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 MW·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

\[ \text{Energy} = \text{Power} \times \text{Time} \]

with

\[ \text{Power} = \frac{\text{Voltage} \times \text{Current}}{\text{Resistance}} \]

where the operational voltage is scaled down and the current consumption is reduced at the cost of increased cycle time. However, DVFS has a number of disadvantages: it cannot achieve the energy saving of the static performance mode (low CPU cycle time), it requires some overhead, and it is difficult to adjust the cycle time under DVFS.

To address these issues, various algorithms have been proposed to schedule stochastic tasks on HCS. These algorithms mainly focus on minimizing the schedule length under energy consumption constraints. However, these algorithms are unsatisfactory in minimizing both schedule length and energy consumption. Moreover, most of these 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.
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 value and variance.

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 of p processors, and pi represents the ith processor. Each heterogeneous processor pi ∈ 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 pi can operate with a set of discrete frequency-voltage pairs {fi;k, vi;k}, where fi;1 < fi;2 < ... < fi;M, and vi;1 < vi;2 < ... < vi;M, and Mi is the frequency/voltage level of processor pi. 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 Pi,t is static power dissipation and αi 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 v1, v2, ..., vn 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
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 \( \begin{bmatrix} w_{j,i,k} \end{bmatrix} \) be an \( n \times p \times M_{\text{max}} \) 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, \( M_{\text{max}} \) 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 random variables and 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 \( \mu \) and variance \( \sigma^2 \).

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.

### Table 1

<table>
<thead>
<tr>
<th>Processor</th>
<th>( P_{s,i} )</th>
<th>( \alpha_i )</th>
<th>( f_{i,k} )</th>
<th>( v_{i,k} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_1 )</td>
<td>65.2</td>
<td>2.15 \times 10^{-8}</td>
<td>(1.8 \times 10^6, 0.89)</td>
<td>(3.0 \times 10^6, 1.34)</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>37.2</td>
<td>4.08 \times 10^{-8}</td>
<td>(2.0 \times 10^6, 0.98)</td>
<td>(3.0 \times 10^6, 1.42)</td>
</tr>
</tbody>
</table>

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., the 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]:

\[
P_i = P_{s,i} + P_{d,i}.
\]

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

\[
EC_{i,k} = P_{s,i}w_{j,i,k} + P_{d,i,k}w_{j,i,k}.
\]

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 \( \mu_j \) and variance \( \sigma^2_j \), for \( j = 1, 2, \ldots, r \), then \( X = \sum_{j=1}^{r} X_j \) is normally distributed with parameters \( \sum_{j=1}^{r} \mu_j \) and \( \sum_{j=1}^{r} \sigma^2_j \). In other words, \( X \sim N(\sum_{j=1}^{r} \mu_j, \sum_{j=1}^{r} \sigma^2_j) \). This attribute can be
applied to calculate the maximum execution time \( C_{i}^{\text{max}} \) of a single processor \( p_{i} \). When tasks \( v_{1}, v_{2}, \ldots, v_{n} \) are scheduled on \( p_{i} \) and executed with different frequencies \( f_{i,1}, f_{i,2}, \ldots, f_{i,k} \), \( C_{i}^{\text{max}} \) is

\[
C_{i}^{\text{max}} = \sum_{j=1}^{r} w_{j,i,k}, \tag{4}
\]

\( C_{i}^{\text{max}} \) follows a normal distribution with expected value and variance of

\[
\begin{align*}
E[C_{i}^{\text{max}}] &= \sum_{j=1}^{r} \mu_{j,i,k}; \\
\text{Var}[C_{i}^{\text{max}}] &= \sum_{j=1}^{r} \sigma_{j,i,k}^{2}.
\end{align*} \tag{5}
\]

### 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 \( C_{i}^{\text{max}} \) defined as

\[
\text{makespan} = \text{MAX}_{i=1}^{p} \{C_{i}^{\text{max}}\}. \tag{6}
\]

From the mathematical theory, we know that the above schedule length does not follow a normal distribution, even though all processors’ maximum execution times \( C_{i}^{\text{max}} \) follow normal distributions. However, in pioneering work \cite{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

\[
\text{makespan} = \text{MAX}_{i=1}^{p} \{C_{i}^{\text{max}}\} = \text{MAX}\{C_{1}^{\text{max}}, C_{2}^{\text{max}}, \ldots, C_{p}^{\text{max}}\} = \text{MAX}\{\text{MAX}\{C_{1}^{\text{max}}, C_{2}^{\text{max}}\}, C_{3}^{\text{max}}, \ldots, C_{p}^{\text{max}}\}. \tag{7}
\]

As the tasks are independent, the random variables \( C_{1}^{\text{max}}, C_{2}^{\text{max}}, \ldots, C_{p}^{\text{max}} \) are all independent of each other, and the correlation coefficient between any pair among them is zero, i.e., \( \rho_{x,y} = \rho(C_{x}^{\text{max}}, C_{y}^{\text{max}}) = 0 \), for all \( 1 \leq x \neq y \leq p \). Computing the expected value and variance of \( \text{MAX}\{C_{i}^{\text{max}}, C_{j}^{\text{max}}\} \) with \( \rho_{1,2} = 0 \) by using Clark’s equations can be done as follows:

\[
E[\text{MAX}\{C_{i}^{\text{max}}, C_{j}^{\text{max}}\}] = E[C_{i}^{\text{max}}] \Phi(\xi_{1,2}) + E[C_{j}^{\text{max}}] \Phi(-\xi_{1,2}) + \varepsilon_{1,2} \psi(\xi_{1,2}),
\]

\[
\text{Var}[\text{MAX}\{C_{i}^{\text{max}}, C_{j}^{\text{max}}\}] = (E^{2}[C_{i}^{\text{max}}] + \text{Var}[C_{i}^{\text{max}}]) \Phi(\xi_{1,2}) + (E^{2}[C_{j}^{\text{max}}] + \text{Var}[C_{j}^{\text{max}}]) \Phi(-\xi_{1,2}) + 2 \varepsilon_{1,2} \psi(\xi_{1,2}). \tag{8}
\]

The variance of \( \text{MAX}\{C_{1}^{\text{max}}, C_{2}^{\text{max}}, C_{3}^{\text{max}}\} \) is presented by the second Clark’s equation. That is,

\[
\begin{align*}
\text{Var}[\text{MAX}\{C_{1}^{\text{max}}, C_{2}^{\text{max}}, C_{3}^{\text{max}}\}] &= (E^{2}[C_{1}^{\text{max}}] + \text{Var}[C_{1}^{\text{max}}]) \Phi(\xi_{1,2,3}) + (E^{2}[C_{2}^{\text{max}}] + \text{Var}[C_{2}^{\text{max}}]) \Phi(-\xi_{1,2,3}) + \varepsilon_{1,2,3} \psi(\xi_{1,2,3}),
\end{align*}
\]

where \( \xi_{1,2,3} = \sqrt{\text{Var}[C_{1}^{\text{max}}] + \text{Var}[C_{2}^{\text{max}}] + \text{Var}[C_{3}^{\text{max}}]} - 2 \rho_{1,2,3} \), \( \varepsilon_{1,2,3} = \sqrt{\text{Var}[C_{1}^{\text{max}}] + \text{Var}[C_{2}^{\text{max}}]} \psi(\xi_{1,2}) / \xi_{1,2,3} \), \( \chi_{1,2} = \sqrt{\text{Var}[C_{1}^{\text{max}}]} \text{Var}[C_{2}^{\text{max}}] \psi(\xi_{1,2}) / \varepsilon_{1,2} \), and

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

### 3.4 Problem Statement

In this paper, we set \( \chi_{i,k}^{l} = 1 \) if task \( v_{j} \) is scheduled on processor \( p_{i} \) at frequency level \( k \), and set \( \chi_{i,k}^{l} = 0 \) otherwise.
According to Section 3.2, processor \( p_i \)'s maximum execution time \( C_{\text{max}}^i \) has a normal distribution, i.e.,

\[
C_{\text{max}}^i \sim N \left( \sum_{j=1}^{n} \sum_{k=1}^{M_j} X_{j,k}^i \mu_{j,i,k}, \sum_{j=1}^{n} \sum_{k=1}^{M_j} X_{j,k}^i \sigma_{j,k}^2 \right).
\]

The probability that a given deadline \( d \) will be met is

\[
Pr_{\text{makespan}} = P[\text{makespan} \leq d] = P[C_{\text{max}}^1, C_{\text{max}}^2, \ldots, C_{\text{max}}^p \leq d] = P[C_{\text{max}}^1 \leq d] \times P[C_{\text{max}}^2 \leq d] \times \cdots \times P[C_{\text{max}}^p \leq d] = F_1(d) \times F_2(d) \times \cdots \times F_p(d) = \prod_{i=1}^{p} F_i(d),
\]

where \( F_i(x) \) is the cumulative density function (CDF) of \( C_{\text{max}}^i \).

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

\[
TE = \text{makespan} \sum_{i=1}^{p} P_{i,s} + \sum_{j=1}^{n} \sum_{i=1}^{p} \sum_{k=1}^{M_j} X_{j,k}^i P_{i,k} w_{j,i,k}.
\]

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

\[
TE \approx \sum_{j=1}^{n} \sum_{i=1}^{p} X_{j,k}^i P_{i,k} w_{j,i,k} = \sum_{i=1}^{p} C_{\text{max}}^i P_{i,s} + \sum_{j=1}^{n} \sum_{i=1}^{p} \sum_{k=1}^{M_j} X_{j,k}^i \sigma_{j,k}^2 f_{j,i,k} w_{j,i,k}.
\]

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

\[
\begin{align*}
E[TE] &= \sum_{j=1}^{n} \sum_{i=1}^{p} X_{j,k}^i P_{i,k} \mu_{j,i,k}; \\
\text{Var}[TE] &= \sum_{j=1}^{n} \sum_{i=1}^{p} \sum_{k=1}^{M_j} X_{j,k}^i P_{i,k} \sigma_{j,k}^2 f_{j,i,k}.
\end{align*}
\]

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

\[
Pr_{TE} = P[TE \leq Eb] = \int_{-\infty}^{Eb} \frac{1}{\sqrt{2\pi \text{Var}[TE]}} \exp \left( -\frac{(t - E[TE])^2}{2\text{Var}[TE]} \right) dt.
\]

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

\[
Pr_{\text{System}} = \theta \times Pr_{\text{makespan}} + (1 - \theta) \times Pr_{TE},
\]

where \( \theta \) 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 \( \theta \) according to the trade-off between schedule length probability and energy consumption probability. The scheduling objective of our problem is to maximize \( Pr_{\text{System}} \), which can be formulated as a linear programming problem:

\[
\begin{align*}
\text{Maximize} & \quad Pr_{\text{System}}, \quad \text{such that} \\
& X_{j,k}^i = 0 \text{ or } 1, \quad \text{and } \sum_{j=1}^{n} \sum_{k=1}^{M_j} X_{j,k}^i = 1, \forall j.
\end{align*}
\]

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 tasks according to the following order:

\[
\frac{w_1}{E[w(v_j)]} \geq \frac{w_2}{E[w(v_j)]} \geq \cdots \geq \frac{w_n}{E[w(v_j)]},
\]

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

\[
w(v_j) = \sum_{j=1}^{n} \sum_{k=1}^{M_j} w_{j,i,k} / \sum_{j=1}^{n} M_j.
\]

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

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

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 \( f_{j,i,k} \) is assumed to follow a normal distribution with expected value \( \mu_{j,i,k} \) and
where $\pi;mean$ is the expected value of the completion time for all tasks scheduled on processor $\pi$. The rationale behind $CTA(\pi;j)$ is based on the fact that if task $\pi;j$’s finish time is less than $CTA(\pi;j)$, it will be a good choice. For processor $\pi$, and its frequency $f_{i;k}$, the completion time $CT(\pi;j, f_{i;k})$ of task $\pi;j$ is

$$CT(\pi;j, f_{i;k}) = N\left(\mu_{j,i,k} + p_{i;mean}, \sigma^2_{j,i,k} + p_{i;variance}\right). \tag{17}$$

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

$$P_{CT}(\pi;j, f_{i;k}) = P\left[CT(\pi;j, f_{i;k}) \leq CTA(\pi;j)\right]$$

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{CTA(\pi;j) - (\mu_{j,i,k} + p_{i;mean})}{\sigma_{j,i,k} + p_{i;variance}}} e^{-\frac{(x - \mu_{j,i,k} - p_{i;mean})^2}{2(\sigma_{j,i,k} + p_{i;variance})^2}} dx$$

$$= \Phi\left(\left(\frac{CTA(\pi;j) - (\mu_{j,i,k} + p_{i;mean})}{\sqrt{\sigma^2_{j,i,k} + p_{i;variance}}}\right)\right), \tag{18}$$

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

According to Eq. (3), the expected value of task $\pi;j$’s average energy consumption on an HCS is

$$AE(\pi;j) = \frac{\sum_{i=1}^{p} \left(\sum_{k=1}^{M_i} E[C_{j}]\right)}{\sum_{i=1}^{p} M_i}$$

$$= \frac{\sum_{i=1}^{p} M_i}{\sum_{i=1}^{p} M_i} \left(\sum_{i=1}^{p} \left(P_{xs,i} + \alpha \sigma^2_{j,i,k} f_{i,k}\mu_{j,i,k}\right)\right)$$

$$= \frac{\sum_{i=1}^{p} M_i}{\sum_{i=1}^{p} M_i} \left(\sum_{i=1}^{p} M_i\right). \tag{19}$$

Therefore, the average energy consumption budget $Eb(\pi;j)$ for task $\pi;j$ is

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

where $\sum_{i=1}^{p} \sum_{k=1}^{M_i} \mu_{j,i,k}$, $\sum_{i=1}^{p} \sum_{k=1}^{M_i} \sigma^2_{j,i,k}$, $\sum_{i=1}^{p} M_i$ are the total completion time, total variance, and total tasks of a BoT, respectively.

The rationale behind $Aw(\pi;j)$ is based on the fact that if task $\pi;j$’s finish time is less than $CTA(\pi;j)$, it will be a good choice. For processor $\pi$, and its frequency $f_{i;k}$, the approximate weight $Aw(\pi;j)$ of task $\pi;j$ can be computed by

$$Aw(\pi;j) = \begin{cases} \sum_{i=1}^{p} \sum_{k=1}^{M_i} \mu_{j,i,k} \sqrt{\sum_{i=1}^{p} \sum_{k=1}^{M_i} \sigma^2_{j,i,k}} + \frac{\sum_{i=1}^{p} \sum_{k=1}^{M_i} \sigma^2_{j,i,k}}{\sum_{i=1}^{p} M_i} & \text{if } \sum_{i=1}^{p} \sum_{k=1}^{M_i} \mu_{j,i,k} \leq 1; \\
\sum_{i=1}^{p} \sum_{k=1}^{M_i} \mu_{j,i,k} \sqrt{\sum_{i=1}^{p} \sum_{k=1}^{M_i} \sigma^2_{j,i,k}} + \frac{\sum_{i=1}^{p} \sum_{k=1}^{M_i} \sigma^2_{j,i,k}}{\sum_{i=1}^{p} M_i} & \text{otherwise} \end{cases} \tag{15}$$
where \( Eb \) is the energy consumption budget for a BoT application. Thus, task \( v_j \)'s energy consumption probability \( P_{EC}(v_j, f_{i,k}) \) on processor \( p_i \) at frequency \( f_{i,k} \) can be computed by

\[
P_{EC}(v_j, f_{i,k}) = P\left[ EC_{i,k} \leq Eb(v_j) \right] = \frac{1}{\sqrt{2\pi} \text{Var}(EC_{i,k})} \exp\left( -\frac{(t - \frac{1}{2} EC_{i,k})^2}{2 \text{Var}(EC_{i,k})} \right) dt
\]

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 \( M = \sum_{i=1}^{M} M_i \). 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 \( \theta = 0.8 \).
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 (12J1011), and a project supported by Scientific Research Fund of Hunan Provincial Education Department (Grant No. 12A062).

REFERENCES


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, ICCP, 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.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.