IEICE TRANS. INF. & SYST., VOL.E93–D, NO.12 DECEMBER 2010

PAPER Special Section on Parallel and Distributed Computing and Networking

SLA-Based Scheduling of Bag-of-Tasks Applications on Power-Aware Cluster Systems

Kyong Hoon KIM[†a)], ***Member***, **Wan Yeon LEE**[††], **Jong KIM**[†††], ***and*** **Rajkumar BUYYA**[††††], ***Nonmembers***

SUMMARY Power-aware scheduling problem has been a recent issue in cluster systems not only for operational cost due to electricity cost, but also for system reliability. In this paper, we provide SLA-based scheduling algorithms for bag-of-tasks applications with deadline constraints on power-aware cluster systems. The scheduling objective is to minimize power consumption as long as the system provides the service levels of users. A bag-of-tasks application should finish all the sub-tasks before the deadline as the service level. We provide the power-aware scheduling algorithms for both time-shared and space-shared resource sharing policies. The simulation results show that the proposed algorithms reduce much power consumption compared to static voltage schemes.

key words: *power-aware, cluster systems, service level agreement, bag-of-tasks, resource allocation*

1. Introduction

The service-oriented computing environment provides seamless access and integration of computing resource for both providers and consumers. Providers deploy their resource in the form of services, while users can easily find the resource throughout the service-oriented architecture. In this environment, consumers and providers agree upon a kind of commitment, such as resource usage, price, quality-of-service, and so on. Service Level Agreements (SLAs) are those obligations between consumers and producers. Thus, this paper deals with a scenario in which users submit and run their jobs on the resource providers with SLAs.

As computational resources in Grid or distributed systems are predominantly based on clusters, we examine resource allocation problem in clusters. Traditional research interest in cluster systems has been high performance, such as high throughput, low turnaround time, load balancing, and so on. However, recent research has focused on reducing power consumption in cluster systems. The objective of power aware computing is to improve power management and consumption using power aware ability of system devices, such as processors, disks, and communication links.

There are two main reasons for need of power aware computing in cluster systems: *operational cost* and *system reliability*. One dominating factor in the operational cost of data centers comes from electricity cost consumed by server systems [1]. As the number of managed servers increases, data centers can consume as much electricity as a city [2], [3]. Another reason is related to reliability of systems due to increased temperature caused by large power consumption. It is well known that computing in high temperature is more error-prone than one in appropriate environment. The expected failure rate of an electronic device doubles for every 10°C increased temperature according to the Arrenhius' equation [4]. In addition, the increased number of nodes in a cluster system results in lowering availability of the system. Thus, efficient power management of cluster systems becomes important issue of data centers not only for reducing their operational cost but also for system reliability.

In order to provide the power-aware ability, there are two main approaches to build power-aware cluster platforms. The first is to design and develop high performance clusters with consideration of energy consumption. BlueGene/L [5], [6] is designed with system-on-chip technology to reduce power in processors and network links. Green Destiny [7] consists of 240 Transmeta processors which consume low power. Orion Multisystem [8] workstations also provide low-power cluster systems.

The second approach to build power-aware clusters is using DVS-enabled commodity systems. Many recent commodity processors support DVS with multiple operating points. Such cluster platforms include a 10 AMD Athlon64 cluster [2], NEMO with 16 Intel Pentium Ms [9], CAFfeine with 16 AMD Opterons [10], [11], and Clusters using Crusoe and Turion [14].

Many recent studies have been conducted to provide power reduction for scientific applications on power-aware cluster systems. Hsu and Feng [10] provide β-adaptation algorithm that automatically adapts CPU frequencies in a DVS-enabled run-time system. They define the intensity level of off-chip accesses as β and propose a method to estimate this β at run time. In [9], three distributed DVS scheduling strategies are proposed: using the CPUSPEED daemon, scheduling from the command-line, and scheduling within application. They develop a software framework to implement and evaluate various scheduling techniques. In [14], they provide a profile-based power-performance opti-

Manuscript received February 3, 2010.
Manuscript revised June 15, 2010.

[†]The author is with the Department of Informatics, Gyeongsang National University, Jinju, South Korea.
[††]The author is with the Department of Ubiquitous Computing, Hallym University, Chunchon, South Korea.
[†††]The author is with the Department of Computer Science and Engineering, POSTECH, Pohang, South Korea.
[††††]The author is with the Department of Computer Science and Software Engineering, The University of Melbourne, Carlton, VIC, Australia.
a) E-mail: khkim@gnu.ac.kr

DOI: 10.1587/transinf.E93.D.3194

mization to select an appropriate gear using DVS scheduling. Their work is based on the developed power-profiling system called PowerWatch.

Since MPI is a commonly used programming model for scientific applications, much effort has been done to reduce energy consumption for MPI programs. *Jitter* [2] addresses inter-node bottlenecks in MPI programs to save energy. It selects an appropriate gear based on the slack time to each synchronization point. In [9], they present a profile-based optimization in MIPCH. One recent research in [15] presents a transparent MPI run-time system which exploits communication phases in MPI programs to reduce energy. In [16], they reduce energy consumption of parallel sparse matrix applications modeled by MPI.

In addition, many studies on cluster computing have been done in order to support *Service Level Agreements* (SLAs) between users and resource providers. SLAs define the negotiated agreements between service providers and consumers and include *Quality of Service* (QoS) parameters, such as deadline. Although it is important to reduce the system power, QoS parameters specified in SLAs should not be violated or the degradation should be minimized. Most of previous work has focused on minimizing performance degradation due to power reduction.

In real-time systems, DVS technique is used in order to save energy consumption as well as to meet the task deadline. Many studies have been done on DVS real-time scheduling on single processor systems [22]–[25]. The basic idea is to slowdown the clock speed using slack time to the task deadline. They mostly focuses on a single processor, while this paper proposes the cluster-level power-aware computing with QoS requirements.

In this paper, we consider deadline as QoS metric in SLAs of users' applications. Few previous power-aware cluster platform has considered both QoS and energy consumption [17]. Thus, we focus on the problem of reducing energy consumption of applications with SLAs.

The rest of this paper is organized as follows. Section 2 describes the power-aware clsuter sytem model and SLA-based job admission control. In Sect. 3, the proposed DVS scheduling algorithms for both space-shared and time-shared approaches are explained. Simulation results are given in Sect. 4, and this paper concludes with Sect. 5.

2. SLA-Based Power-Aware Clsuter Systems

2.1 DVS-Based Cluster Systems

A cluster system is composed of multiple Processing Elements (PEs) and a central resource controller. Each PE executes submitted jobs as an independent processing unit so that it manages its own job queue and scheduler. When users submit their jobs to the cluster system, the resource controller plays a role for admission control based on information from PEs in the system. PEs are assumed to be homogeneous so that they provide the same processing performance, such as the same MIPS (Million Instruction Per Second) rating.

The main power consumption in CMOS circuits is composed of dynamic and static power. The dynamic energy consumption ($E_{dynamic}$) by a task is proportional to V_{dd}^2 and N_{cycl} ($E_{dynamic} = k_1 V_{dd}^2 N_{cycl}$), where V_{dd} is the supply voltage and N_{cycl} is the number of clock cycles of the task [18]. The DVS (Dynamic Voltage Scaling) scheme reduces the dynamic energy consumption by decreasing the supplying voltage, which results in slowdown of the execution time. As for static energy consumption (E_{static}), we use a fraction of the dynamic power consumption as an approximate value ($E_{static} = k_2 E_{dynamic}$), which is usually less than 30% [19], [20].

Let us consider that a task of L Million Instructions (MIs) is executed on a processor with V supply voltage. The energy consumption during the task execution is defined by Eq. (1) since the number of clock cycles is in proportion to the number of instructions. In Eq. (1), α is a proportional constant.

$$\begin{aligned} E &= E_{dynamic} + E_{static} \\ &= k_1 V^2 L + k_2 (k_1 V^2 L) = \alpha V^2 L \end{aligned} \tag{1}$$

We assume that the PE in a cluster system can adjust its supply voltage from V_1 to V_m discretely. The associated CPU frequency with each supply voltage V_i is denoted as f_i ($i = 1, \cdots, m$). The execution time of a task varies according to each frequency. We use the execution time model with frequency f_i as in Eq. (2) [10], [28].

$$T(f_i) = T(f_{max}) \times \beta \left(\frac{f_{max}}{f_i} - 1 \right) + 1 \tag{2}$$

where $T(f_{max})$ is the execution time of the task at the top frequency f_{max}. The parameter β represents CPU boundness of a task [10], [28].

The value of β of a task can be obtained by profiling of the execution behavior of a bag-of-tasks job. Feasibility studies of [12], [13] have been shown by recording the execution time of the job at each available voltage level, and then using the profile for DVS schems. Thus, we assume that the value of β is to be known at the time of submission.

Since tasks have different β values, the relative speed level of the PE with each supply voltage also depends on running tasks. For a given task j with β_j CPU-boundess, the relative speed level of the PE with each supply voltage V_i is denoted as $S_{i,j}$ and defined by Eq. (3).

$$S_{i,j} = \frac{T(f_i)}{T(f_{max})} = \beta_j \left(\frac{f_{max}}{f_i} - 1 \right) + 1 \tag{3}$$

2.2 Job Model

A job in this paper is considered to be a bag-of-tasks application [21], which consists of multiple independent tasks with no communication among each other. In order to obtain the job's result, these tasks should be completed. In addition, we specify deadline of a job as QoS parameter, so

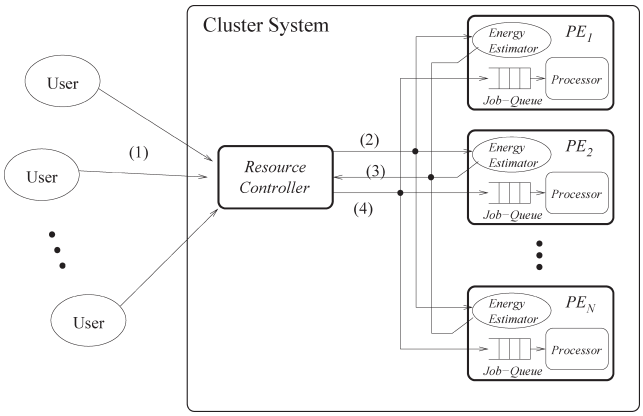

Fig. 1 Resource allocation framework.

that the job execution must be finished before the deadline.

Thus, a user's job is defined as $(p, \{l_1, l_2, \cdots, l_p\}, \beta, d)$, where p is the number of sub-tasks, l_i is the number of instructions of the i-th task in Million Instructions (MIs), β is the CPU-boundness parameter, and d is the deadline. The execution time of a task of length l_i varies according to the processor performance on which the task is run. Since the execution time is easily obtained from the task length on a processor, we use the task length as a task specification instead of the execution time. We also assume that the number of instructions of each task is known in advance.

2.3 SLA-Based Job Admission Control

When a cluster system receives a job from a user, the resource controller decides whether to accept the job. The proposed job admission scheme guarantees the deadlines of previously accepted jobs in the system. Thus, it allocates PEs to the new job as long as all the tasks can meet their deadlines. Figure 1 shows the job admission and execution steps in the system.

(1) *Job submission*: A user submits a new bag-of-tasks job with deadline to the cluster system.
(2) *Schedulability test & Energy estimation*: The resource controller requests schedulability and required energy consumption for each task of the job to all PEs.
(3) *Acknowledgement of schedulability and energy amount*: Each PE tests the schedulability of the new task and returns the estimated energy consumption in case of being schedulable.
(4) *Selection of PEs*: The resource controller selects the lowest-energy PE which can run each task.

Since a job consists of multiple tasks, steps from (2) to (4) are repeated until all the tasks are allocated. Provided that all tasks meet the deadlines, the resource controller accepts the new job. Otherwise, it rejects the job because it cannot guarantee the deadline of the job. Figure 2 describes the pseudo-algorithm of admission control of a new job.

For each sub-task of a job J, PEs checks the schedulability of the task (line 5). The function *schedulable*

```
Algorithm SLA_based_Job_Admission (J)
/* - J = (p, {l_1, ..., l_p}, β, d) : a new job
   - N : the number of processing elements
*/
1:  for i from 1 to p do
2:      PE_alloc ← null;
3:      energy_min ← MAX_VALUE;
4:      for k from 1 to N do
5:          if (schedulable (PE_k, l_i, d) == true) then
6:              energy_k ← energy_estimate (PE_k, l_i, β, d);
7:              if energy_k < energy_min then
8:                  energy_min ← energy_k;
9:                  PE_alloc ← PE_k;
10:             endif
11:         endif
12:     endfor
13:     if PE_alloc ≠ null then
14:         Allocate the i-th task of J to PE_alloc.
15:     else
16:         Cancel all jobs of J.
17:         return reject;
18:     endelse
19: endfor
20: return accept;
```

Fig. 2 SLA-based job admission and resource allocation.

$(proc, l, \beta, d)$ returns the schedulability of a task with length l and deadline d on the PE $proc$. And, the function *energy_estimate* () returns the estimated energy consumption on that PE. Since PE_{alloc} indicates the PE with the lowest energy consumption (line 7–10), the task is allocated to PE_{alloc} (line 13–14).

3. DVS-Based Cluster Scheduling

Each PE in the cluster system controls its supply voltage and schedules the jobs in its own job queue. A PE can share its processing unit among available jobs in the queue. The traditional sharing policies are classified into *space-sharing* and *time-sharing* schemes. The space-shared policy executes one task at a time, which is generally implemented by priority-driven scheduling algorithms. In time-shared policy, multiple tasks share the processing unit for their time slices. This paper provides one space-shared scheduling algorithm based on EDF (Earliest Deadline First) in Sect. 3.1 and one time-shared scheduling algorithm in Sect. 3.2.

3.1 EDF-Based DVS Scheduling

In this subsection, we focus on scheduling of tasks in a PE. A bag-of-tasks of a job are distributed to different PEs according to the energy consumption shown in Fig. 1. Thus, we denote the current available task set in the k-th PE as $T_k = \{\tau_{k,i}(e_{k,i}, d_{k,i}) | i = 1, \cdots, n_k\}$, where $e_{k,i}$ is the remaining execution time at the top frequency f_{max}, and $d_{k,i}$ are the

relative deadline of the i-th task.

Since the priority assignment scheme is based on EDF, T_k is sorted by the deadline so that it follows $d_{k,i} \leq d_{k,i+1}$, where $i = 1, \cdots, n_k - 1$. The scheduler always executes the earliest-deadline task in the queue.

Let us denote the current supply voltage level of PE_k as v_k. In order to derive the supply voltage, the temporary utilization, $u_{k,i}$, is defined as the following.

$$u_{k,i} = \frac{\sum_{j=1}^{i} e_{k,j}}{d_{k,i}}$$

The temporary utilization ($u_{k,i}$) implies the required processor utilization for task $\tau_{k,i}$ by EDF. The supply voltage control scheme is based on [22], [23], so that the highest-priority task's speed level under continuous voltage level, $\tilde{s}_k$, is defined by the following.

$$\tilde{s}_k = \max_{i=1}^{n_k}\{u_{k,i}\}$$

Since voltage levels in this paper are discrete from V_1 to V_m, the supply voltage v_k during $\tau_{k,1}$'s execution is the lowest V_i such that $S_{i,1}$ is greater than or equal to $\tilde{s}_k$, where S_i, 1 is the relative CPU speed level of $\tau_{k,1}$ at the voltage V_i. It is followed by Eq. (4). When PE_k dispatches the earliest-deadline task in its local queue, it changes the current voltage as v_k.

$$v_k = \min_{i=1}^{m}\{V_i | S_{i,1} \geq \tilde{s}_k\} \tag{4}$$

Let us consider a task set $T_k = \{\tau_{k,1}(1, 4), \tau_{k,2}(2, 6), \tau_{k,3}(2, 10)\}$ as an example. Let us assume that relative speeds of tasks under each supply voltage level are given in Table 1. At time 0, $u_{k,1}$, $u_{k,2}$, and $u_{k,3}$ are 1/4, 3/6, and 5/10, respectively, so that $\tilde{s}_k$ is 0.5. Since the lowest voltage with which the speed level of $\tau_{k,1}$ is more than or equal to 0.5 is 1.1 V, v_k at time 0 becomes 1.1 V.

At time 1.67, $u_{k,2}$ and $u_{k,3}$ are 0.46 ($\approx 2/(6 - 1.67)$) and 0.48 ($= (2 + 2)/(10 - 1.67)$). Since the least speed level $S_{i,2}$ of $\tau_{k,2}$ greater than 0.48 is 0.7 in Table 1, v_k at time 1.67 becomes 0.9 V. Figure 3 shows the scheduling result of the task set under EDF-DVS. And, the total energy consumption of Fig. 3 is approximately 6.63, where $\alpha = 10^{-6}$.

In the algorithm of Fig. 2, two functions are to be defined for schedulability test and energy estimation. Figure 4 shows the schedulability test algorithm based on EDF. When the temporary utilization of $\tau_{k,i}$ is greater than one, it cannot be scheduled by EDF.

As shown in Fig. 5, the energy estimation is calculated by the increased amount of energy consumption by a new task. In the function *energy_consumption* of Fig. 5, e_j and d_j are the remaining execution time and deadline of the j-th task in T.

Table 1 An example of energy model.

Voltage (V_i)	Frequency (f_i)	Relative Speed		
		$S_{i,1}$	$S_{i,2}$	$S_{i,3}$
0.9 V	0.8 GHz	0.4	0.7	0.55
1.1 V	1.2 GHz	0.6	0.8	0.7
1.3 V	1.6 GHz	0.8	0.9	0.85
1.5 V	2.0 GHz	1.0	1.0	1.0

3.2 Proportional Share-Based DVS Scheduling

The proportional share-based scheduling scheme provides tasks with the resource in proportion to each task's weight. Each task in PE_k should be given at least $e_{k,i}/d_{k,i}$ amount of processor utilization under the maximum clock speed in order to guarantee tasks' deadlines. Thus, we propose an adaptive proportional share scheduling that guarantees the minimum required proportion of each task.

The supply voltage of a processor is kept as low as required to meet tasks' deadlines. Let us consider a task set T_k of PE_k in the system. Since each task $\tau_{k,i}$ requires $e_{k,i}/d_{k,i}$ during its execution time, the required utilization of the task set is $\sum e_{k,i}/d_{k,i}$. The scheduling scheme should provide the minimum speed level $e_{k,i}/d_{k,i}$ for each task $\tau_{k,i}$. With consideration of sharing, the minimum relative speed level of $\tau_{k,i}$ is larger than or equal to $\sum e_{k,i}/d_{k,i}$. Thus, the supply voltage level is the maximum among such voltages, as shown in

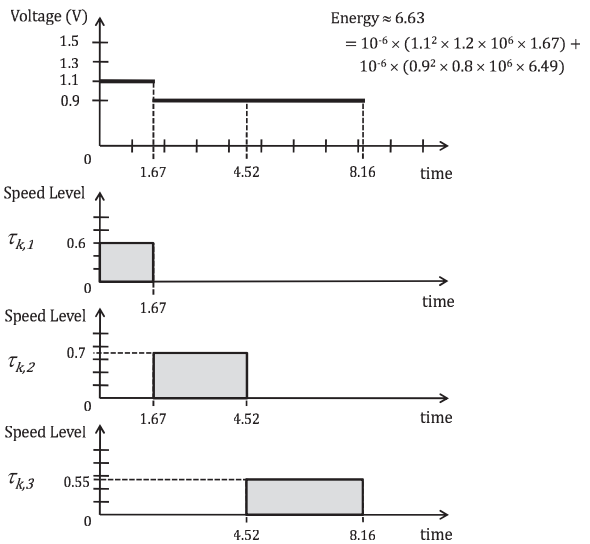

Fig. 3 An example of DVS-based EDF scheduling.

Algorithm schedulable_EDF (PE_k, e, d)
/* - e : the execution time of a task at frequency f_{max}
- d : the deadline of a task
*/
$T_{k'} \leftarrow T_k \cup \{(e, d)\}$;
Sort $T_{k'}$ in the order of deadline.
for i **from** 1 **to** $n_k + 1$ **do**
$u_{k',i} \leftarrow \frac{\sum_{j=1}^{i} e_{k',j}}{d_{k',i}}$;
if $u_{k',i} > 1$ **then return** *false*;
endfor
return *true*;

Fig. 4 Schedulability test for EDF.

Eq. (5).

$$v_k = \max_{i=1}^{n_k} \left\{ \min_{j=1}^{m} \left\{ V_j | S_{j,i} \geq \sum_{l=1}^{n_k} \frac{e_{k,l}}{d_{k,l}} \right\} \right\} \quad (5)$$

Under the current voltage level v_k, the share amount of each task $\tau_{k,i}$ should be defined. We denote the share amount of $\tau_{k,i}$ as $share_{k,i}$. In this paper, the sharing is in proportion to the required utilization, as shown in Eq. (6).

Algorithm energy_estimate_EDF (PE_k, e, β, d)
/* - e : the execution time of a task at frequency f_{max}
- d : the deadline of a task
*/
Construct the relative speed levels of the task using β.
$E_{current} \leftarrow$ *energy_consumption* (T_k, n_k);
$T_{k'} \leftarrow T_k \cup \{(e, d)\}$;
$E_{new} \leftarrow$ *energy_consumption* $(T_{k'}, n_k{+}1)$;
return $(E_{new} - E_{current})$;

function energy_consumption (T, n)
/* - T : a task set
- n : the number of tasks
- $t_{current}$: the current time
*/
$Energy \leftarrow 0$;
$time \leftarrow t_{current}$;
for i **from** 1 **to** n **do**
  for j **from** i **to** n **do** $u_j \leftarrow \frac{\sum_{k=1}^{j} e_k}{d_j}$;
  $\tilde{s} \leftarrow \max_{j=i}^{n} \{u_j\}$;
  $v \leftarrow \min_{j=1}^{m} \{V_j | S_{j,i} \geq \tilde{s}\}$;
  $s \leftarrow \min_{j=1}^{m} \{S_{j,i} | S_{j,i} \geq \tilde{s}\}$;
  $Energy \leftarrow Energy + \alpha v^2 l_i$;
  $time \leftarrow time + e_i/s$;
  for j **from** i **to** n **do** $d_j \leftarrow d_j - e_i/s$;
endfor
return *Energy*;

Fig. 5 Energy estimation for EDF.

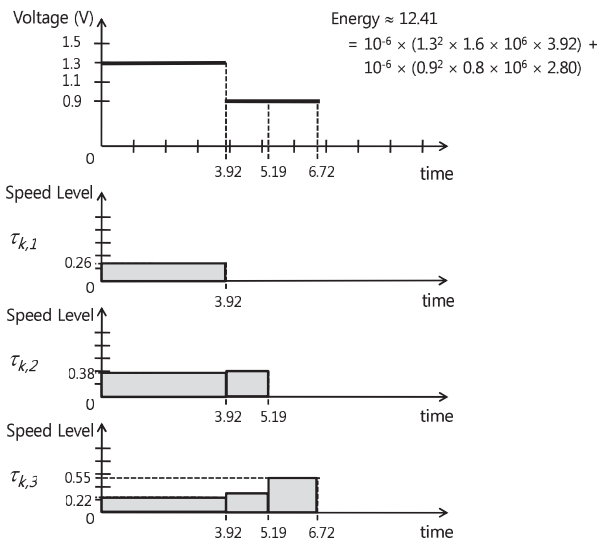

Fig. 6 An example of DVS-based proportional share scheduling.

$$share_{k,i} = \frac{e_{k,i}}{d_{k,i}} / \sum_{l=1}^{n_k} \frac{e_{k,l}}{d_{k,l}} \quad (6)$$

Thus, the actual relative speed level of task $\tau_{k,i}$ is defined as follows.

Actual relative speed of $\tau_{k,i} = S_{v_k,i} \times share_{k,i}$

Figure 6 shows the scheduling results of the same example $T_k = \{\tau_{k,1}(1, 4), \tau_{k,2}(2, 6), \tau_{k,3}(2, 10)\}$ in Sect. 3.1. The total energy consumption is approximately 12.41, where $\alpha = 10^{-6}$. Generally, the proportional sharing scheme runs at higher voltage levels and executes faster than the EDF-based scheme, which results in more energy consumption.

Schedulability test and energy estimation for the proportional share scheduling algorithm are similar to those of EDF, as shown in Fig. 4 and 5. The schedulability condition is that the summation of $e_{k,i}/d_{k,i}$ should be less than or equal to one. In order to calculate the energy consumption of a given task set, execution time of each task can be obtained based on $share_{k,i}$. And, the energy consumption of each task is defined in proportion to the share amount.

4. Simulation Results

In this section, we present simulation results of the proposed DVS-based cluster scheduling algorithms using the GridSim toolkit [26], [27]. Since the current GridSim toolkit does not support for power-aware simulations, we additionally developed DVS-related functions in the resource site of the GridSim toolkit. Thus, each processing element has an ability to adjust its supply voltage and clock speed. We create a cluster system with 32 DVS-enabled processors. Each processor is modeled with Athlon-64, so that the operating points of the processor are shown in Table 2. The performance of the processor at 2 GHz is assumed to be 10,000 MIPS. The processing performance under lower frequency is in proportion to the relative clock speed, as shown in Table 2.

We simulate two proposed DVS-based cluster scheduling algorithms of EDF and proportional share, which are denoted as **EDF-DVS** and **EDF-PShare**, respectively. For the performance comparison, we also simulate each scheduling algorithm under static voltage levels: one at the lowest supply voltage (=0.9 V) and the other at the highest supply voltage (=1.5 V).

In the simulations, we generate 1000 bag-of-tasks jobs. The number of tasks in a job is randomly selected from 2

Table 2 Operating points of simulated processor.

Frequency	Voltage	Relative Speed
0.8 GHz	0.9 V	0.4
1.0 GHz	1.0 V	0.5
1.2 GHz	1.1 V	0.6
1.4 GHz	1.2 V	0.7
1.6 GHz	1.3 V	0.8
1.8 GHz	1.4 V	0.9
2.0 GHz	1.5 V	1.0

and 32. The length of a task is in range from 600 MIs to 7,200 MIs. The job deadline is selected from 20% to 100% more than the average execution time on the processor at 1.4 GHz. For simplicity, we assume all tasks have the same β so that they have the same relative speeds under voltage leves, as shown in Table 2. The inter-arrival time between two consecutive jobs follows a Poisson distribution. In the simulations, we vary the mean time of the inter-arrival time from 2 minutes to 8 minutes.

The job acceptance ratio in Fig. 7 indicates how many jobs are accepted and meet their deadlines. The proposed DVS-enabled schemes show high job acceptance ratio. Since **EDF-1.5V** always executes processors at the maximum clock speed, it shows the highest acceptance ratio with the highest energy consumption, as shown in Fig. 8.

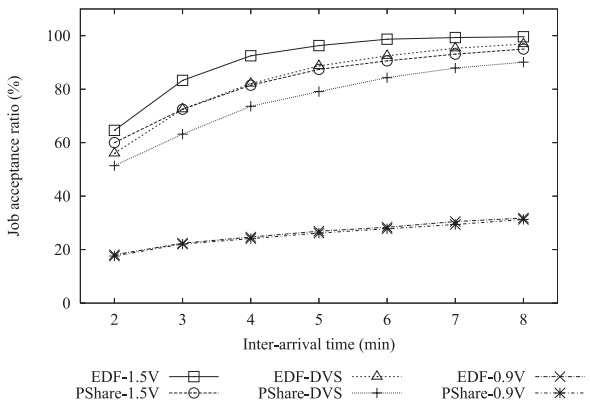

Fig. 7 Job acceptance ratio.

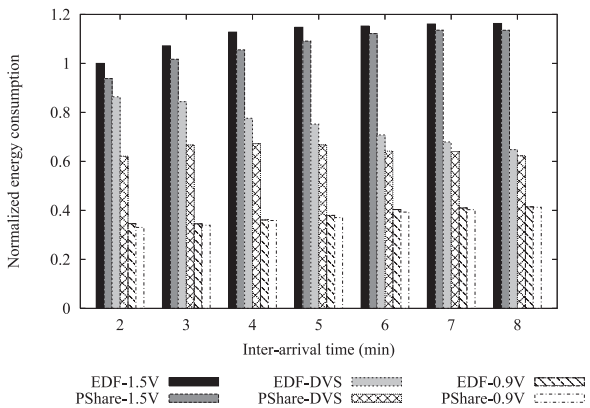

Fig. 8 Energy consumption of job normalized to EDF-1.5V at inter-arrival time of 2 mins.

Figure 8 shows the average energy consumption per accepted job in the simulations. **EDF-1.5V** and **PShare-1.5V** consume large amount of energy because they fix the supply voltage with 1.5 V. On the contrary, **EDF-0.9** and **PShare-0.9V** show lower energy consumption. However, they show poor job acceptance ratio less than 40% even under low overloaded condition, as shown in Fig. 7. The proposed DVS schemes consume less energy compared to 1.5 V-static schemes and show similar acceptance ratio.

In Fig. 8, static schemes show different energy consumption according to the system load. This is due to different number of subtasks of accepted jobs. A bag-of-tasks job consists of multiple subtasks, and Fig. 9 shows the average number of subtasks per accepted job in the simulations. As shown in Fig. 9, the average number of subtasks per job increases as the system load becomes lower. Thus, energy consumptions of accepted task of static schemes vary in proportion to the number of subtasks of accepted tasks. Figure 10 shows the normalized energy consumption per job in the simulations. As shown in Fig. 10, energy consumptions of static schemes show little change regardless of system load.

In Fig. 11, we compare two dynamic schemes, **EDF-**

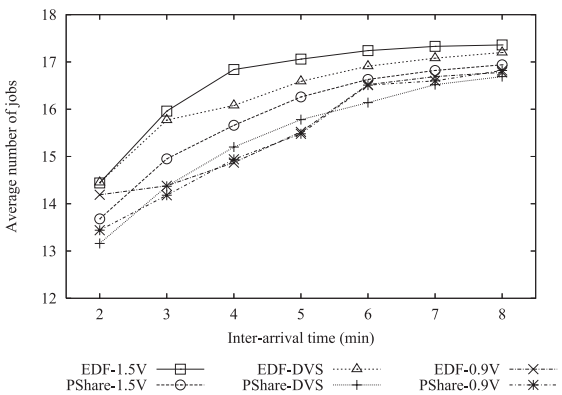

Fig. 9 Jobs per accepted task.

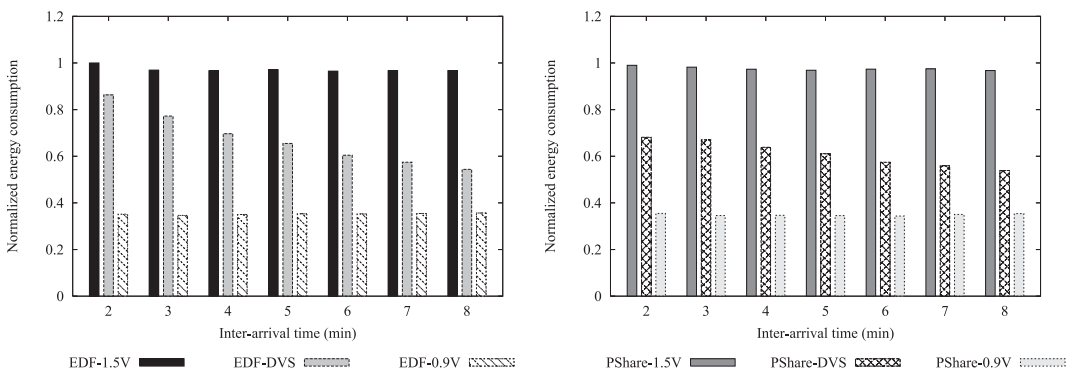

Fig. 10 Normalized energy consumption per job.

Table 3 Normalized performance of DVS.

Inter-arrival time (min)	EDF-DVS vs. EDF-1.5V		PShare-DVS vs PShare-1.5V	
	Energy Reduction (%)	Acceptance Degradation (%)	Energy Reduction (%)	Acceptance Degradation (%)
2	13.6	13.3	33.8	14.3
3	21.3	13.0	34.4	12.8
4	31.2	11.2	36.3	9.7
5	34.4	7.9	38.8	9.5
6	38.6	6.3	42.8	7.0
7	41.5	4.0	43.7	5.6
8	44.3	2.7	45.2	5.2

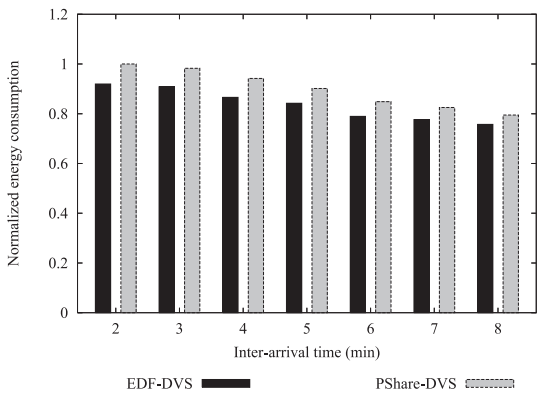

Fig. 11 EDF-DVS vs. PShare-DVS.

DVS and **PShare-DVS**. Since acceptacne ratio of **PShare-DVS** shows lower than that of **EDF-DVS**, as shown in Fig. 7. We evaluated **EDF-DVS** scheme with all the accepted jobs by **PShare-DVS** in the simulations. As shown in Fig. 11, **EDF-DVS** shows lower energy consumption than **PShare-DVS** since it generally runs at lower voltage level than **PShare-DVS**. Let us note that **EDF-DVS** shows higher energy consumption in Fig. 8 because its average number of subtasks per job is higher than **PShare-DVS**, as shown in Fig. 9.

Table 3 shows performance comparison between DVS and 1.5 V-static schemes in terms of success ratio and energy consumption. The improvement in energy reduction always shows more than degradation of acceptance ratio. As the system load becomes low, more improvement in energy saving is achieved and little loss of acceptance ratio is shown.

5. Conclusions

As recent processors support multiple supply voltage levels, power-aware cluster systems are easily built with commodity processors. SLA-based scheduling of applications on power-aware cluster systems can reduce much energy consumption, which decrease the operational cost and increases the system reliability. In this paper, we proposed power-aware scheduling algorithms for bag-of-tasks applications with deadline constraints on DVS-enabled cluster systems. The proposed scheduling algorithms select appropriate supply voltages of processing elements to minimize energy consumption.

Two DVS scheduling algorithms were considered: one for space-shared policy and the other for time-shared policy. Simulation results show that both DVS schemes reduce much energy consumption with little degradation of deadline missing. In this paper, we simply approximate static energy consumption as a fraction of dynamic power consumption. We will investigate various energy models on static energy consumption and apply it. Based on the proposed framework, we plan to conduct further research on SLA-based scheduling on multicore-based cluster systems.

Acknowledgments

This research was supported in part by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2010-(C1090-1031-0007)), and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2010-0015770).

References

[1] R. Bianchini and R. Rajamony, "Power and energy management for server systems," Computer, vol.37, no.11, pp.68–74, 2004.
[2] N. Kappiah, V.W. Freeh, and D.K. Lowenthal, "Just in time dynamic voltage scaling: Exploiting inter-node slack to save energy in mpi programs," Proc. ACM/IEEE SC 2005, pp.562–567, 2006.
[3] J. Markoff and S. Lohr, "Intel's huge bet turnsiffy," New York Times Technology Section, vol.3, no.2, Jan. 2002.
[4] W. Feng, "Making a case for efficient supercomputing," ACM Queue, vol.1, no.7, pp.54–64, 2003.
[5] Ibm research blue gene project.
[6] N.R. Adiga, et al., "An overview of the bluegene/l supercomputer," Proc. 2002 ACM/IEEE Conference on Supercomputing, 2002.
[7] W. Warren, E. Weigle, and W. Feng, "High-density computing: A 240-node beowulf in one cubic meter," Proc. 2002 ACM/IEEE Conference on Supercomputing, 2002.
[8] Orion multisystems.
[9] R. Ge, X. Feng, and K.W. Cameron, "Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters," Proc. ACM/IEEE SC 2005, 2005.
[10] C. Hsu and W. Feng, "A power-aware run-time system for high-performance computing," Proc. ACM/IEEE SC 2005, 2005.
[11] C. Hsu and W. Feng, "A feasibility analysis of power awareness in commodity-based high-performance clusters," Cluster 2005, 2005.
[12] X. Feng, R. Ge, and K. Cameron, "Power and energy profiling of

scientific applications on distributed systems," Proc. 19th Intl. Parallel and Distributed Processing Symp. (IPDPS 05), April 2005.
[13] V. Freeh, D. Lowenthal, R. Springer, F. Pan, and N. Kappiah, "Exploring the energy-time tradeoff in MPI programs on a power-scalable cluster," Proc. 19th Intl. Parallel and Distributed Processing Symp. (IPDPS 05), April 2005.
[14] Y. Hotta, M. Sato, H. Kimura, S. Matsuoka, T. Boku, and D. Takahashi, "Profile-based optimization of power performance by using dynamic voltage scaling on a pc cluster," Proc. 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2006.
[15] M.Y. Lim, V.W. Freeh, and D.K. Lowenthal, "Adaptive, transparent frequency and voltage scaling of communication phases in mpi programs," SC'06, 2006.
[16] S.W. Son, K. Malkowski, G. Chen, M. Kandemir, and P. Raghavan, "Integrated link/cpu voltage scaling for reducing energy consumption of parallel sparse matrix applications," Proc. 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2006.
[17] K.H. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters," Proc. 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007), 2007.
[18] T.D. Burd and R.W. Brodersen, "Energy efficient cmos microprocessor design," Proc. 28th Annual Hawaii International Conference on System Sciences, pp.288–297, 1995.
[19] J. Li and J.F. Martínez, "Dynamic power-performance adaptation of parallel computation on chip multiprocessors," Proc. 12th International Symposium on High-Performance Computer Architecture, 2006.
[20] C. Piguet, C. Schuster, and J.-L. Nagel, "Optimizing architecture activity and logic depth for static and dynamic power reduction," Proc. 2nd Annual IEEE Northeast Workshop on Circuits and Systems, pp.41–44, 2004.
[21] W. Cirne, F. Brasileiro, J. Sauve, N. Andrade, D. Paranhos, E. Santos-Neto, and R. Medeiros, "Grid computing for bag of tasks applications," Proc. 3rd IFIP Conference on E-Commerce, E-Business and E-Government, 2003.
[22] T. Pering and R. Brodersen, "Energy efficient voltage scheduling for real-time operating systems," Proc. 4th IEEE Real-Time Technology and Application Symposium, 1998.
[23] C. Krishna and Y.H. Lee, "Voltage-clock-scaling techniques for low power in hard real-time systems," Proc. IEEE Real-Time Technology and Applications Symposium, pp.156–165, 2000.
[24] P. Pillai and K. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," Proc. 18th ACM Symposium on Operating System Principles, pp.89–102, Banf, Canada, Oct. 2001.
[25] N.K. Jha, "Low-power system scheduling, synthesis and displays," Proc. IEE Computers and Digital Techniques, vol.152, no.3, pp.344–352, 2005.
[26] R. Buyya and M. Murshed, "Gridsim: A toolkit for the modeling and simulation of distributed management and scheduling for grid computing, concurrency and computation," Practice and Experience, vol.14, no.13, pp.1175–1220, 2002.
[27] A. Sulistio, G. Poduvaly, R. Buyya, and C.K. Tham, "Constructing a grid simulation with differentiated network service using gridsim," Proc. 16th International Conference on Internet Computing (ICOMP'05), 2005.
[28] C. Hsu and U. Kremer, "The design, implementation, and evaluation of a compiler algorithm for CPU energy reduction," Proc. ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation, 2003.

Kyong Hoon Kim received his B.S., M.S., and Ph.D. degrees in Computer Science and Engineering from POSTECH, Korea, in 1998, 2000, 2005, respectively. Since 2007, he has been an assistant professor at the Department of Informatics, Gyeongsang National University, Jinju, Korea. From 2005 to 2007, he was a post-doctoral research fellow at CLOUDS lab in the Department of Computer Science and Software Engineering, the University of Melbourne, Australia. His research interest include real-time systems, Grid and Cloud computing, and security.

Wan Yeon Lee received his B.S., M.S., and Ph.D. in Computer Science and Engineering from POSTECH in 1994, 1996, and 2000, respectively. He is currently an Associate Professor in the Department of Ubiquitous Computing, Hallym University, Chunchon, South Korea. From 2000 to 2003, he was a research engineer in LG Electronics and worked for the standardization of Next Generation Mobile Network of 3GPP Group. From 2006 to 2007, he was a visiting professor in the School of Electrical Engineering, Purdue University, West Lafayette, USA. His research interest includes embedded system, real-time system, mobile computing, computer security, and parallel computing.

Jong Kim received the B.S. in electronic engineering from Hanyang university, Korea, in 1981, the M.S. degree in computer science from the Korean Advanced Institute of Science and Technology, Korea, in 1983, and the Ph.D. degree in computer engineering from Pennsylvania State University in 1991. He is currently a professor in the Department of Computer Science and Engineering, Pohang University of Science and Technology, Pohang, Korea. From 1991 and 1992, he was a research fellow in the Real-Time Computing Laboratory of the Department of Electrical Engineering and Computer Science, University of Michigan. His major areas of interest are security, dependable computing, and distributed computing.

Rajkumar Buyya is a Professor of Computer Science and Software Engineering; and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He received B.E. and M.E. in Computer Science and Engineering from Mysore and Bangalore Universities in 1992 and 1995 respectively; and Doctor of Philosophy (Ph.D.) from Monash University, Melbourne, Australia in 2002. He has authored over 220 publications and three books. The books on emerging topics that Dr. Buyya edited include, High Performance Cluster Computing (Prentice Hall, USA, 1999) and Market-Oriented Grid and Utility Computing (Wiley, 2008). He has pioneered Economic Paradigm for Service-Oriented Grid computing and demonstrated its utility through his contributions to conceptualisation, design and development of Cloud and Grid technologies such as Aneka, Alchemi, Nimrod-G and Gridbus that power the emerging eScience and eBusiness applications.