Power Aware Scheduling of Bag-of-Tasks Applications with Deadline Constraints on DVS-enabled Clusters

Kyong Hoon Kim, Rajkumar Buyya
Grid Computing and Distributed Systems Lab
Dept. of Computer Science & Software Eng.
University of Melbourne, Australia
E-mail: {jysh, raj}@csse.unimelb.edu.au

Jong Kim
Dept. of Computer Science and Engineering
Pohang University of Science and Technology
San 31, Hyoja-dong, Pohang, Korea
E-mail: jkim@postech.ac.kr

Abstract

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. As recent commodity processors support multiple operating points under various supply voltage levels, Dynamic Voltage Scaling (DVS) scheduling algorithms can reduce power consumption by controlling appropriate voltage levels. In this paper, we provide power-aware scheduling algorithms for bag-of-tasks applications with deadline constraints on DVS-enabled cluster systems in order to minimize power consumption as well as to meet the deadlines specified by application users. A bag-of-tasks application should finish all the sub-tasks before the deadline, so that the DVS scheduling scheme should consider the deadline as well. We provide the DVS 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.

1. Introduction

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.

Dynamic Voltage Scaling (DVS) is an efficient way to manage dynamic dissipation during computation [5, 6]. The dynamic power consumption can be reduced by lowering the supply voltage of systems. The DVS scheme reduces dynamic power consumption by adjusting the supply voltage in an appropriate manner. Much recent research [2, 7, 8, 9] has been done to provide power-aware cluster computing by using the DVS scheme.

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 this paper, we deal with power-aware scheduling problem in cluster systems to minimize the QoS degradation in terms of meeting deadlines. We propose DVS scheduling algorithms for bag-of-tasks applications with deadline constraints based on two different resource sharing policies: one for space-sharing and the other for time-sharing.

The rest of this paper is organized as follows. Section 2

describes related work on power-aware cluster systems. In Section 3, the system model is provided, including cluster, energy, and job models. The proposed DVS scheduling algorithms for both space-shared and time-shared approaches are explained in Section 4. Simulation results are given in Section 5, and this paper concludes with Section 6.

2. Related Work

Reducing energy consumption has been one of hot research topics in the area of embedded systems because of the limitation of battery lifetime. Since there is no dedicated energy source in mobile devices, efficient power management is a critical problem in those systems. Recently, research in high performance computing has also introduced and developed power-aware platform to reduce the total energy not only for the operational cost but also for the system reliability. The main goal is to minimize the consumed energy in the system with little degradation of performance.

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 [10, 11] is designed with system-on-chip technology to reduce power in processors and network links. Green Destiny [12] consists of 240 Transmeta processors which consume low power. Orion Multisystem [13] 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 [7], CAFfeine with 16 AMD Opterons [8, 17], and Clusters using Crusoe and Turion [9].

Many recent studies have been conducted to provide power reduction for scientific applications on power-aware cluster systems. Hsu and Feng [8] 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 [7], 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 [9], they provide a profile-based power-performance optimization 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 [7], they present a profile-based optimization in MIPCH. One recent research in [14] presents a transparent MPI run-time system which exploits communication phases in MPI programs to reduce energy. In [15], they reduce energy consumption of parallel sparse matrix applications modeled by MPI.

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 [19, 20, 23]. The basic idea is to slowdown the clock speed using slack time to the task deadline. In this paper, we consider deadline as QoS metric of applications submitted to the cluster systems. Few previous power-aware cluster platform has considered both QoS and energy consumption. Thus, we focus on the problem to reduce energy for applications with deadlines.

3. System Model

3.1. Cluster model

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 in terms of MIPS (Million Instruction Per Second). Thus, a cluster system in this paper is defined as (N, Q), where N is the number of PEs and Q is the processing performance of each PE in terms of MIPS.

3.2. Energy model

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 [6]. 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% [21, 22].

Let us consider that a task of L Million Instructions (MIs) is executed on a processor with V supply voltage and M MIPS performance. The execution time is defined by

L/M seconds. The energy consumption during the task execution is defined by Equation (1) since the number of clock cycles is in proportion to the number of instructions. In Equation (1), α is a proportional constant.

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

We assume that the PE in a cluster system can adjust its supply voltage from V_1 to V_m discretely. The associated processor speed with each supply voltage V_i is denoted as Q_i ($i = 1, \cdots, m$) in terms of MIPS. Without loss of generality, Q_{i+1} is assumed to be larger than Q_i. We also define the normalized speed of each voltage V_i as S_i, which is determined by Q_i/Q_m. Table 1 shows an example of four voltage levels.

Table 1. An example of energy model

Voltage (V_i)	MIPS (Q_i)	Relative Speed (S_i)
0.9 V	4,000	0.4
1.1 V	6,000	0.6
1.3 V	8,000	0.8
1.5 V	10,000	1.0

3.3. Job model

A job in this paper is considered to be a bag-of-tasks application [16], 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 that the job execution must be finished before the deadline.

Thus, a user's job is defined as $(p, \{l_1, l_2, \cdots, l_p\}, d)$, where p is the number of sub-tasks, l_i is the number of instructions of the i-th task in Million Instructions (MIs), 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.

3.4. Problem to solve

In this paper, we consider power-aware scheduling of bag-of-tasks applications with deadline constraints in a DVS-enabled cluster system. Users submit their jobs with deadline constraints as QoS parameters. The cluster system should allocate the resource to jobs for the purpose of meeting their QoS requirements of deadlines.

The cluster system needs to reduce the energy consumption not only for operational cost but also for system reliability. However, there are some trade-offs between reducing energy consumption and meeting deadlines. Running processing elements under low supply voltage decreases the energy consumption but causes jobs to miss deadlines due to low processor speeds. On the contrary, controlling processors under high supply voltage can meet job deadlines, which incurs much energy consumption. Thus, it is required to control supply voltages of PEs in the cluster as low as possible to reduce the energy consumption, under the constraint that all the deadlines of accepted jobs can be met.

This paper deals with the problem of adjusting each PE's supply voltage as well as scheduling jobs in a DVS-enabled cluster system. Since we consider dynamically arriving jobs, the proposed approach focuses on the scheduling of currently available jobs in a best-effort manner to meet the deadlines and reduce the energy consumption.

4. DVS-based Cluster Scheduling

4.1. 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.

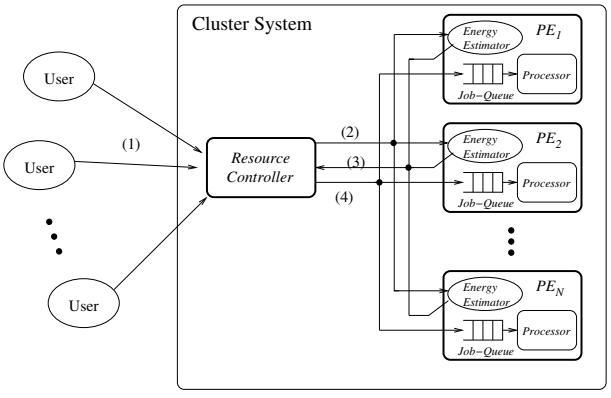

Figure 1. Resource allocation framework

(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* $(proc, l, d)$ 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).

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 Section 4.2 and one time-shared scheduling algorithm in Section 4.3.

```
Algorithm Admission_Resource_Allocation (J)
/* - J = (p, {l_1, ..., l_p}, 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, 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;
```

Figure 2. Application admission and resource allocation

4.2. 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 Figure 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 $e_{k,i}$ and $d_{k,i}$ are the remaining execution time and deadline of the i-th task in each. If the remaining task length is $l_{k,i}$, then the remaining execution time $e_{k,i}$ is defined by $l_{k,i}/Q_m$. And, n_k is the number of tasks in T_k.

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 [18, 19], 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 is greater than or equal to $\tilde{s}_k$. It is followed by Equation (2). When PE_k dispatches the earliest-deadline task in its local queue, it changes the current voltage as v_k.

$$v_k = \min_{i=1}^{m} \{V_i | S_i \geq \tilde{s}_k\} \quad (2)$$

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 under the voltage level in Table 1. At time 0, $u_{k,1}$, $u_{k,2}$, and $u_{k,3}$ are $1/4$, $3/6$, and $5/6$, respectively, so that $\tilde{s}_k$ is 0.5. Since the lowest voltage with speed level more than 0.5 is 1.1 V, v_k at time 0 becomes 1.1 V. After executing $\tau_{k,1}$, v_k

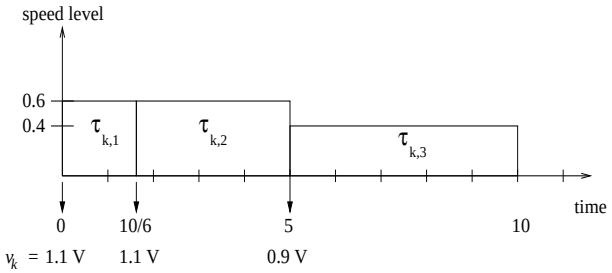

Figure 3. An example of DVS-based EDF scheduling

Algorithm schedulable_EDF (PE_k, l, d)
/* - l : the length of a task
- d : the deadline of a task
*/
$T_{k'} \leftarrow T_k \cup \{(l/Q_m, d)\}$;
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*;

Figure 4. Schedulability test for EDF

Algorithm energy_estimate_EDF (PE_k, l, d)
/* - l : the length of a task
- d : the deadline of a task
*/
$E_{current} \leftarrow$ *energy_consumption* (T_k, n_k);
$T_{k'} \leftarrow T_k \cup \{(l/Q_m, d)\}$;
$E_{new} \leftarrow$ *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** $u_j \leftarrow \frac{\sum_{k=1}^{j} e_k}{d_j}$;
$\tilde{s} \leftarrow \max_{j=i}^{n}\{u_j\}$;
$v \leftarrow \min_{j=1}^{m}\{V_j | S_j \geq \tilde{s}\}$;
$s \leftarrow \min_{j=1}^{m}\{S_j | S_j \geq \tilde{s}\}$;
$Energy \leftarrow Energy + \alpha v^2 e_i Q_m$;
$time \leftarrow time + e_i/s$;
for j **from** i **to** n **do** $d_j \leftarrow d_j - e_i/s$;
endfor
return *Energy*;

Figure 5. Energy estimation for EDF

at time $10/6$ can be obtained similarly. Figure 3 shows the scheduling result of the task set until time 10.

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

4.3. 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 $\sum e_{k,i}/d_{k,i}$. Thus, the speed level under continuous voltage control, $\tilde{s}_k$, is defined by $\sum e_{k,i}/d_{k,i}$. And, the supply voltage level is the lowest voltage of which speed level is larger than or equal to $\sum e_{k,i}/d_{k,i}$, as shown in Equation (2).

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 $share_{k,i}$. If the corresponding speed level of v_k is s_k, each task's share amount is defined by Equation (3). As shown in Equation (3), the utilization of each task is at least $e_{k,i}/d_{k,i}$. For the remaining processor utilization $s_k - \sum e_{k,i}/d_{k,i}$ is distributed according to each task's weight $e_{k,i}/d_{k,i}$.

$$share_{k,i} = \frac{1}{s_k}\left\{\frac{e_{k,i}}{d_{k,i}} + (s_k - \sum_{j=1}^{n_k}\frac{e_{k,j}}{d_{k,j}}) \cdot \frac{e_{k,i}/d_{k,i}}{\sum_{j=1}^{n_k}\frac{e_{k,j}}{d_{k,j}}}\right\} \quad (3)$$

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 Section

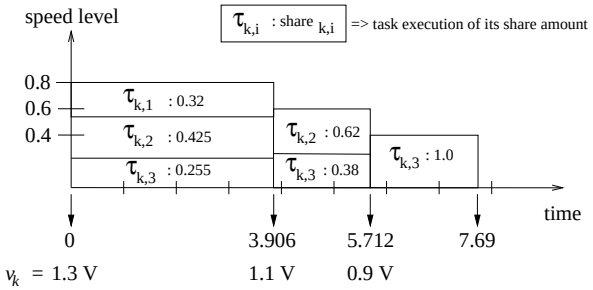

Figure 6. An example of DVS-based proportional share scheduling

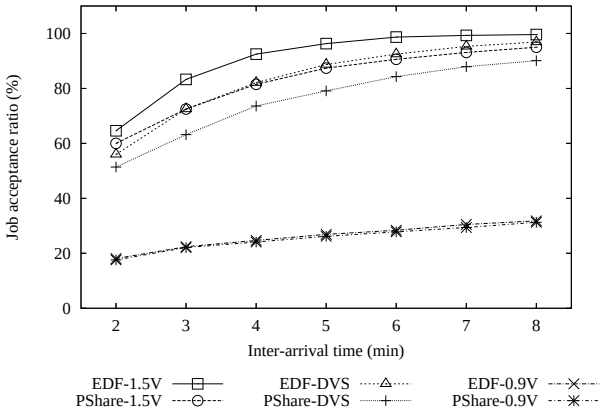

Figure 7. Job acceptance ratio

4.2. Schedulability test and energy estimation for the proportional share scheduling algorithm are similar to those of EDF, as shown in Figure 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.

5. 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 2GHz 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.

Table 2. Operating points of simulated processor

Frequency	Voltage	MIPS	Relative Speed
0.8 GHz	0.9 V	4,000	0.4
1.0 GHz	1.0 V	5,000	0.5
1.2 GHz	1.1 V	6,000	0.6
1.4 GHz	1.2 V	7,000	0.7
1.6 GHz	1.3 V	8,000	0.8
1.8 GHz	1.4 V	9,000	0.9
2.0 GHz	1.5 V	10,000	1.0

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 (=1.5V).

In the simulations, we generate 1000 bag-of-tasks jobs. The number of tasks in a job is randomly selected from 2 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. 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 Figure 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 Figure 8.

Figure 8 shows the average energy consumption per accepted task in the simulations. **EDF-1.5V** and **PShare-1.5V** consume large amount of energy because they fix the supply voltage with 1.5V. 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 Figure 7. The proposed DVS schemes consume less energy compared to 1.5V-static schemes and show similar acceptance ratio.

Table 3 shows performance comparison between DVS and 1.5V-static schemes in terms of success ratio and en-

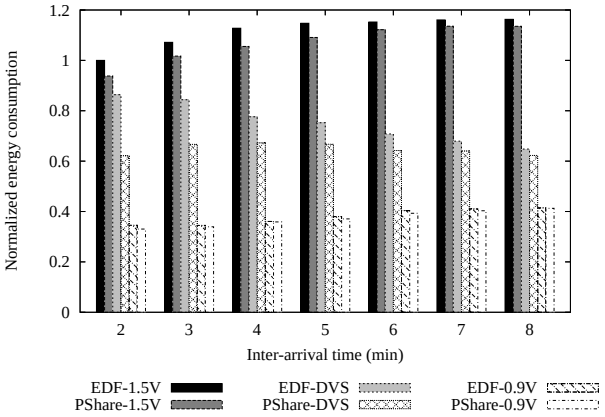

Figure 8. Energy consumption normalized to EDF-1.5V at inter-arrival time of 2 mins.

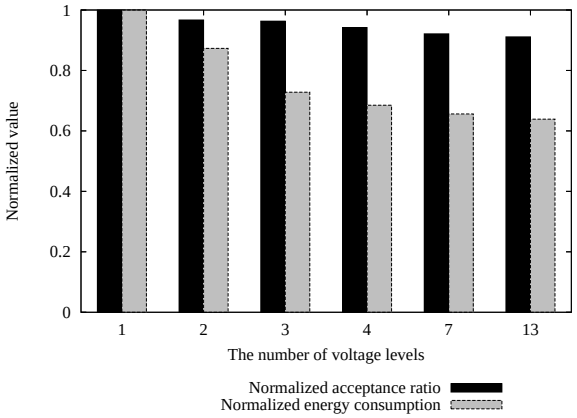

Figure 9. Normalized performance of EDF

ergy 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.

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

Next, we vary the number of supply voltage levels in order to analyze the impact of granularity of controllable voltage levels. The number of voltages is changed from 1 to 13 based on Table 2. The inter-arrival time between two consecutive jobs are generated by a Poisson distribution with a mean of 5 minutes. When the number is one, it is the same as **EDF-1.5V** in Figure 7. Figure 9 and Figure 10 show normalized performance of EDF and proportional share, respectively. Energy consumption of DVS scheme decreases as the number of controllable voltages increases. Fine-grained voltage level can reduce more energy consumption with a little degradation of acceptance ratio.

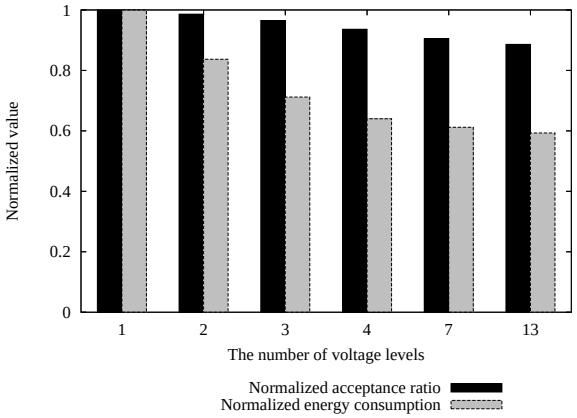

Figure 10. Normalized performance of PShare

6. Conclusions

As recent processors support multiple supply voltage levels, power-aware cluster systems are easily built with commodity processors. Power-aware scheduling of applications on DVS-enabled 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 dead-

line 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 budget-constrained scheduling or workflow scheduling in the Grid, since the energy cost is an important factor in resource cost in the Grid.

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," *Proceedings of the ACM/IEEE SC 2005*, Seattle, USA, November 2005.

[3] J. Markoff and S. Lohr, Intel's huge bet turns iffy. New York Times Technology Section, September 29, 2002. Section 3, Page 1, Column 2.

[4] W. Feng, "Making a Case for Efficient Supercomputing," *ACM Queue*, vol. 1, no. 7, pp. 54-64, 2003.

[5] I. Hong, D. Kirovski, G. Qu, M. Potkonjak, and M. B. Srivastava, "Power Optimization of Variable-Voltage Core-based Systems," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 12, pp. 1702-1714, 1999.

[6] T. D. Burd and R. W. Brodersen, "Energy Efficient CMOS Microprocessor Design," *Proceedings of the 28th Annual Hawaii International Conference on System Sciences*, pp. 288-297, January 1995.

[7] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained Distributed DVS Scheduling for Scientific Applications on Power-aware Clusters," *Proceedings of the ACM/IEEE SC 2005*, Seattle, USA, November 2005.

[8] C. Hsu and W. Feng, "A Power-Aware Run-Time System for High-Performance Computing," *Proceedings of the ACM/IEEE SC 2005*, Seattle, USA, November 2005.

[9] 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," *Proceedings of 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Rhodes Island, Greece, April 2006.

[10] IBM Research Blue Gene Project. http://www.research.ibm.com/bluegene.

[11] N. D. Adiga, et al., "An Overview of the BlueGene/L Supercomputer," *Proceedings of the 2002 ACM/IEEE Conference on Supercomputing*, Baltimore, USA, November 2002.

[12] W. Warren, E. Weigle, and W. Feng, "High-density Computing: A 240-node Beowulf in one cubic meter," *Proceedings of the 2002 ACM/IEEE Conference on Supercomputing*, Baltimore, USA, November 2002.

[13] Orion Multisystems. http://www.orionmult.com/.

[14] M. Y. Lim, V. W. Freeh, and D. K. Lowenthal, "Adaptive, Transparent Frequency and Voltage Scaling of Communication Phases in MPI Programs," *SC'06*, November 2006.

[15] 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," *Proceedings of 20th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Rhodes Island, Greece, April 2006.

[16] W. Cirne, F. Brasileiro, J. Sauve, N. Andrade, D. Paranhos, E. Santos-Neto, and R. Medeiros, "Grid Computing for Bag of Tasks Applications," *Proceedings of the 3rd IFIP Conference on E-Commerce, E-Business and E-Government*, September 2003.

[17] C. Hsu and W. Feng, "A Feasibility Analysis of Power Awareness in Commodity-Based High-Performance Clusters," *Cluster 2005*, Boston, USA, September 2005.

[18] T. Pering and R. Brodersen, "Energy Efficient Voltage Scheduling for Real-Time Operating Systems," *Proceedings of 4th IEEE Real-Time Technology and Application Symposium*, Denver, USA, June 1998.

[19] C.M. Krishna and Y. H. Lee, "Voltage-Clock-Scaling Techniques for Low Power in Hard Real-time Systems," *Proceedings of IEEE Real-Time Technology and Applications Symposium*, pp. 156-165, Washington, DC, USA, May 2000.

[20] P. Pillai and K. Shin, "Real-time Dynamic Voltage Scaling for Low-power Embedded Operating Systems," *Proceedings of 18th ACM Symposium on Operating System Principles*, pp. 89-102, Banf, Canada, October 2001.

[21] J. Li and J. F. Martínez, "Dynamic Power-Performance Adaptation of Parallel Computation on Chip Multiprocessors," *Proceedings of 12th International Symposium on High-Performance Computer Architecture*, Austin, USA, February 2006.

[22] C. Piguet, C. Schuster, and J.-L. Nagel, "Optimizing Architecture Activity and Logic Depth for Static and Dynamic Power Reduction," *Proceedings of the 2nd Annual IEEE Northeast Workshop on Circuits and Systems*, pp. 41-44, Montreal, Canada, June 2004.

[23] N. K. Jha, "Low-Power System Scheduling, Synthesis and Displays," *IEE Proceedings on Computers and Digital Techniques*, vol. 152, no. 3, pp. 344-352, 2005.

[24] F. Worm, P. Ienne, P. Thiran, and G. D. Micheli, "An Adaptive Low-Power Transmission Scheme for On-chip Networks," *Proceedings of the 15th International Symposium on System Synthesis*, pp. 92-100, 2002.

[25] L. Shang, L. S. Peh, and N. K. Jha, "Dynamic Voltage Scaling with Links for Power Optimization of Interconnection Networks," *Proceedings of the 9th International Symposium on High-Performance Computer Architecture*, pp. 91-102, 2003.

[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. 13-15, 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," *Proceedings of the 16th International Conference on Internet Computing (ICOMP'05)*, Las Vegas, USA, June 2005.