Power Aware Job Scheduling with QoS Guarantees Based on Feedback Control

Congfeng Jiang
Grid and Service Computing Technology Lab, Hangzhou Dianzi University, Hangzhou, 310037, China
Email:cjiang@hdu.edu.cn

Xianghua Xu, Jian Wan,Jilin Zhang
Grid and Service Computing Technology Lab, Hangzhou Dianzi University, Hangzhou, 310037, China
Email:{xhxu,wanjian,zhangjilin}@hdu.edu.cn

Yinghui Zhao
Department of Hydraulic Engineering and Agriculture, Zhejiang Tongji Vocational College of Science and Technology, Hangzhou, 311231, China
Email: zhaoyinghuihust@gmail.com

***Abstract*—With the scale of computing system increases, system performance and reliability, described by various Quality of Service(QoS) metrics, cannot be guaranteed if only the objective is to minimize the total power consumptions separately, despite of the violations of QoS. In this paper a feedback control based power aware job scheduling algorithm is proposed to minimize power consumption in computing system and to provide QoS guarantees. In the proposed algorithm, jobs are scheduled according to the real-time and historical power consumption as well as the QoS requirements. Simulations show that the proposed algorithm can reduce power consumptions significantly while still providing QoS guarantees and the performance degradation is acceptable. The results also show that fine-grained job-level power aware scheduling can achieve better power/performance balancing between multiple processors or cores than coarse-grained methods. And the results also suggest that conventional hardware based per-component and system-wide power management methods can save more power consumptions if they are in assistance with job-level adaptation.**

***Index Terms*—power aware computing system; job scheduling; Quality of Service (QoS); feedback control**

I. INTRODUCTION

Power consumption has been a major concern for not only large scale computing system but also mobile and embedded systems powered by batteries. With the scale of computing system increases, power consumption has become the major challenge to system performance and reliability [1, 2, and 3]. For example, server farms today consume more than 1.5% of the total electricity in the U.S. at a cost of nearly $4.5 billion, which is estimated to about 61 billion kilowatt-hours (kWh) in 2006 and it is more than the electricity consumed by the nation's color televisions and similar to the amount of electricity consumed by approximately 5.8 million average U.S. households [4]. According to current efficiency trends, power consumption by servers and data centers could nearly double again in the next five years.

Current processors are provided with support for dynamic frequency and voltage scaling (DFS/DVS) to allow software to regulate power consumption by varying operating frequency and/or supply voltage. However, with DFS/DVS support, processors are simply switched to a sleep mode while transitioning between frequencies and voltages and the system performance will be heavily deteriorated by the transition delay between various frequency and voltage levels in uniprocessors. This situation becomes even worse in multi core processor which does not support per-core DFS/DVS [5]. Conventional hardware based per-component and system-wide power management methods cannot save considerable power consumptions because they are coarse-grained and not adaptive to various fluctuating workloads in real scenarios. Moreover, the system performances, for example, availability, responsiveness, and throughput, do not scale with the number of processors but the power consumption does. Most unfortunately, the performance of the whole system can be deteriorated greatly if the objective is only to minimize the total power consumptions separately, despite of the violations of Quality of Service (QoS) requirements.

Powering servers or computing components on and off is frequently used for tacking transactional workloads which consist of independent requests and short-lived jobs. However, in modern data centers with service consolidation and virtualization deployment, there are coarse-grained and heterogeneous workloads whose performance requirements are specified in terms of QoS for each upper application in the computing environment. Server virtualization consolidates multiple under-utilized servers into a single physical server to exhaust physical machines. Therefore, it is important to reduce power

Corresponding author: Congfeng Jiang,email:cjiang@hdu.edu.cn

doi:10.4304/jsw.6.8.1562-1569

consumption in virtualization environments because such advantage of virtualization also leads itself to the original power consumption problem because there usually are high power densities in virtualization environments.

In this paper, we investigate the feasibility of hardware-software joint regulation for a feedback control based power aware job scheduling algorithm and test the algorithm by simulations and real workload. In the proposed algorithm, the feedback based power-aware job scheduler regulates the job dispatching and system performance dynamically for different workload characteristics. A testbed for investigating this scheme is implemented on a web server using Intel quad core processor. The performance and overhead of the algorithm are assessed under different workload. The results show the potential of power consumption reductions for hardware-software joint adaptations. We use a multimeter with USB connection cable to measure the real time power consumption of the system and the measurement results show that fine-grained job-level power aware scheduling can reduce considerable power consumptions. The results also show that the best algorithm varies for different experiment settings and real workload scenarios. In particular, feedback control based power aware job scheduling algorithm is not always better than other algorithms in all the performance dimensions. The performance depends on many factors such as the accuracy of workload characterization, the arrival rate of jobs, frequency-voltage transition delays in specific processors, the frequency/voltage levels available, etc.

The remainder of this paper is organized as follows: In section 2 we propose the controlling framework of the power aware job scheduling scheme. In section 3 we present the feedback control based power aware job scheduling algorithm model with QoS guarantees. Then, in Section 4, we present simulation results and real platform experiment results of the proposed scheduling algorithm. We also compare the performance data with conventional power-unaware job scheduling algorithms or job scheduling algorithms without QoS guarantees. Finally, we summarize the work in Section 5..

II. FEEDBACK CONTROL MODEL

There has been increasing research effort in applying control-theoretic approaches to power and performance management for computer systems such as internet web servers, databases and storage systems. Since today's large scale servers and applications are highly dynamical and change load conditions frequently, feedback control designs may provide desired performance guarantees.

It is commonly believed that system performance can be maximized by operating servers at their highest power levels under a given power budget. The main idea of this paper is the intuition that in lower loaded periods, there is a potential to save power consumption by dynamically powering off part of or whole servers to address the actual computing demands. Under such lower-load conditions, an appropriate fine-grained job scheduling scheme can considerably reduce power consumption. In the meantime, under higher load condition, power aware scheduling can also schedule jobs properly to balance power consumption between various processors and avoid hotspots. Therefore, the proposed power aware job scheduling in this paper contains three parts: workload characterization and prediction, power consumption measuring and estimation, feedback control of power consumption through job scheduling with QoS constraints.

A. Controlling Framework

In dynamic computing system, different tasks demonstrate variant execution time behavior. In this paper we use feedback control to capture the dynamic workload behavior, which is one of the fundamental mechanisms for dynamic systems to achieve equilibrium. In order to precisely control the power of a system to the desired preset point while guaranteeing the QoS requirements of each job, an online model estimator should be integrated in the scheduler to achieve analytical assurance of control accuracy and system stability, even in presence of significant workload variations or unpredictable job variations.

In a feedback system, some variables, i.e., controlled variables are monitored and measured by the feedback controller and compared to their desired values, i.e., the preset points. The differences or errors between the controlled variables and the preset points are the input of the controlling system. At the meantime, the corresponding system states are adjusted according to the differences to let the system variables approximate the preset points as closely as possible. In order to assess the suitability and energy saving performance of the proposed algorithm, we regard the entire system as consisting of the following components: inputs, actuator, control object, outputs, sensors, etc. These components are independent of each other such that the scheduler is capable of working with different algorithms.

In our feedback scheme, the scheduler chooses the preset threshold as controlling input according to the feedback information collected from the previous task executions and performance data such as QoS satisfactions, power consumptions, etc. As long as the actual real-time performance is less than or equal to the preset threshold, the tasks can therefore be executed at a lower frequency and voltage level. The error is measured periodically by the sensor unit. Its output is fed back to the scheduler to adjust the value for inputs. If a task's actual performance is worse than the preset threshold, the rest of the task runs at the higher frequency or voltage to meet the QoS requirements of the tasks. In order to provide performance guarantees and minimize the power consumption, the algorithm keeps the total system utilization not lower than a specific level when reducing processor frequency and voltage. In the following, we propose a feedback schemes for different computing scenarios. In the following, we describe in detail of the feedback scheduler used in the framework.

B. Power-aware and feedback mechanism (PFM)

In the power aware feedback mechanism, at the beginning of a job scheduling round, the scheduler chooses the average value of previous performance data

as the controlled variable for our simple feedback mechanism. The average value is only the initial value for the future scheduling and it is a heuristic setting which can provide a minimum guarantee for QoS satisfactions. In the following scheduling period, each time when a job completes, the real time performance is sensed and collected by the feedback scheme and fed to the controller. And the following performance values are computed based on the initial value of performance threshold using moving average computation.

Here are the pseudo codes of the power aware feedback mechanism.

1. When scheduling event occurs {
2. for each task in the task set
3. Compute estimated performance data of each task and its execution time
4. End for
5. Compute estimated power consumption of each job on specific processor through code profiling
6. Compute QoS gains of each job
7. for each task in the task set
8. Schedule the tasks with minimum QoS gains and power consumptions
9. Delete the task from the task set
10. Update job table
11. If cpu_queue of targeted processor is exceeded the maximum length
12. Insert the task into next scheduling tasks set
13. Update job table
14. End if
15. Else
16. Insert the task into next scheduling tasks set
17. Update job table
18. End if
19. End for
20. }

Figure 1. The pseudo codes of PFM

III. POWER AWARE JOB SCHEDULING WITH QOS GUARANTEES

We use a periodic and independent task model in our framework. Let $J = \{J_i \mid i = 1,2,3,...,n\}$ denote jobs set, $J_i = (a_i, b_i, e_i, c_i, s_i, Q_i)$, M is set of processors, $M = \{M_j \mid j = 1,2,3,...,m\}$, $M_j = (p_j, f_j, d_j, BW_j, PW_j)$.

Where:

a_i is arrival time of job J_i, b_i is starting time of job J_i, e_i is the average execution time of job J_i on all the processors, i.e. expected executed time on processor M_j where there is no other running jobs except for J_i, $e_i = (e_i^1, e_i^2, e_i^3, ..., e_i^m)$, c_{ij} is the expected completion time of job J_i on processor M_j, s_i is the size of data needed by job J_i (MB), Q_i is the QoS of job J_i.

As for hosts set M, p_j is the speed of processor M_j (MHz), f_j is the available memory capacity of host M_j (MB), d_j is the available disk space on host M_j (MB), BW_j is the bandwidth of host M_j (Mb/s), PW_j is the level of power consumption of host M_j, $PW_j = (PW_j^1, PW_j^2, PW_j^3, ..., PW_j^u)$, $\forall v \in [1,u], 0 < PW_j^v < 1$, where 0 stands for the lowest and 1 the highest.

Let Q denote a set of Quality of Service constraints, $Q = \{Q_i \mid i = 1,2,...,n\}$, $Q_i = (T_i, R_i, S_i, A_i, P_i)$, where:

T_i is timeliness requirement, R_i is reliability requirement, S_i is security requirement, A_i is accuracy requirement, P_i (Priority) is priority requirement.

For simplicity, we use discrete values to modeling the Quality of Service constraints, i.e., the QoS constraint is presented by several levels like very low, low, medium, high, and very high, not a specific number like 10% or 90% because in real computing system with user interaction a user only cares the interactive experience, not the specific performance numbers.

Let $ec^J = (ec_1, ec_2, ..., ec_m)$ denote the power consumption of m threads, and the matrix of n performance counters in m threads is $C = [c_{i,j}](1 \le i \le m, 1 \le j \le n)$.

We define G_i^j is the gains of QoS of job J_i on host M_j, i.e.

$$G_i^j = \sum_{k=1}^{q} w_i^k \cdot g(Q_i^k, V_j^k) \quad \text{(Eq.1)}$$

Where w_i^k is the weights of different QoS requirements of job J_i, and $\sum_{k=1}^{q} w_i^k = 1$; $g(Q_i^k, V_j^k)$ is the kth gains of QoS requirements of job J_i:

$$g(Q_i^k, V_j^k) = \begin{cases} Q_j^k - V_i^k & , when\ V_j^k > Q_i^k \\ 0 & , when\ V_j^k < Q_i^k \end{cases} \quad \text{(Eq.2)}$$

Where V_j^k is the available QoS capacity of the corresponding host.

We define D_i as the available theoretical scheduling set of job J_i with QoS satisfactions, $\exists D_i = (D_i^j \mid G_i^j > 0)$

In order to avoid the QoS contention, the gains of QoS satisfactions must be minimized while still guarantying the QoS requirements. Assume that

$$OP_i = (D_i^j \mid G_i^j > 0) \cap \min\left\{\sum_{k=1}^{q} w_i^k \cdot g(Q_i^k, V_j^k)\right\} \quad \text{(Eq.3)}$$

Then the objective function for power and QoS constrained scheduling for job set J_i is

$$S = \min\left\{\sum_{J_i \in J} OP_i\right\} \cap \min(\max\{_{T_i \in T}(c_i)\}) \quad \text{(Eq.4)}$$

Eq.4 is NP-hard and can be solved by heuristics scheduling. With the above task model, we use heuristics scheduling algorithm to solve this problem of power

aware job scheduling with QoS constraints. The algorithm is triggered by a scheduling event. When the number of jobs in the job set becomes a fixed maximum number, like 5, we call this a scheduling event. A job is submitted to the primary scheduler and the backup scheduler respectively. Thus for different objectives higher power savings can be achieved when higher missed deadlines or QoS violations are allowed. The time-varying workload parameters, such as workload intensity, real-time power consumed, are specified as the scheduling variable that is used to parameterize the scheduling model. The utilization of scheduling parameters can be generalized to accommodate more sophisticated workload characterizations and more complicated multiple server environments.

IV. SIMULATION AND EXPERIMENT RESULTS AND PERFORMANCE ANALYSIS

A. Simulation Setup and Parameters Settings

Feedback control based scheduling algorithm is capable of modifying its own scheduling decision and program behavior through time, depending on the execution characteristics. Here, the objective of the feedback control is to schedule jobs to processors while guarantying the QoS requirements and minimizing the total power consumptions of the computing system, preserving its simplicity and low overhead. We test this algorithm and describe its behavior under a number of workloads. Simulations include an analysis of the performance sensibility with the variation of the control parameters and its application in a multi processor computing system. Although the processors of a Massively Parallel Processing system such as a computing cluster or a supercomputer may slightly differ in clock frequency and available memory, these differences usually have little influence on most applications in practice [6]. Hence, in the simulation we use an MPP with multiple machines as the testbed and it is feasible for job migration.

In our simulations, we construct synthetic workloads using jobs with varying arrival rate and execution time. Moreover, in order to evaluate the robustness of our algorithm, we allow the jobs to follow different distributions, such as Gaussian, Possion, Uniform, Weibull, and heavy tailed distribution. In the simulation process, different data sets are used and the data sets represent different orders of magnitude in server activity and time duration. We generate a wide range of workloads by varying the number of jobs and their execution times in our simulations. Specifically, we conduct over 7 sets of experiments, and the number of jobs in each experiment is selected according to a specific distribution. The relative performance of an algorithm in each experiment is compared by normalizing its original values. Table 1 lists the key simulation parameters of one simulation.

TABLE I. REPRESENTATIVE SIMULATION PARAMETERS AND SETTINGS

Number of jobs	100,000,000
Number of processors	16
Site processing speed	8 nodes with 2.4GHz and 8 nodes with 1.8GHz
Job arrival rate	Poisson distributed in [0.2, 0.9]
Job execution time	Normal distributed in [0.1, 10000]

B. Simulation Results and Analysis

We use simulations and real workload experiments to study the performance of the proposed scheduling algorithm. Extensive sets of tests are executed to simulate the performance of our feedback scheduling power-aware framework under overload and under loaded conditions. In simulations, we use various matrix and vectors to modeling the controlling system, including task sets (i.e., arrival rate, task's period, deadline, actual execution time, worst case execution time, estimated execution time, etc), QoS requirements, power consumptions, and other system parameters. In a simulation setting, all tasks being scheduled are assumed to be periodic and each task's actual execution time is assumed to be known in advance and is indicated in matrix declared in the same .m program. Each task indicates its QoS needs quantitively to the scheduler, which in turn is able to know the global QoS requirements of the system to meet all task requirements. At the initial round when scheduling events occur (the threshold is reached and the scheduling is triggered), the scheduler make the task scheduling decision and the processor voltage/frequency scaling. After several scheduling periods, the scheduler makes the scheduling decision and adjusts the processor frequency and voltage level according to the feedback information collected from the sensor units.

The common approach to study the performance of the scheduling algorithm is to compare it with a non-power-aware or non-QoS-aware scheduling algorithm. Thus we studied and compared the performance of the simple and frequently used heuristics such as EDF, Min-min [7], Max-min [7], QoS guided Min-Min [8], Sufferage [9], and MCT (Minimum Completion Time [10], with PFM by testing various scenarios.

To evaluate the scheduling algorithm, we use the following metrics:

1) Makespan: the total running time of all jobs;

2) Average waiting time: the average waiting time spent by a job in the grid.

3) Scheduling success rate: the percentage of jobs successfully completed in the system;

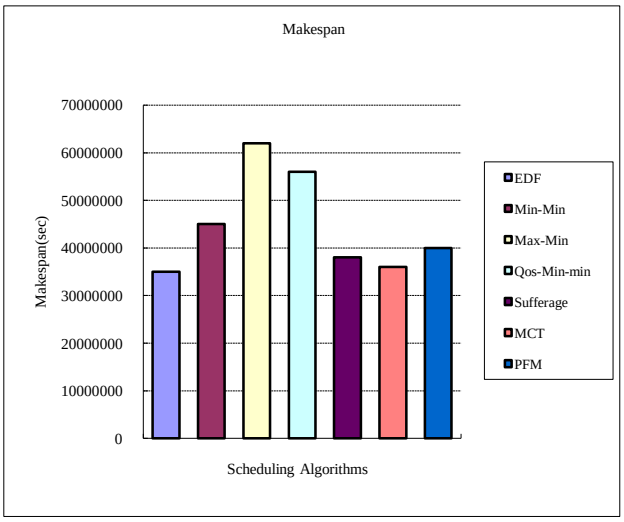
4) Power consumption: the power consumed by the jobs.

5) Average violating rate of QoS: the percentage of QoS violation when scheduling user jobs out of total jobs.

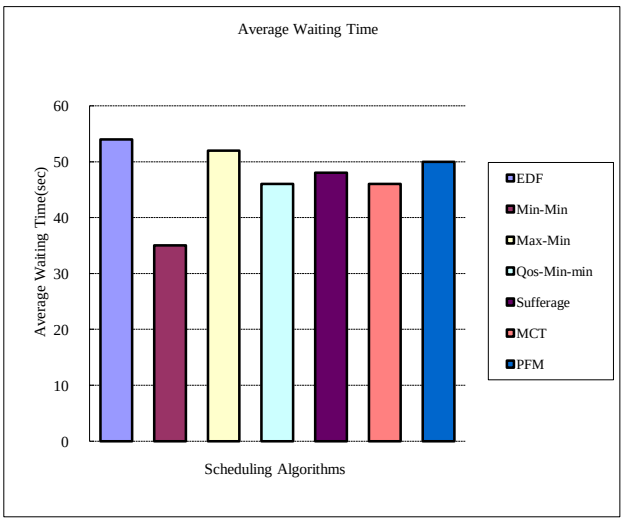
6) Average migration rate: the percentage of migrated jobs out of total scheduled jobs.

7) Overall utilization: the percentage of resources used out of total available resources.

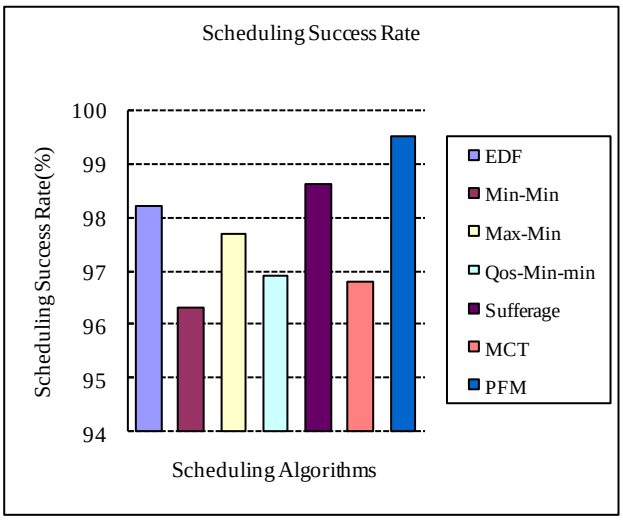
The simulation results are shown in Figure 2. All the data in the figures are mean values of 20 simulation results.



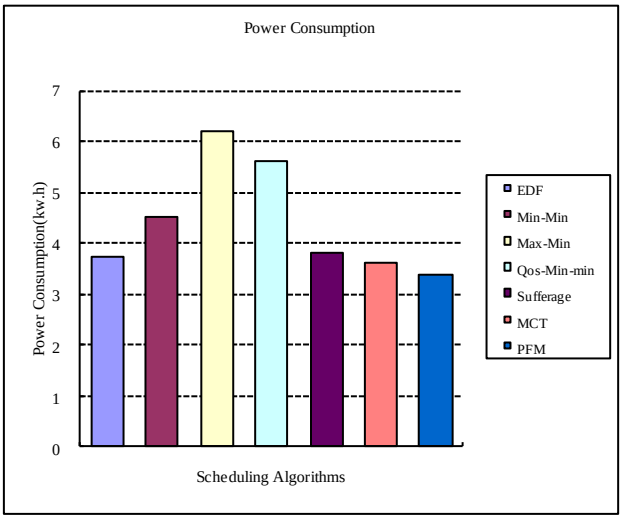
(a) Makespan



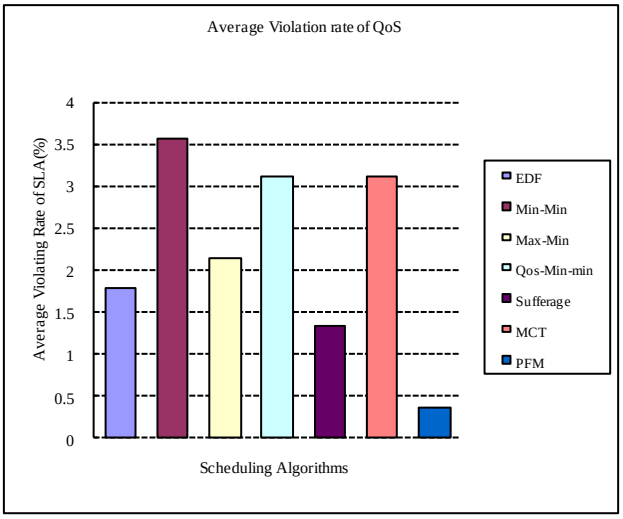
(b) Average waiting time



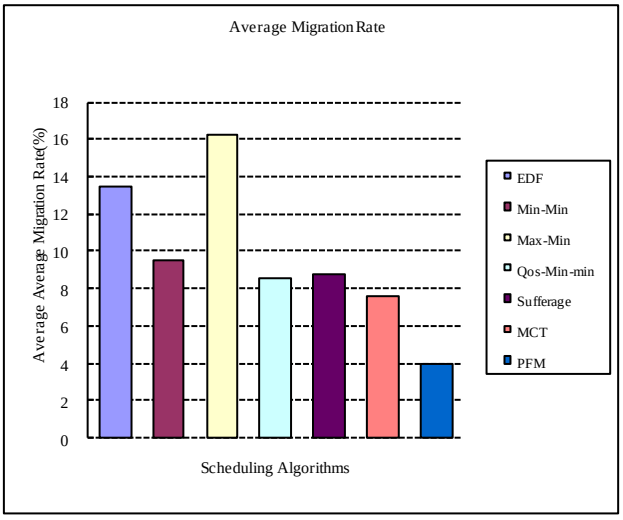
(c) Scheduling success rate



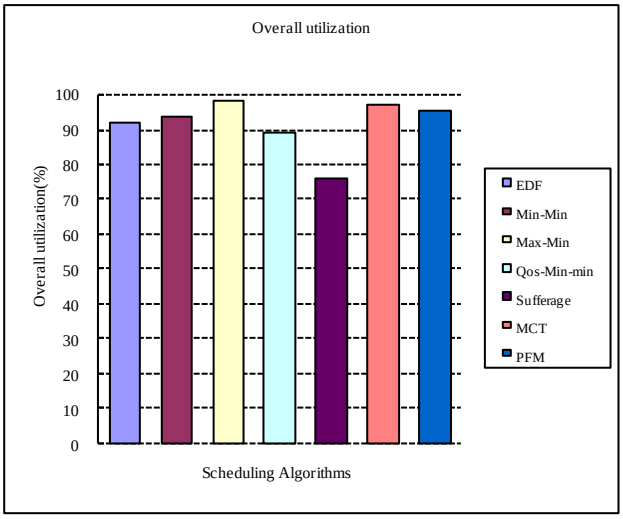
(d) Power consumption



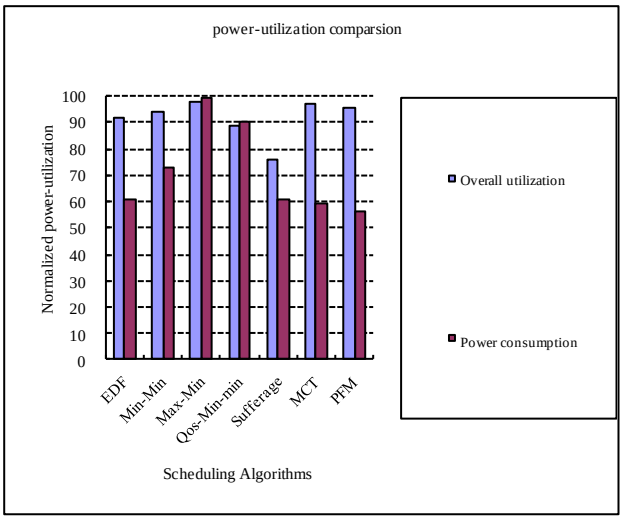
(e) Average violation of QoS



(f) Average migration rate



(g))Overall utilization



(h) Power-utilization comparison

Figure 2. Relative performance

In Figure 2(a), the makespan order of the scheduling algorithms from maximum to minimum is: (1) Max-Min, (2) QoS-Min-min, (3) Min-Min, (4) PFM, (5) Sufferage, (6) MCT, and (7) EDF. The makespan of EDF is the smallest because of its smallest computation consumption. PFM dynamically schedules jobs to computing sites according to the real time power consumption and QoS constraints. Thus the makespan of PFM is relatively large.

In Figure 2(b), the average waiting time order of the scheduling algorithms from maximum to minimum is: (1) EDF, (2) Max-Min, (3) PFM, (4) Sufferage, (5) QoS-Min-min and MCT, and (6) Min-Min. EDF has the longest average waiting time because it executes tasks without global information such as waiting times of other tasks. Consequently, EDF makes a significant increase of total execution time and makes the average waiting time longest eventually.

In our simulations, a task will be dropped if it couldn't be finished successfully after ten times. Thus, the scheduling success rate can't reach to 100%.

In Figure2(c), PFM has the highest scheduling success rate in a failure-prone multi-processor environment. PFM reschedules the tasks whose demand couldn't be satisfied on the current time when next scheduling event occurs. Thus, PFM increases the scheduling success rate significantly.

In Figure 2(d), the power consumption order of the scheduling algorithms from highest to lowest is: (1) Max-Min, (2) Qos-Min-min,(3)Min-Min,(4)Sufferage, (5)EDF, (6)MCT, and (7)PFM. PFM has the lowest power consumption because it takes into account the real time power consumption when scheduling tasks.

Since PFM also optimizes the QoS requirements and satisfactions while scheduling tasks, it has the lowest average violation rate of QoS and the lowest average migration rate through Figure 2(e) and Figure 2(f).

We also compare the overall utilization and power-utilization in Figure 2(g) and Figure 2(h). The results show that the power consumption is highly correlated with the utilizations.

The results in Figure 2 show that no single algorithm achieves the highest performance for all metrics. However, PFM exhibits relatively better performance with highest success rate, moderate level of makespan and average waiting time, lowest power consumption, average violation rate of QoS and the lowest average migration rate due to its power aware and QoS aware scheme. The simulation results show that substantial performance improvements can be obtained. As the experimental results finally turn out, the performances of our algorithm is fairly insensitive to the distributions we choose.

V. CONCLUSIONS

Recent processor support for dynamic frequency and voltage scaling makes it possible for power consumption regulation in software level. However, DVS and DFS are not enough for processor power reductions if they are issued only by entering a sleep mode. Therefore, it is feasible to harness job scheduling for power management in computer system. Moreover, fine-grained job-level power aware scheduling can achieve application specific performance QoS guarantees than system wide or per-component power management.

In this paper we propose a job scheduling algorithm based on feedback control theory and characterize workload through data gathered from trace data to capture possible application behavior. The job scheduling model is independent of implementation platforms and therefore feasible for future applications on multi-processor systems. By exploring the nature of dependence of server performance on time-varying load and operating conditions, the proposed general framework is possibly applicable to a diverse spectrum of server-based applications.

Performance such as Makespan, Average waiting time, Scheduling success rate, Power consumption, Average violating rate of QoS, Average migration rate are assessed for different job scheduling algorithms. Measurements provides a quantitative assessment of the

potential of energy savings for power and QoS aware job scheduling algorithms as opposed to conventional job scheduling algorithms that disregard power consumption and QoS constraints. Simulation and experiment results show that the proposed algorithm can save significant power consumptions while still providing QoS guarantees and the performance degradation is acceptable. The results also show that fine-grained job-level power aware scheduling can achieve better power/performance balancing between multiple processors than coarse grained methods.

Moreover, fundamental to the goal of power savings and performance maximization is an understanding of the dedicated workloads. Since global power consumption mode is different in different systems with specific performance-oriented applications and it is also different for different platforms with different performance constraints and QoS requirements, estimating power consumption is critical for job scheduling and obtaining processor and system power consumption is non-trivial and using real platform simulators is time consuming and prone to error. Therefore, it is still an open problem to reduce energy consumption, while still meeting performance demands, system loads and reliability.

Although there are also tradeoffs such as latencies, performance degradations, conflicts and coordination, between the power reduction and performance of the specific application, it's worthy of implementing a bin-level power management schemes to reduce power consumption as much as possible. With the ongoing trend of server consolidation and emerging hardware, the power management problem is still active and researchers and engineers are still trying to address this issue through various efforts. It is highly likely that combination of hardware level and software level power management can overcome this power management problem and provide better performance for large scale computing systems.

In the future, several extensions can be made further under different dimensions. For example, the scheduled tasks will be extended to be real tasks whose actual execution time is not known until its completion. And the scheduler can be implemented as a kernel module in an OS to provide power aware job scheduling.

ACKNOWLEDGMENT

The funding support of this work by State Key Development Program of Basic Research of China (Grant No. 2007CB310900), Natural Science Fund of China (NSFC, Grant No. 60873023 and 60973029), Science and Technology Research and Development Program of Zhejiang Province, China (Grant No. 2009C31033),and Hangzhou Dianzi University Startup Fund (Grant No. KYS 055608058 and KYS 055608033,) are greatly appreciated.

REFERENCES

[1] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T.W. Keller, "Energy management for commercial servers,"Computer, Vol.36, pp.39-48,2003.

[2] X. Wang, M. Chen, C. Lefurgy,and T.W. Keller, "SHIP: Scalable hierarchical power control for large-scale data centers,"in Proceedings of 2009 18th International Conference on Parallel Architectures and Compilation Techniques(PACT'09),pp.91-100,2009.

[3] X. Wang,and M. Chen , "Cluster-level feedback power control for performance optimization," in Proceedings of IEEE 14th International Symposium on High Performance Computer Architecture(HPCA 2008), pp.101-110, Feb. 2008.

[4] Report to Congress on Server and Data Center Energy Efficiency. U.S. Environmental Protection Agency, ENERGY STAR Program, August 2, 2007.Available at:http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf

[5] Y. Wang,K. Ma, and X. Wang, "Temperature-constrained power control for chip multiprocessors with online model estimation," in Proceedings of the 36th annual international symposium on Computer architecture (ISCA'09),pp.314-324,June 2009.

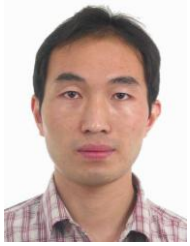
[6] C. Franke, J. Lepping, and U. Schwiegelshohn, "Greedy scheduling with custom-made objectives,"Annals of Operations Research,in press.

[7] T.D. Braun, H.J. Siegel, N. Beck,L. L. Bölöni,M. Maheswaran,A.I. Reuther,J.P. Robertson, et al., "A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems, " Journal of Parallel and Distributed Computing, Vol. 61,pp.810-837,2001.

[8] X. He, X. Sun, and G.V. Laszewski, "QoS guided Min-Min heuristic for grid task scheduling, "Journal of Computer Science and Technology, Vol.18,pp. 442-451,2003.

[9] M. Maheswaran, S. Ali, H.J. Siegel, D. Hensgen, and R.F. Freund, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems," Journal of Parallel and Distributed Computing, Vol. 59, pp.107-131,1999.

[10] L.D. Briceño , M. Oltikar, H.J. Siegel, and A.A. Maciejewski, (). "Study of an iterative technique to minimize completion times of non-makespan machines," in Proceedings of 2007 IEEE International Parallel and Distributed Processing Symposium (IPDPS 2007),pp.1-14,2007



Congfeng Jiang is a lecturer of School of Computer Science and Technology, Hangzhou Dianzi University, China. He is with the Grid and Service Computing Lab in Hangzhou Dianzi University.

Before joining Hangzhou Dianzi University, he was a PhD candidate in Huazhong University of Science and Technology from 2002 to 2007. He received his PhD degree in 2007. His current research areas include power aware computing system, virtualization, grid computing, etc.

Dr. Jiang is a member of China Computer Federation (CCF), IEEE and IEEE Computer Society.

Xianghua Xu is an associate professor at Hangzhou Dianzi University and the vice director of Grid and Service Computing Lab in Hangzhou Dianzi University.

Dr Xu received his PhD degree in 2005 from Zhejiang University, China. His research areas include service computing, parallel and distributed computing system, virtualization, grid computing, etc.

Dr. Xu is a member of China Computer Federation (CCF) and the IEEE.

Jian Wan is the director of Grid and Service Computing Lab in Hangzhou Dianzi University and he is the dean of School of Computer Science and Technology, Hangzhou Dianzi University, China.

Prof. Wan received his PhD degree in 1996 from Zhejiang University, China. His research areas include parallel and distributed computing system, virtualization, grid computing, etc.

Prof. Wan is a member of China Computer Federation (CCF).

Jilin Zhang is a lecturer of School of Computer Science and Technology, Hangzhou Dianzi University, China. He is with the Grid and Service Computing Lab in Hangzhou Dianzi University.

Before joining Hangzhou Dianzi University, he was a PhD candidate in Beijing University of Technology from 2005 to 2009. He received his PhD degree in 2009. His current research areas include parallel processing, complier optimization, virtualization, distributed computing, etc.

Dr. Zhang is a member of China Computer Federation (CCF) and the IEEE.

Yinghui Zhao is a lecturer of Department of Hydraulic Engineering and Agriculture, Zhejiang Tongji Vocational College of Science and Technology, Hangzhou, China.

Before joining Zhejiang Tongji Vocational College of Science and Technology, she was an engineer in China Southern Power Grid Company. She received her Master degree from Huazhong University of Science and Technology in 2007. Her current research areas include remote sensing images processing, geographical information system (GIS), etc.