is affected by the variance of the task's execution time. Variance is an important measure of the amount of variation within the values of a task's execution time, taking account of all possible values and their probabilities.

Thus, the variance of a task's execution time is considered throughout this paper. By taking the variance into account, the approximate weight of task v_j's execution time $Aw(v_j)$ on an HCS is defined as

$$Aw(v_j) = \begin{cases} E\left[w(v_j)\right] + \sqrt{Var\left(w(v_j)\right)} \\ \qquad \text{if } Var\left(w(v_j)\right)/E\left[w(v_j)\right]^2 \le 1; \\ E\left[w(v_j)\right]\left(1 + \frac{1}{\sqrt{Var\left(w(v_j)\right)}}\right), \quad \text{otherwise.} \end{cases}$$

It should be pointed out that this definition is first examined in our previous work [20] and the experimental results show good performance in heterogeneous computing systems. On the other hand, the approximate weight $Aw(v_j)$ is defined only by itself and not dependent on other tasks. So, this method can also be applied to independent BoT workload of this study. As the task execution time of v_j on processor p_i at frequency $f_{i,k}$ is assumed to follow a normal distribution with expected value $\mu_{j,i,k}$ and

TABLE 3
Deadline d and Energy Consumption Budget Eb

Tasks	200	400	600	800	1000	1200	1400	1600	1800	2000
HCS1 d	175	320	470	610	765	910	1070	1200	1350	1500
HCS1 Eb	260000	505000	763000	1011300	1280000	1520000	1770000	2011000	2280000	2550000
HCS2 d	185	335	490	635	785	925	1070	1210	1360	1510
HCS2 Eb	275000	557500	795000	1100000	1350000	1595000	1847000	2110000	2390000	2700000

variance $\sigma^2_{j,i,k}$, the approximate weight of task v_j is (see equation (15) at bottom of page)

In our proposed ESTS scheduling algorithm, we use the approximate weight $Aw(v_j)$ to sort the tasks of a BoT application and schedule tasks according to this scheduling list.

4.2 The ESTS Algorithm

The goal of ESTS is to get the shortest schedule length and the minimal energy consumption under deadline and energy budget constraints. Therefore, we try to search the optimal heterogeneous processor/frequency for each task, which aims to achieve shorter task complete time and lower energy consumption.

In order to get shorter task complete time, we introduce the average expected completion time $CTA(v_j)$ for task v_j, which is given as follows,

$$CTA(v_j) = \frac{\sum_{i=1}^{p}\sum_{k=1}^{M_i}(\mu_{j,i,k} + p_{i,mean})}{\sum_{i=1}^{p} M_i}, \tag{16}$$

where $p_{i,mean}$ is the expected value of the completion time for all tasks scheduled on processor p_i. The rationale behind $CTA(v_j)$ is based on the fact that if task v_j's finish time is less than $CTA(v_j)$, it will be a good choice. For processor p_i and its frequency $f_{i,k}$, the completion time $CT(v_j, f_{i,k})$ of task v_j is

$$CT(v_j, f_{i,k}) = N\Big(\mu_{j,i,k} + p_{i,mean}, \sigma^2_{j,i,k} + p_{i,variance}\Big), \tag{17}$$

where $p_{i,variance}$ is the variance of the completion time for all tasks scheduled on processor p_i. As task execution time follows a normal distribution, task v_j's completion time probability $Pr_{CT}(v_j, f_{i,k})$ on processor p_i with frequency $f_{i,k}$ can be computed by

$$\begin{aligned} &Pr_{CT}(v_j, f_{i,k}) \\ &\quad = P\big[CT(v_j, f_{i,k}) \le CTA(v_j)\big] \\ &\quad = \int_{-\infty}^{CTA(v_j)} \frac{1}{\sqrt{2\pi\Big(\sigma^2_{j,i,k} + p_{i,variance}\Big)}} \\ &\qquad \times \exp\left(-\frac{\big(t - (\mu_{j,i,k} + p_{i,mean})\big)^2}{2\Big(\sigma^2_{j,i,k} + p_{i,variance}\Big)}\right) dt \\ &\quad = \Phi\left(\frac{CTA(v_j) - (\mu_{j,i,k} + p_{i,mean})}{\sqrt{\sigma^2_{j,i,k} + p_{i,variance}}}\right), \end{aligned} \tag{18}$$

where $\Phi(x)$ is the CDF of a standard normal distribution.

According to Eq. (3), the expected value of task v_j's average energy consumption on an HCS is

$$\begin{aligned} AE(v_j) &= \frac{\sum_{i=1}^{p}\sum_{k=1}^{M_i} E\Big[EC^j_{i,k}\Big]}{\sum_{i=1}^{p} M_i} \\ &= \frac{\sum_{i=1}^{p}\sum_{k=1}^{M_i}\Big(P_{s,i} + \alpha_i v^2_{i,k} f_{i,k}\Big)\mu_{j,i,k}}{\sum_{i=1}^{p} M_i}. \end{aligned} \tag{19}$$

Therefore, the average energy consumption budget $Eb(v_j)$ for task v_j is

$$Eb(v_j) = \left(\frac{AE(v_j)}{\sum_{j=1}^{n} AE(v_j)}\right) Eb, \tag{20}$$

$$Aw(v_j) = \begin{cases} \dfrac{\sum_{i=1}^{p}\sum_{k=1}^{M_i}\mu_{j,i,k}}{\sum_{i=1}^{p} M_i} + \sqrt{\dfrac{\sum_{i=1}^{p}\sum_{k=1}^{M_i}\sigma^2_{j,i,k}}{\sum_{i=1}^{p} M_i}}, & \text{if } \dfrac{\sum_{i=1}^{p}\sum_{k=1}^{M_i}\sigma^2_{j,i,k}}{\Big(\sum_{i=1}^{p}\sum_{k=1}^{M_i}\mu_{j,i,k}\Big)^2} \sum_{i=1}^{p} M_i \le 1; \\ \dfrac{\sum_{i=1}^{p}\sum_{k=1}^{M_i}\mu_{j,i,k}}{\sum_{i=1}^{p} M_i}\left[1 + \sqrt{\dfrac{\sum_{i=1}^{p} M_i}{\sum_{i=1}^{p}\sum_{k=1}^{M_i}\sigma^2_{j,i,k}}}\right], & \text{otherwise} \end{cases} \tag{15}$$

TABLE 4
Task Parameters of Multimedia Applications

Application	Description	Expected Execution Time	Standard Deviation	Number
mpegplay	MPEG video decoder	113.4	38.9	30
madplay	MP3 audio decoder	43.1	34.8	40
tmndec	H.263 video decoder	89.5	34.7	20
toast	GSM speech encoder	5.6	4.7	30

where Eb is the energy consumption budget for a BoT application. Thus, task v_j's energy consumption probability $Pr_{EC}(v_j, f_{i,k})$ on processor p_i at frequency $f_{i,k}$ can be computed by

$$\begin{aligned} &Pr_{EC}(v_j, f_{i,k}) \\ &= P\left[EC^j_{i,k} \le Eb(v_j)\right] \\ &= \int_{-\infty}^{Eb(v_j)} \frac{1}{\sqrt{2\pi Var\left(EC^j_{i,k}\right)}} \\ &\quad \times \exp\left(-\frac{\left(t - E\left[EC^j_{i,k}\right]\right)^2}{2Var\left(EC^j_{i,k}\right)}\right) dt \\ &= \int_{-\infty}^{Eb(v_j)} \frac{1}{\sqrt{2\pi\left(P_{s,i} + \alpha_i v^2_{i,k} f_{i,k}\right)\sigma^2_{j,i,k}}} \\ &\quad \times \exp\left(-\frac{\left(t - \left(P_{s,i} + \alpha_i v^2_{i,k} f_{i,k}\right)\mu_{j,i,k}\right)^2}{2\left(P_{s,i} + \alpha_i v^2_{i,k} f_{i,k}\right)\sigma^2_{j,i,k}}\right) dt \\ &= \Phi\left(\frac{Eb(v_j) - \left(P_{s,i} + \alpha_i v^2_{i,k} f_{i,k}\right)\mu_{j,i,k}}{\sqrt{P_{s,i} + \alpha_i v^2_{i,k} f_{i,k}} \times \sigma_{j,i,k}}\right). \end{aligned} \quad (21)$$

According to the problem stated in Eq. (14), we define the following weighted probability for task v_j on processor p_i at frequency $f_{i,k}$,

$$\begin{aligned} Pr_S(v_j, f_{i,k}) = \theta \times Pr_{CT}(v_j, f_{i,k}) + (1-\theta) \\ \times Pr_{EC}(v_j, f_{i,k}). \end{aligned} \quad (22)$$

Our proposed ESTS scheduling algorithm assigns task v_j to a processor p_i at frequency $f_{i,k}$ in such a way that the probability $Pr_S(v_j, f_{i,k})$ is maximized. The pseudo code of our *E*nergy-aware *S*tochastic *T*ask *S*cheduling (ESTS) algorithm is shown in Algorithm 1.

Algorithm 1: The ESTS Algorithm.

Input: A BoT application and its task execution time distributions, a deadline, an energy budget, and an HCS.

Output: An assignment $(v_j, p_i, f_{i,k})$ of each task v_j to a processor p_i with a frequency $f_{i,k}$.

1: Compute the approximate weights of task execution times in Eq. (15);
2: Sort the tasks into a scheduling list by non-increasing order of $Aw(v_j)$ values;
3: **while** *there are unscheduled tasks in the list* **do**
4: Remove the first task v_j from the scheduling list;
5: Compute v_j's average expected completion time in Eq. (16);
6: Compute v_j's average energy consumption budget in Eq. (20);
7: **for** *each processor* p_i and every p_i*'s frequency* $f_{i,k}$ **do**
8: Compute v_j's completion time probability in Eq. (18);
9: Compute v_j's energy consumption probability in Eq. (21);
10: Compute v_j's weighted probability in Eq. (22);
11: **end**
12: Find the processor p_i and frequency $f_{i,k}$ which give the maximal weighted probability for task v_j;
13: Assign task v_j to processor p_i with frequency $f_{i,k}$, i.e., set $X^j_{i,k} = 1$;
14: Update the expected value and variance of processor p_i's completion time;
15: **end**
16: Compute the probability that the given deadline is met in Eq. (10), the probability that the energy consumption budget constraint is satisfied in Eq. (12), and the weighted probability in Eq. (13);
17: Return the scheduling solution.

4.3 Time Complexity

The time complexity of scheduling algorithms for BoT applications is usually expressed in terms of the number of tasks n and the total number of processor frequencies $M = \sum_{i=1}^{p} M_i$. The time-complexity of ESTS is analyzed as follows. Computing the approximate weights of task execution time in Step 1 and all the probabilities in Step 16 can be done in $O(n)$ time. Sorting the tasks in Step 2 can be performed in $O(n \log n)$ time. Selection of processor and frequency pairs for all tasks in Steps 3-15 can be done in $O(nM)$ time, where we notice that the most time consuming computation, i.e., Step 5 and the loop in Steps 7-11 can be performed in $O(M)$ time for each task. Thus, the overall time complexity of algorithm ESTS is $O(n(M + \log n))$.

5 Experimental Results and Discussion

To evaluate the performance of our proposed algorithm, we developed a discrete event simulator using C++. In the simulations, we compare our proposed scheduling algorithm ESTS with one existing well-known energy-aware scheduling algorithm EDF-DVS [36] and two traditional scheduling algorithms Min-Min and Max-Min [15], [17] on HCS. Here, ESTS1 denotes the ESTS algorithm with $\theta = 0.3$, and ESTS2 denotes the ESTS algorithm with $\theta = 0.8$.

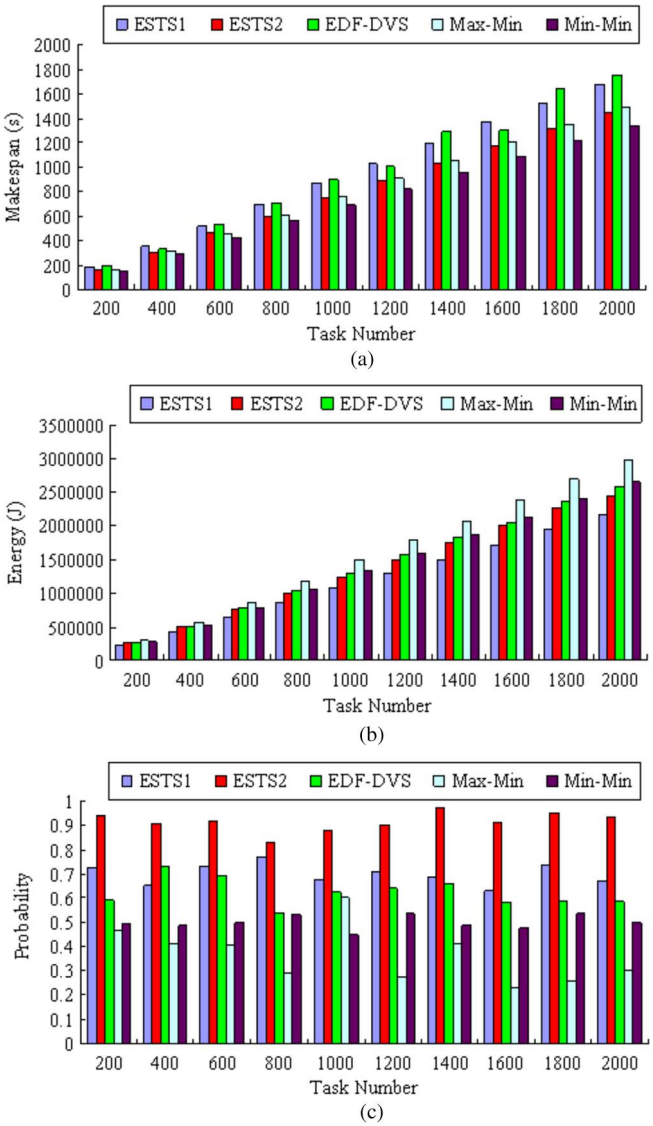

Fig. 1. Random tasks experimental results on HCS1. (a) Schedule length. (b) Energy consumption. (c) Weighted probability.

The performance metrics chosen for the comparison are the expected schedule length in Eq. (6), the expected total energy consumption in Eq. (11), and the weighted probability in Eq. (13). The comparison is intended not only to present quantitative results, but also to qualitatively analyze the results and to explain the reasons, for better understanding of the energy-aware stochastic scheduling problem.

5.1 Experimental Setting and Environment

5.1.1 DVFS-Enabled HCS

We have simulated two groups of DVFS-enabled HCS. The first group, we call HCS1, includes 8 Intel Core Duo with 4 frequencies, 4 Intel Xeon with 3 frequencies, 2 AMD Athlon with 3 frequencies, and 2 TI DSP with 2 frequencies, which are mostly based on Intel processors. The second group, we call HCS2, includes 2 Intel Core Duo with 4 frequencies, 10 AMD Athlon with 3 frequencies, 2 TI DSP with 2 frequencies, and 2 SPARC64 with 2 frequencies, which are mostly based on AMD processors.

5.1.2 Task Information

In order to gain practical insights into the effectiveness of the stochastic task scheduling approaches, we use both randomly generated stochastic BoT applications and real-world multimedia applications to test the scheduling effectiveness in a heterogeneous computing environment. In our study, a random task generator was implemented to generate stochastic tasks. The expected value and standard deviation of every task execution time on a processor are specified in the intervals [1, 30] and [3, 90] respectively, according to the processor's capacity. The number of tasks is in the range between 200 and 2,000, in steps of 200. These parameters of task execution time are obtained from Fortran+MPI parallel programming of molecular dynamics simulations for metal silver solidification at different cooling rates in our laboratory. The deadline d and energy consumption budget Eb are listed in Table 3.

Four practical multimedia programs, mpegplay, madplay, tmndec, and toast are adopted for evaluations, whose task parameters are shown in Table 4. The distributions of their average execution time on HCS1 are obtained by profiling their off-line traces. The deadline d is 420 s and the energy consumption budget Eb is 750,000J.

5.2 Performance Results for Random Tasks

The simulation results for randomly generated stochastic BoT applications are shown in Fig. 1, where each data point is the average of the simulation data obtained from 1,000 experiments.

Fig. 1 shows the simulation results of the ESTS, EDF-DVS, Max-Min, and Min-Min algorithms on the first group HCS1, i.e., a DVFS-enabled HCS, which is mostly based on Intel processors. We observe from Fig. 1a that the traditional algorithms Min-Min and Max-Min have shorter schedule length than ESTS and EDF-DVS, and the schedule length increases as the task number increases. This is mainly due to the fact that Min-Min and Max-Min schedule tasks according to the earliest completion time of a task without considering the task's energy consumption on HCS. In fact, all tasks scheduled by Min-Min and Max-Min are always running at the maximum frequency of each heterogeneous processor. As task energy consumption is proportional to the task execution voltage and frequency [8], task energy consumption increases rapidly as the frequency increases. Thus, each task has the minimum completion time and the highest energy consumption while running at the maximum processor frequency. This conclusion can be confirmed from Fig. 1b, which reveals that Min-Min consumes more energy than ESTS1 by 18.9 percent, ESTS2 by 6.3 percent, and EDF-DVS by 2.4 percent, and Max-Min consumes more energy than ESTS1 by 27.2 percent, ESTS2 by 15.9 percent, and EDF-DVS by 12.4 percent, respectively. Furthermore, Fig. 2(c) shows that the traditional algorithms Max-Min and Min-Min have the worst performance in terms of the weighted probability that both the deadline and the energy consumption budget constraints are met.

The experimental results of Fig. 1 also demonstrate that ESTS1 outperforms EDF-DVS by 2.4 percent, and ESTS2 outperforms EDF-DVS by 18.8 percent in terms of schedule

length; furthermore, ESTS1 outperforms EDF-DVS by 20.3 percent, and ESTS2 outperforms EDF-DVS by 4.2 percent in terms of energy consumption. Thus, the ESTS algorithm has better performance with respect to the weighted probability under deadline and energy consumption budget constraints. In fact, ESTS1 significantly outperforms EDF-DVS by 11.1 percent and ESTS2 significantly outperforms EDF-DVS by 32.0 percent. This phenomena can attribute to the fact that EDF-DVS schedules a task according to its average-case execution time and the actual execution time on an HCS is different from that.

We also observe from Fig. 1c that the best scheduling algorithm is ESTS2, which is better than ESTS1 by 23.4 percent, EDF-DVS by 32.0 percent, Max-Min by 60.2 percent, and Min-Min by 45.4 percent. These results reveal that the ESTS algorithm can improve system scheduling performance under deadline and energy budget constraints. This improvement is due to the fact that ESTS is capable of employing stochastic attributes of task execution times to improve the quality of scheduling. However, classical scheduling algorithms such as Max-Min and Min-Min are non-energy-aware, which merely assign a time optimal task to a processor without considering the task's execution time distribution and energy consumption. An energy-aware algorithm such as EDF-DVS selects tasks only according to their average-case or worst-case execution times.

More experimental results on HCS1 and HCS2 are provided in Section 4.1 of the supplementary material which is available online.

5.3 Performance Results for Real Applications

See Section 4.2 of the supplementary material which is available online.

6 Conclusion and Future Work

In this paper, our main objective is to improve the weighted probability that both a deadline and an energy consumption budget constraints are met. In fact, ESTS2 is better than EDF-DVS by 32.0 percent, Max-Min by 60.2 percent, and Min-Min by 45.4 percent on HCS1, and ESTS2 outperforms EDF-DVS by 19.8 percent, Max-Min by 63.3 percent, and Min-Min by 36.4 percent on HCS2. We believe that such performance improvement is significant in scientific research and not typically seen in the published literature. Although our algorithm does not guarantee optimal performance, the above performance gain is satisfactory for heuristic algorithms. We plan to propose an exact algorithm based on the method of dynamic programming, which may improve the performance. However, it is conceivable that an exact algorithm would have very high time complexity, which may need a few hours or even a few days to find a solution. We believe that an algorithm with such high complexity is not suitable for task scheduling in a real heterogeneous computing system.

The most significant merit of our proposed ESTS algorithm is its outstanding capability in dealing with tight deadline and energy budgets, i.e., its high weighted probability to satisfy a tight time deadline and a tight energy consumption budget constraint simultaneously. Such capability is critical in high-performance and highly energy-efficient heterogeneous computing systems. If a deadline and/or an energy consumption budget constraint is relaxed, the weighted probability will still be higher than those of EDF-DVS, Max-Min, and Min-Min.

We would like to mention a number of directions for further research.

- One important direction is to validate that task execution times follow normal distributions and to test the performance of our algorithm in a real heterogeneous computing system.
- It would be very interesting to deal with stochastic applications with dependence graphs such as pipelines, fork-join graphs, and general DAGs.
- Another direction is to investigate the reliability of a system and incorporate both reliability and energy consumption into task scheduling.

Acknowledgment

This research was partially funded by the Key Program of National Natural Science Foundation of China (Grant No. 61133005), National Science Foundation of China (Grant Nos. 61070057, 61370098, 61370095), the National Science Foundation for Distinguished Young Scholars of Hunan (12JJ1011), and a project supported by Scientific Research Fund of Hunan Provincial Education Department (Grant No. 12A062).

References

[1] TOP 500 Supercomputers, 2013. [Online]. Available: http://www.top500.org/lists /2013/06/

[2] V. Venkatachalam and M. Franz, ''Power Reduction Techniques for Microprocessor Systems,'' *ACM Comput. Survey*, vol. 37, no. 3, pp. 195-237, Sept. 2005.

[3] S. Zhuravlev, J.C. Saez, S. Blagodurov, A. Fedorova, and M. Prieto, ''Survey of Energy-Cognizant Scheduling Techniques,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1447-1464, July 2013.

[4] D. Li and J. Wu, ''Energy-Aware Scheduling for Frame-Based Tasks on Heterogeneous Multiprocessor Platforms,'' in *Proc. 41st ICPP*, pp. 430-439.

[5] N.B. Rizvandi, J. Taheri, and A.Y. Zomaya, ''Some Observations on Optimal Frequency Selection in DVFS-Based Energy Consumption Minimization,'' *J. Parallel Distrib. Comput.*, vol. 71, no. 8, pp. 1154-1164, Aug. 2011.

[6] H. Kimura, M. Sato, Y. Hotta, T. Boku, and D. Takahashi, ''Empirical Study on Reducing Energy of Parallel Programs Using Slack Reclamation by DVFS in a Power-Scalable High Performance Cluster,'' in *Proc. IEEE Int'l Conf. Cluster Comput.*, 2006, pp. 1-10.

[7] J. Lorch and A. Smith, ''PACE: A New Approach to Dynamic Voltage Scaling,'' *IEEE Trans. Comput.*, vol. 53, no. 7, pp. 856-869, July 2004.

[8] S. Garg, C. Yeo, A. Anandasivam, and R. Buyya, ''Environment-Conscious Scheduling of HPC Applications on Distributed Cloud-Oriented Data Centers,'' *J. Parallel Distrib. Comput.*, vol. 71, no. 6, pp. 732-749, June 2011.

[9] Y.C. Lee and A.Y. Zomaya, ''Energy Conscious Scheduling for Distributed Computing Systems Under Different Operating Conditions,'' *IEEE Trans. Parallel Distrib. Systems*, vol. 22, no. 8, pp. 1374-1381, Aug. 2011.

[10] N. Muralimanohar, K. Ramani, and R. Balasubramonian, ''Power Efficient Resource Scaling in Partitioned Architectures through Dynamic Heterogeneity,'' in *Proc. IEEE Int'l Symp. Perform. Anal. Syst. Softw.*, 2006, pp. 100-111.

[11] B. Mochocki, X. Hu, and G. Quan, ''A Unified Approach to Variable Voltage Scheduling for Nonideal DVS Processors,'' *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 9, pp. 1370-1377, Sept. 2004.
[12] I. Takouna, W. Dawoud, and C. Meinel, ''Energy Efficient Scheduling of HPC-Jobs on Virtualized Clusters Using Host and VM Dynamic Configuration,'' *Proc. ACM SIGOPS Oper. Syst. Rev.*, vol. 46, no. 2, pp. 19-27, July 2012.
[13] Z. Zong, A. Manzanares, X. Ruan, and X. Qin, ''EAD and PEBD: Two Energy-Aware Duplication Scheduling Algorithms for Parallel Tasks on Homogeneous Clusters,'' *IEEE Trans. Comput.*, vol. 60, no. 3, pp. 360-374, Mar. 2011.
[14] X. Zhong and C. Xu, ''System-Wide Energy Minimization for Real-Time Tasks: Lower Bound and Approximation,'' in *Proc. IEEE/ACM Int'l Conf. Comput.-Aided Des.*, Nov. 2006, pp. 516-521.
[15] F. Dong and S.G. Akl, ''Scheduling Algorithms for Grid Computing: State of the Art and Open Problems,'' School Comput., Queen's Univ., Kingston, ON, USA, Tech. Rep 2006-504, Jan. 2006.
[16] R. Freund, M. Gherrity, S. Ambrosius, M. Campbelly, M. Halderman, D. Hensgen, E. Keith, T. Kidd, M. Kussow, J.D. Lima, F. Mirabile, L. Moore, B. Rust, and H.J. Siegel, ''Scheduling Resources in Multi-User, Heterogeneous Computing Environments with SmartNet,'' in *Proc. 7th IEEE HCW*, Mar. 1998, pp. 184-199.
[17] T. Braun, D. Hensgen, R. Freund, H.J. Siegel, N. Beck, L. Boloni, M. Maheswaran, A. Reuther, J. Robertson, M. Theys, and B. Yao, ''A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems,'' *J. Parallel Distrib. Comput.*, vol. 61, no. 6, pp. 810-837, June 2001.
[18] M. Maheswaran, S. Ali, H. Siegel, D. Hensgen, and R. Freund, ''Dynamic Matching and Scheduling of a Class of Independent Tasks onto Heterogeneous Computing Systems,'' in *Proc. 8th Heterogeneous Comput. Workshop*, 1999, pp. 30-44.
[19] J.O. Gutierrez-Garcia and K.M. Sim, ''A Family of Heuristics for Agent-Based Elastic Cloud Bag-of-Tasks Concurrent Scheduling,'' *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1682-1699, Sept. 2013.
[20] X. Tang, K. Li, G. Liao, K. Fang, and F. Wu, ''A Stochastic Scheduling Algorithm for Precedence Constrained Tasks on Grid,'' *Future Gener. Comput. Syst.*, vol. 27, no. 8, pp. 1083-1091, Oct. 2011.
[21] M. Qiu and E.H.-M. Sha, ''Cost Minimization While Satisfying Hard/Soft Timing Constraints for Heterogeneous Embedded Systems,'' *ACM Trans. Des. Autom. Electron. Syst.*, vol. 14, no. 2, p. 25, Mar. 2009.
[22] J. Gu, X. Gu, and M. Gu, ''A Novel Parallel Quantum Genetic Algorithm for Stochastic Job Shop Scheduling,'' *J. Math. Anal. Appl.*, vol. 355, no. 1, pp. 63-81, July 2009.
[23] E. Ando, T. Nakata, and M. Yamashita, ''Approximating the Longest Path Length of a Stochastic DAG by a Normal Distribution in Linear Time,'' *J. Discrete Algorithms*, vol. 7, no. 4, pp. 420-438, Dec. 2009.
[24] J. Bruno, P. Downey, and G.N. Frederickson, ''Sequencing Tasks with Exponential Service Times to Minimize the Expected Flow Time or Makespan,'' *J. ACM*, vol. 28, no. 1, pp. 100-113, Jan. 1981.
[25] R.H. Möring, A.S. Schulz, and M. Uetz, ''Approximation in Stochastic Scheduling: The Power of LP-Based Priority Policies,'' *J. ACM*, vol. 46, no. 6, pp. 924-942, Nov. 1999.
[26] M. Scharbrodt, T. Schickingera, and A. Steger, ''A New Average Case Analysis for Completion Time Scheduling,'' *J. ACM*, vol. 53, no. 1, pp. 121-146, Jan. 2006.
[27] M.H. Rothkopf, ''Scheduling with Random Service Times,'' *Manage. Sci.*, vol. 12, no. 9, pp. 707-713, May 1966.
[28] R. Weber, ''Optimal Organization of Multi-Server Systems,'' Ph.D. dissertation, Cambridge Univ., Cambridge, U.K., 1980.
[29] W. Yuan and K. Nahrstedt, ''Energy-Efficient Soft Real-Time CPU Scheduling for Mobile Multimedia Systems,'' in *Proc. ACM SOSP*, 2003, pp. 149-163.
[30] C. Xian, Y. Lu, and Z. Li, ''Dynamic Voltage Scaling for Multitasking Real-Time Systems with Uncertain Execution Time,'' *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 8, pp. 1467-1678, Aug. 2008.
[31] S.C. Sarin, B. Nagarajan, and L. Liao, *Stochastic Scheduling. Cambridge, U.K.: Cambridge Univ. Press, 2010.*
[32] A.A. Khokhar, V.K. Prasanna, M.E. Shaaban, and C.L. Wang, ''Heterogeneous Computing: Challenges and Opportunities,'' *Computer*, vol. 26, no. 6, pp. 18-27, June 1993.
[33] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, ''Managing Server Energy and Operational Costs in Hosting Centers,'' *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, pp. 303-314, June 2005.
[34] L. Wang and Y. Lu, ''Efficient Power Management of Heterogeneous Soft Real-Time Clusters,'' in *Proc. Real-Time Syst. Symp.*, Barcelona, Spain, 2008, pp. 323-332.
[35] M. Skutella and M. Uetz, ''Stochastic Machine Scheduling with Precedence Constraints,'' *SIAM J. COMPUT.*, vol. 34, no. 4, pp. 788-802, 2005.
[36] K. Kim, R. Buyya, and J. Kim, ''Power Aware Scheduling of Bag-of-Tasks Applications with Deadline Constraints on DVS-Enabled Clusters,'' in *Proc. 7th IEEE Int'l Symp. Cluster Comput. Grid*, 2007, pp. 541-548.
[37] J. Cao, A. Chan, Y. Sun, S.K. Das, and M. Guo, ''A Taxonomy of Application Scheduling Tools for High Performance Cluster Computing,'' *J. Cluster Comput.*, vol. 9, no. 3, pp. 355-371, July 2006.
[38] C. Clark, ''The Greatest of a Finite Set of Random Variables,'' *Oper. Res.*, vol. 9, no. 2, pp. 145-162, Mar./Apr. 1961.
[39] D. Letic and V. Jevtic, ''The Distribution of Time for Clark's Flow and Risk Assessment for the Activities of Pert Network Structure,'' *Yugoslav J. Oper. Res.*, vol. 19, no. 1, pp. 195-207, 2009.
[40] V. Petrucci, E.V. Carreray, O. Loques, J. Leite, and D. Mosse, ''Optimized Management of Power and Performance for Virtualized Heterogeneous Server Clusters,'' in *Proc. 11th IEEE/ACM Int'l Symp. Cluster, Cloud Grid Comput.*, 2011, pp. 23-32.

Kenli Li received the PhD degree in computer science from Huazhong University of Science and Technology, China, in 2003. He was a visiting scholar at University of Illinois at Urbana-Champaign from 2004 to 2005. He is currently a Full Professor of computer science and technology at Hunan University and Associate Director of National Supercomputing Center in Changsha. His major research includes parallel computing, grid and cloud computing, and DNA computing. He has published more than 90 papers in international conferences and journals, such as *IEEE-TC*, *IEEE-TPDS*, *JPDC*, *ICPP*, *CCGrid*. He is an outstanding member of CCF.

Xiaoyong Tang received the MS and PhD degrees from Hunan University, China, in 2007 and 2013, respectively. His research interests include modeling and scheduling in distributed computing systems, parallel computing, distributed system reliability, and parallel algorithms. He is a reviewer of *IEEE-TC*, *IEEE-TPDS*, *JPDC*, *FGCS*, *JS*, and so on.

Keqin Li is a SUNY Distinguished Professor of computer science and an Intellectual Ventures endowed visiting chair professor at Tsinghua University, China. His research interests are mainly in design and analysis of algorithms, parallel and distributed computing, and computer networking. Dr. Li has over 290 research publications and has received several Best Paper Awards for his highest quality work. He is currently or has served on the editorial board of *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, *IEEE Transactions on Cloud Computing*, *Journal of Parallel and Distributed Computing*, *International Journal of Parallel, Emergent and Distributed Systems*, *International Journal of High Performance Computing and Networking*, and *Optimization Letters*. He is a Senior Member of the IEEE.