SHIP: A Scalable Hierarchical Power Control Architecture for Large-Scale Data Centers

Xiaorui Wang, *Member*, *IEEE*, Ming Chen, Charles Lefurgy, *Member*, *IEEE*, and Tom W. Keller

Abstract—In today's data centers, precisely controlling server power consumption is an essential way to avoid system failures caused by power capacity overload or overheating due to increasingly high server density. While various power control strategies have been recently proposed, existing solutions are not scalable to control the power consumption of an entire large-scale data center, because these solutions are designed only for a single server or a rack enclosure. In a modern data center, however, power control needs to be enforced at three levels: rack enclosure, power distribution unit, and the entire data center, due to the physical and contractual power limits at each level. This paper presents SHIP, a highly scalable hierarchical power control architecture for large-scale data centers. SHIP is designed based on well-established control theory for analytical assurance of control accuracy and system stability. Empirical results on a physical testbed show that our control solution can provide precise power control, as well as power differentiations for optimized system performance and desired server priorities. In addition, our extensive simulation results based on a real trace file demonstrate the efficacy of our control solution in large-scale data centers composed of 5,415 servers.

Index Terms—Power capping, data centers, control theory, power management, scalability, servers.

1 Introduction

Power consumed by computer servers has become a serious concern in the design of large-scale enterprise data centers. In addition to high electricity bills and negative environmental implications, increased power consumption may lead to system failures caused by power capacity overload or system overheating, as data centers increasingly deploy new high-density servers (e.g., blade servers), while their power distribution and cooling systems have already approached the peak capacities. The goal of power control (also called power capping) is to have runtime measurement and control of the power consumed by servers, so that we can achieve the highest system performance while keeping the power consumption lower than a given power budget, which can be determined by various factors such as the capacity of the power distribution system. Precise power control, combined with power differentiation based on server performance needs, can prevent system failures while allowing data centers to operate at peak efficiencies for a higher return on investment.

In today's data centers, power needs to be controlled at three levels: rack enclosure, Power Distribution Unit (PDU), and an entire data center, due to the physical and contractual power limits at each level [2]. For example, if the physical power limits are violated, overloading of electrical circuits may cause circuit breakers to trip, resulting in undesired outages. Even though data centers commonly rely on power provisioning, the actual power consumption of the IT equipment in a data center may still exceed the power distribution capacity of the facility. A real scenario that many data centers face is that business needs require deploying new servers rapidly while upgrades of the power and cooling systems lag far behind. In some geographies, it is either impossible or cost-prohibitive to provide more power from the utility company to the data centers. For example, the power consumption of National Security Agency (NSA) headquarters in 2006, which is greater than that of the city of Annapolis, reached the power limit of the facility [3]. The agency responded by turning off noncritical equipment. In 2007, the power constraint delayed deployment of new computing equipment and caused planned outages and rolling brownouts in the NSA data center. Similar incidents are expected to increasingly occur in the coming years as more data centers reach their power limits. Therefore, it is important to control the power consumption of an entire data center.

However, to date, most existing work on server power control focuses exclusively on controlling the power consumption of a single server. Only a few recently proposed control strategies are designed for the rack enclosure level [4], [5], [6]. These centralized solutions cannot be easily extended to control an entire large-scale data center due to several reasons. First, the worst-case computational complexity of a centralized controller is commonly proportional to the system size, and thus, cannot scale well for large-scale systems [7]. Second, since every server in the data center may need to communicate with the centralized controller in every control period, the controller may become a communication bottleneck. Furthermore, a centralized controller may have long communication delays in large-scale systems. Therefore, highly scalable control solutions need to be developed.

In addition, most existing power control solutions heavily rely on heuristics for decision making. In recent

- *X. Wang is with the Department of Electrical and Computer Engineering, The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210. E-mail: xwang@ece.osu.edu.*
- *M. Chen is with NetApp, Inc, 495 East Java Drive, Sunnyvale, CA 94089. E-mail: mchen11@eecs.utk.edu.*
- *C. Lefurgy and T.W. Keller are with the Power-Aware Systems Department, IBM Austin Research Laboratory, Austin, TX 78758. E-mail: {lefurgy, tkeller}@us.ibm.com.*

Manuscript received 1 July 2010; revised 24 Nov. 2010; accepted 6 Dec. 2010; published online 14 Mar. 2011.
Recommended for acceptance by Y.C. Hu.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2010-07-0387.
Digital Object Identifier no. 10.1109/TPDS.2011.93.

years, feedback control theory has been identified as an effective tool for power control due to its theoretically guaranteed control accuracy and system stability. Control theory also provides well-established controller design approaches, e.g., standard ways to choose the right control parameters, such that exhaustive iterations of tuning and testing can be avoided. Furthermore, control theory can be applied to quantitatively analyze control performance (e.g., stability, settling time) even when the system is suffering unpredictable workload variations. This rigorous design methodology is in sharp contrast to heuristic-based adaptive solutions that heavily rely on extensive manual tuning. For example, recent work [8], [5] has shown that control-theoretic power management outperforms commonly used heuristic solutions by having more accurate power control and better application performance.

There are several challenges in developing scalable power control algorithms. First, the global control problem (i.e., power control for an entire data center) needs to be decomposed into a set of control subproblems for scalability. The decomposition strategy must comply with the data centers' power distribution hierarchy. Second, the local controller designed for each decomposed subproblem needs to achieve local stability and control accuracy despite significantly varying workloads. Third, each local controller needs to coordinate with other controllers at different levels for global stability and control accuracy. Finally, the system performance of the data center needs to be optimized based on optimal control theory, subject to various system constraints.

In this paper, we present SHIP, a highly scalable hierarchical power control architecture for large-scale data centers composed of thousands of servers. Our control architecture is systematically designed based on advanced optimal control theory for theoretically guaranteed control accuracy and system stability. Specifically, the contributions of this paper are four-fold:

- We decompose the problem of power control for a data center into control subproblems at the three levels of the common power distribution hierarchy, and then model the power consumption of each level.
- We design and analyze Multi-Input-Multi-Output (MIMO) power control algorithms for different levels based on Model Predictive Control (MPC) theory to optimize system performance, while controlling the total power to stay within the desired constraints.
- We implement our control architecture on a physical testbed and provide the implementation details of each component in the control loops.
- We present empirical results on a physical testbed to demonstrate that our solution can provide precise power control and desired power differentiation for optimized system performance and desired server priorities. With scalability constraints, our control solution outperforms a state-of-the-art centralized power controller by having better benchmark performance. We also present simulation results based on a real trace file of 5,415 servers to show the effectiveness of our solution in large-scale data centers.

The rest of the paper is organized as follows: Section 2 introduces the overall architecture of our hierarchical power control solution. Section 3 describes the system modeling, controller design and analysis of the PDU-level power controller. Section 4 discusses the coordination among controllers at different levels. Section 6 provides the implementation details of our control architecture and our empirical results on a physical testbed. Section 7 highlights the distinction of our work by discussing the related work. Section 8 concludes the paper.

2 Hierarchical Power Control Architecture

In this section, we provide a high-level description of the SHIP power control architecture, which features a three-level power control solution. First, the rack-level power controller adaptively manages the power consumption of a rack by manipulating the *CPU frequency* (e.g., via Dynamic Voltage and Frequency Scaling (DVFS)) of the processors of each server in the rack. Second, the PDU-level power controller manages the total power consumption of a PDU by manipulating the *power budget* of each rack in the PDU. Similar to the PDU-level controller, the data center-level controller manages the total power consumption of the entire data center by manipulating the *power budget* of each PDU. Our control architecture is directly applicable to data centers, where applications (e.g., scientific computing and background data processing) can allow degraded performance when power must be controlled to stay below a budget at runtime (e.g., due to thermal emergency). For data centers, where applications need to achieve specified service-level agreements (SLAs) (e.g., response time), our solution can be integrated with application-level performance control solutions (e.g., [9], [10], [11]) for simultaneous control of power and application performance.

We assume that the power limit of the upper level (e.g., the data center) is lower than the sum of the maximum power limits of all the lower-level units (e.g., PDUs). This assumption is based on two key observations of data center operation. First, many data centers are rapidly increasing their number of hosted servers to support new business in the short term, while infrastructure upgrades at upper levels happen over much longer time scales due to cost considerations. Second, lower-level units commonly have nonuniform workloads and so can rarely reach their power limits simultaneously.

There are several reasons for us to use processor frequency (and voltage) scaling as our actuation method at the rack level. First, processors commonly contribute a large portion of the total power consumption of a server [12]. As a result, the processor power difference between the highest and lowest power states is large enough to compensate for the power variation of other components, and can thus provide an effective way for server power control. Second, frequency scaling has a small overhead while some other actuation methods, like turning servers on/off, may lead to service interruption and undesired long delays. Finally, current processors support frequency scaling by DVFS or clock modulation [8], while there are still very few real disks or memory devices that are designed for servers and allow runtime transition among different *active* power modes. Note that other actuation methods can also be included in our control architecture, which is our future work.

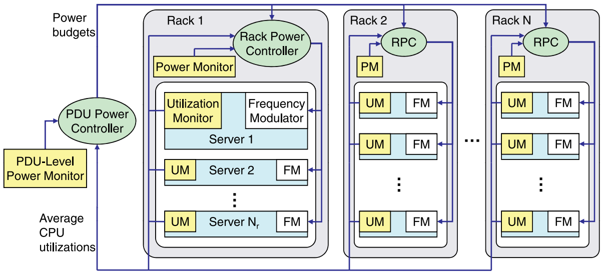


Fig. 1. Proposed power and performance control solution for virtualized server clusters.

As shown in Fig. 1, the key components in a rack-level control loop include a *power controller* and a *power monitor* at the rack level, as well as a *CPU utilization monitor* and a *CPU frequency modulator* on each server. The control loop is invoked periodically, and its period is chosen based on a trade-off between actuation overhead and system settling time. The following steps are invoked at the end of every control period:

1. The power monitor (e.g., a power meter) measures the average value of the total power consumption of all the servers in the last control period and sends the value to the controller. The total power consumption is the *controlled variable* of the control loop.
2. The utilization monitor on each server sends its CPU utilization in the last control period to the controller. The utilization values can be used by the controller to optimize system performance by allowing servers with higher utilizations to run at higher CPU frequencies. Please note that application-level performance metrics, such as response time and throughput can also be used in place of CPU utilization to optimize power allocation in our solution.
3. The controller computes the new CPU frequency level for the processors of each server, and then sends the level to the CPU frequency modulator on each server. The levels are the *manipulated variables* of the control loop.
4. The CPU frequency modulator on each server changes the CPU frequency (and voltage if using DVFS) of the processors accordingly. The rack-level power controller is designed based on the power control algorithm presented in [5]. The focus of this paper is on the power control loops at the PDU and data center levels and the coordination among controllers at different levels.

The key components in a PDU-level power control loop include a *power controller* and a *power monitor* at the PDU level, as well as the rack-level power controllers and the utilization monitors of all the racks located within the PDU. The control loop is invoked periodically to change the power budgets of the rack-level control loops of all the racks in the PDU. Therefore, to minimize the impact on the stability of a rack-level control loop, the control period of the PDU-level loop is selected to be longer than the settling time of the rack-level control loop. This guarantees that the rack-level control loop can always enter its steady state within one control period of the PDU-level loop, so that the two control loops are decoupled and can be designed independently. The following steps are invoked at the end of every control period of the PDU-level loop:

1. The PDU-level power controller receives the power consumption of the entire PDU in the last control period from the PDU-level power monitor. The power consumption is the *controlled variable* of this control loop.
2. The PDU-level controller also receives the average CPU utilization of (all the servers in) each rack from the rack-level utilization monitor. The utilizations are used to optimize system performance by allocating higher power budgets to racks with higher utilizations.
3. The PDU-level controller then computes the power budget for each rack to have in the next control period based on MPC control theory [13]. The power budgets are the *manipulated variables* of the control loop.
4. The power budget of each rack is then sent to the rack-level power controller of that rack. Since the rack-level power controller is in its steady state at the end of each control period of the PDU-level controller, the desired power budget of each rack can be achieved by the rack-level controller by the end of the next control period of the PDU-level controller.

Similar to the PDU-level control loop, the data center-level power control loop controls the power consumption of the *entire* data center by manipulating the power budgets of the PDU-level power control loops of all the PDUs in the data center. The control period of the data center-level power control loop is selected in the same way to be longer than the settling time of each PDU-level control loop.

3 PDU-Level Power Controller

In this section, we introduce the design and analysis of the PDU-level power controller. The data center-level controller is designed in the same way.

3.1 Problem Formulation

PDU-level power control can be formulated as a dynamic optimization problem. In this section, we analytically model the power consumption of a PDU. We first introduce the following notation. T_p is the control period. $pr_i(k)$ is the power consumption of Rack i in the kth control period. $\Delta pr_i(k)$ is the power consumption change of Rack i, i.e., $\Delta pr_i(k) = pr_i(k+1) - pr_i(k)$. $br_i(k)$ is the power budget of Rack i in the kth control period. $\Delta br_i(k)$ is the power budget change of Rack i, i.e., $\Delta br_i(k) = br_i(k+1) - br_i(k)$. $ur_i(k)$ is the average CPU utilization of all the servers in Rack i in the kth control period. N is the total number of racks in the PDU. $pp(k)$ is the aggregated power consumption of the PDU. P_s is the power set point, i.e., the desired power constraint of the PDU.

Given a control error, $pp(k) - P_s$, the control goal at the kth control point (i.e., time kT_p) is to dynamically choose a power budget change vector $\mathbf{\Delta br(k)} = [\Delta br_1(k) \ldots \Delta br_N(k)]^T$ to minimize the difference between the power consumption of the PDU in the next control period and the desired power set point

$$\min_{\{\Delta br_j(k)|1\le j\le N\}} (pp(k+1)-P_s)^2. \tag{1}$$

This optimization problem is subject to three constraints. First, the power budget of each rack should be within an allowed range, which is estimated based on the number of servers in that rack and the maximum and minimum possible power consumption of each server. This constraint is to prevent the controller from allocating a power budget that is infeasible for the rack-level power controller to achieve. Second, power differentiation can be enforced for two or more racks. For example, in some commercial data centers that host server racks for different clients, racks may have different priorities for power budget allocation. As power is directly related to application performance, the power budget allocated to one rack may be required to be n (e.g., 1.2) times that allocated to another rack. This is referred to as *proportional power differentiation*. The differentiation is particularly important when the entire data center is experiencing temporary power budget reduction. In that case, with power differentiation, premium clients may have just slightly worse application performance while ordinary clients may suffer significant performance degradation. Finally, the total power consumption should not be higher than the desired power constraint. The three constraints are modeled as:

$$\begin{aligned} &P_{min,j} \le \Delta br_j(k) + br_j(k) \le P_{max,j}\ (1\le j\le N),\\ &\quad \Delta br_i(k) + br_i(k) = n(\Delta br_j(k) + br_j(k))\ (1\le i\ne j\le N),\\ &\quad pp(k+1) \le P_s, \end{aligned}$$

where $P_{min,j}$ and $P_{max,j}$ are the *estimated* minimum and maximum power consumption of a rack. The two values are estimated based on the number of servers in the rack and the estimated maximum and minimum power consumption of a server when it is running a nominal workload, which can be the typical applications of the servers with the most typical load profiled based on history data. The two values may be different in a real system due to different server configurations and workloads, which could cause the controller to allocate a power budget that is infeasible (e.g., too high or too low) for a rack-level controller to achieve. This uncertainty is modeled in the system model described in the next section. Therefore, PDU-level power management has been formulated as a constrained MIMO optimal control problem.

3.2 System Modeling

We now consider the total power consumption of a PDU. The total power consumption in the $(k+1)$th control period, $pp(k+1)$, is the result of the power consumption of the PDU in the previous control period, $pp(k)$, plus the sum of the power consumption changes of all the racks in the PDU.

$$pp(k+1) = pp(k) + \sum_{i=1}^{N} \Delta pr_i(k). \tag{2}$$

As introduced in Section 2, the control period of the PDU-level controller is longer than the settling time of the rack-level controller. As a result, at the end of each control period of the PDU-level controller, the desired power budget of each rack should have already been achieved by the corresponding rack-level controller, i.e., the power consumption change $\Delta pr_i(k)$ should be equal to the power budget change $\Delta br_i(k)$. However, there could be situations that a rack may fail to achieve a given power budget because it is infeasible to do so. For example, a rack may fail to reach a given high power budget because its current workload is not as power intensive as the nominal workload used to estimate the maximum power consumption of a rack used in constraint (2). As a result, the current workload may not be enough for the rack to achieve the given power budget even when all the servers in the rack are running at their highest frequencies. In that case, the power consumption change of the rack may become a function of the change of its assigned budget, i.e., $\Delta pr_i(k) = g_i \Delta br_i(k)$, where g_i is the system gain, which is also called the *power change ratio*. Note that g_i is used to model the uncertainties of the PDU-level power controller and its value is unknown at design time.

In general, the relationship between the power consumption of all the servers in a PDU and the power budget change of each rack in the PDU can be modeled as follows:

$$pp(k+1) = pp(k) + \mathbf{G}\boldsymbol{\Delta}\mathbf{br}(\mathbf{k}), \tag{3}$$

where $\mathbf{G} = [g_1 \dots g_N]$, and $\boldsymbol{\Delta}\mathbf{br}(\mathbf{k}) = [\Delta br_1(k) \dots \Delta br_N(k)]^T$.

We apply MPC theory [13] to design the controller. MPC is an advanced control technique that can deal with MIMO control problems with constraints on the plant and the actuators. This characteristic makes MPC well suited for power control in data centers. The detailed controller design and analysis are available in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.93, of this paper. A fundamental benefit of the control-theoretic approach is that it gives us confidence for system stability. Our analysis shows that the designed MPC controller can remain stable even when the system model changes significantly due to runtime workload variations.

4 Coordination with Rack-Level Controller

In this section, we analyze the coordination among the controllers at different levels.

As discussed in Section 2, to achieve global stability, the period of an upper-level (e.g., PDU) control loop is preferred to be longer than the settling time of a lower-level (e.g., rack) control loop. This guarantees that the lower-level loop can always enter its steady state within one control period of the upper-level control loop, so that the two control loops are decoupled and can be designed independently. As long as the two controllers are stable individually, the combined system is stable. Note that the configuration of settling time is a sufficient but *not* necessary condition for achieving global stability. In other words, global stability can be achieved in some cases even when the control period is shorter than the settling time of the lower-level control loop [14].

We now analyze the settling times of the PDU-level control loop and the rack-level control loop. The settling time analysis includes three general steps. First, we compute the feedback and feedforward matrices for the controller by solving the control input based on the system model (e.g., (3)) of a specific system. The analysis needs to consider the composite system consisting of the dynamics of the original system and the controller. Second, we derive the closed-loop model of the composite system by substituting the control inputs derived in the first step into the actual system model. Finally, we calculate the dominant pole (i.e., the pole with the largest magnitude) of the closed-loop system. According to control theory, the dominant pole determines the system's transient response such as settling time.

As an example, we follow the above steps to analyze the settling times of the PDU-level controller and a rack-level controller used in our experiments. The PDU-level controller has a nominal gain vector $\mathbf{G} = [1, 1, 1]$. Our results show that the magnitude of the dominant pole of the closed-loop system is 0.479. As a result, the number of control periods for the PDU-level loop to settle is six. The rack-level controller has a nominal vector $\mathbf{A} = [56, 56, 58]$. Therefore, the settling time of the rack-level loop is 16 control periods.

5 Discussion

The key advantage of power capping is that it provides a safe way for a data center to support more servers within the limited cooling and power supply capacities. As a result, data centers can gain a maximized return on their nonrecurring facility investment. In this section, we discuss the selection of control periods in the SHIP control architecture.

There are several factors to consider regarding the selection of control periods at different levels. For example, at the PDU level, the primary factors are the circuit breaker trip time on the input power to the PDU, the amount of oversubscription on the PDU power, and the number of control periods required to settle to the desired power set point. Secondary factors include the time to measure the power consumption and server utilization, the time to perform the control algorithm, and the time to actuate DVFS at the server level for power control.

The data center power infrastructure must adhere to safety regulations by using appropriately sized circuit breakers. For example, in the United States, the National Electric Code (NEC) [15] requires the continuous power load on a circuit breaker to be at most 80 percent of the circuit breaker rating. This 80 percent power load represents the set point of the power capping controller. Circuit breakers have two types of trip-time behavior which are specified in the UL489 standard. First, short-circuits (for example, over 500 percent of rated load) cause the circuit breaker to trip within a few milliseconds. Second, overload conditions for less severe current draw can trip the circuit breaker on a time scale from milliseconds to hours, depending on the severity of the overload. Only the overload condition is relevant for the control period selection, since practical uses of power oversubscription do not reach load levels sufficient to cause a short-circuit trip condition.

A data center may safely use power shifting to oversubscribe the circuit breaker by up to 25 percent, according to the above NEC rule. At this level, the range of momentary overshoot by the controller is limited to 100 percent of the circuit breaker rated load and the breaker will never be put into an overload condition. In this case, secondary factors can be used to set the control period. We believe that oversubscription up to 25 percent is practical, low-risk, and financially attractive for data centers. For example, a selection of IBM's US data centers showed a total power consumption increase of 4 percent per year [16]. The 25 percent increase in power oversubscription from power capping would allow new data center construction costs, ranging in 100s of millions USD, to be deferred for about five years.

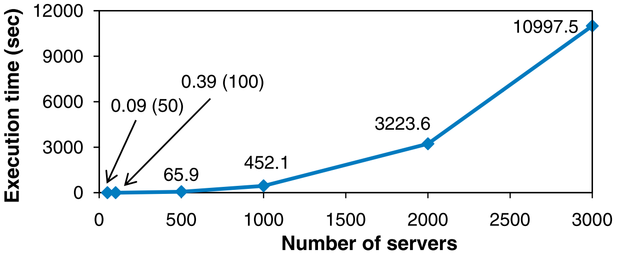


Fig. 2. Average execution time of the MPC controller for different numbers of servers.

The overload trip times must be taken into account to deal with oversubscription beyond 25 percent or unexpected power spikes caused by workload variations. For example, circuit breakers based on UL489 available from Rockwell Automation exhibit trip times of more than 2 minutes when overloaded to 125 percent of rated load (oversubscription of 56 percent) [17]. In order to avoid tripping breakers, power must be controlled to stay below the rated load within the specified trip time. This means that PDU-level controllers could use a control period of at most 20 seconds (2 minutes/6 control periods to settle). Consequently, to ensure system stability with settling time configuration (as discussed in Section 4), the rack-level controllers could use a control period of at most 1.25 seconds (20 seconds/16 control periods to settle). The capabilities of power metering equipment today can easily achieve these time intervals. For example, the Yokogawa WT210 power meter specified in the SPECpower benchmark [18] for measuring server energy-efficiency can measure power down to intervals of 40 milliseconds. The specific control periods used by our experimental cluster (5 s for the rack-level controller and 80 s for the PDU-level controller) were chosen to keep prototyping costs low and increase measurement accuracy. Our prototype control periods can be easily scaled to comply with breaker trip times by using higher class power meters.

The control period is also related to the computational complexity of the MPC control algorithm. In our prototype system the controller is based on the `lsqlin` solver in Matlab. The computational complexity of `lsqlin` is polynomial in the number of servers and the control and prediction horizons. Fig. 2 shows that the average execution time of the MPC controller increases dramatically as the number of directly controlled servers increases. For example, the MPC controller with 100 servers takes approximately 0.39 s. For a rack of 100 servers, this is well below the control period time of 1.25 s required for 56 percent oversubscription.

A final factor to consider for control period selection is the overhead of DVFS. Recent server products are able to slew frequency at a rate of 2 GHz in 50 microseconds [19]. An industry standard voltage regulator specification for servers recommends a minimum voltage slew time across the voltage range of 0.5 to 1.6 V to be no more than 110 microseconds [20]. Therefore, both frequency and voltage slew rates are well within the control period time constraints imposed by circuit breakers.

6 Empirical Results

In this section, we first introduce the physical testbed and benchmarks used in our experiments, as well as the implementation details of the control components. We then present our empirical results to demonstrate that the SHIP hierarchical control solution can provide precise power control for different power set points and when the set point is reduced at runtime (e.g., due to thermal emergencies).

We have also examined the capability of SHIP to provide desired power differentiation and compared SHIP with a state-of-the-art centralized control solution on our testbed. Furthermore, we have conducted simulations to stress test SHIP in large-scale data centers using a trace file from real-world data centers, which has the utilization data of 5,415 servers. Those additional empirical and simulation results are available in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2011.93, of this paper.

6.1 Testbed Implementation

Our testbed includes nine Linux servers to run workloads and a Linux machine to run the controllers. The nine servers are divided into three groups with three servers in each group. Each group emulates a rack while the whole testbed emulates a PDU. Servers 1 to 4 are equipped with 2.4 GHz AMD Athlon 64 3800+ processors and run openSUSE 11.0 with kernel 2.6.25. Servers 5 to 8 are equipped with 2.2 GHz AMD Athlon 64 X2 4200+ processors and run openSUSE 10.3 with kernel 2.6.22. Server 9 is equipped with 2.3 GHz AMD Athlon 64 X2 4400+ processors and runs openSUSE 10.3. All the servers have 1 GB RAM and 512 KB L2 cache. Rack 1 includes Servers 1 to 3. Rack 2 includes Servers 4 to 6. Rack 3 includes Servers 7 to 9. The controller machine is equipped with 3.00 GHz Intel Xeon Processor 5,160 and 8 GB RAM, and runs openSUSE 10.3. All the machines are connected via an internal Ethernet switch.

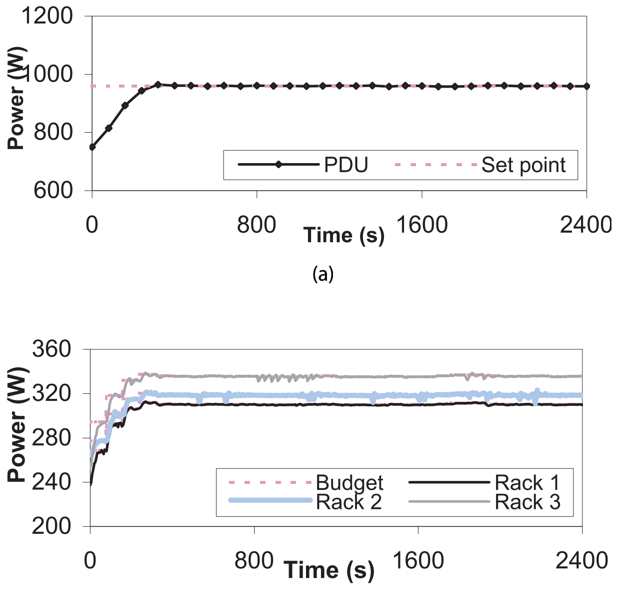
In our experiments on the testbed, we use two standard benchmarks: High Performance Computing Linpack Benchmark (HPL) (V1.0a) and SPEC CPU2006 (V1.0), as our workloads. HPL is a software package that solves a (random) dense linear system in double precision (64 bits) arithmetic. The problem size of HPL is configured to be $10{,}000 \times 10{,}000$ and the block size is set as 64 in all experiments unless otherwise noted. SPEC CPU2006 is configured with one user thread and recorded as performance ratio, i.e., the relative speed of the server to finish each benchmark (compared to a reference Sun UltraSparc II machine at 296 MHz). CPU2006 includes CINT2006 and CFP2006, which consist of integer and floating-point benchmarks, respectively. The reported result is the average of all the benchmarks in each category. Note that, we use HPL and SPEC CPU2006 as our workloads because they provide standard ways to quantify the performance improvement achieved by our control solution. Our control algorithm is *not* limited to the two benchmarks and can be used to achieve similar performance improvement for other workloads in data centers.

We now introduce the implementation details of each component in our power control architecture.

Power monitor. The power consumptions of the emulated PDU and three racks are measured with four WattsUp Pro power meters, which have an accuracy of 1.5 percent of the measured value. The power meters sample the power data every second and then send the readings to the four controllers through system files */dev/ttyUSB0* to *ttyUSB3*.

Utilization monitor. The utilization monitor uses the */proc/stat* file in Linux to estimate the CPU utilization in each control period. The file records the number of jiffies (usually 10ms in Linux) when the CPU is in user mode, user mode with low priority (nice), system mode, and when used by the idle task, since the system starts. At the end of each sampling period, the utilization monitor reads the counters, and estimates the CPU utilization as 1 minus the number of jiffies used by the idle task divided by the total number of jiffies in the last control period.

CPU frequency modulator. We use AMD's Cool'n'Quiet technology to enforce the new frequency (and voltage) level by DVFS. The AMD microprocessors have four or five discrete DVFS levels. To change CPU frequency, one needs to install the *cpufreq* package and then use the root privilege to write the new frequency level into the system file */sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed*. The AMD processors used in our experiments support only several discrete frequency levels. However, the new frequency level periodically received from a rack-level power controller could be any value that is not exactly one of the supported frequency levels. Therefore, the modulator code must resolve the output value of the controller to a series of supported frequency levels to approximate the desired value. For example, to approximate 2.89 GHz during a control period, the modulator would output a sequence of supported levels: 2.67, 3, 3, 2.67, 3, 3, etc. on a smaller timescale. The detailed modulator algorithm can be found in [8]. Clearly, when the sequence has more numbers during a control period, the approximation will be better but the actuation overhead may become higher. In this paper, we choose to use 50 discrete values to approximate the fractional frequency level, which leads to a subinterval of 100 ms during an example control period of 5 s. Since the average overhead (i.e., transition latency) of changing the DVFS level in AMD Athlon processors is about $100\ \mu s$ according to the AMD white paper report [21], the impact of actuation overhead on system performance is no more than 0.1 percent ($100\ \mu s/100\ \mathrm{ms}$), even in the worst case when the frequency needs to be changed in every subinterval. This amount of overhead is acceptable to most computer systems. In addition, recent studies [22] have shown that the overhead of DVFS in future processors can be in nanoseconds. Therefore, the overhead of DVFS is small enough to be used in real systems even when a much smaller control period is adopted.



(a)

(b)

Fig. 3. A typical run of the SHIP hierarchical control solution on the physical testbed. (a) Power consumption of the PDU. (b) Power consumptions of the three racks.

6.2 Precise Power Control

In this experiment, we run the HPL benchmark on each of the nine servers. The power set point of the PDU is 960 W. Fig. 3 shows a typical run of the SHIP hierarchical control solution. At the beginning of the run, the total power of the PDU is lower than the set point because all the servers are initially running at the lowest frequency levels. The PDU-level controller responds by giving more power budgets to all the three racks. The rack-level controllers then step up the servers' frequency levels to achieve the new power budgets within one control period of the PDU-level loop. After four control periods, the power consumption of the PDU has been precisely controlled at the desired set point, without causing an undesired overshoot. After the transient state, as shown in Fig. 3b, the power budget allocated to each rack is kept at a stable value with only minor variations. The power consumption of each rack has also been precisely controlled at their respective allocated budgets. As discussed in Section 3.2, the PDU controller tries to minimize the difference between the estimated maximum power consumption (i.e., $P_{max,j}$) and the allocated power budget for each rack in its cost function. Specifically, the maximum power consumption for Racks 1 to 3 is 339 W, 347.5 W, and 364.5 W, respectively. Since all the racks have the same weight (100 percent CPU utilization), their budgets are allocated to have the same distance with their maximum power consumptions.

In a data center, a PDU may be given different power set points at different times. For example, a data center may need to deploy a new PDU before an upgrade of its power distribution capacity can be done. As a result, the power set points of all other PDUs need to be reduced to accommodate the new PDU. Therefore, it is important to precisely control power for different power set points. We test our control solution for different set points (from 800 to 980 W). Fig. 4 plots the average power consumption of the emulated PDU with the standard deviation on the top of each bar. Each value is the average of 20 power measurements of the PDU after the PDU-level controller enters its steady state. The maximum standard deviation is only 1.08 W around the desired set point. This experiment demonstrates that SHIP can provide precise power control.

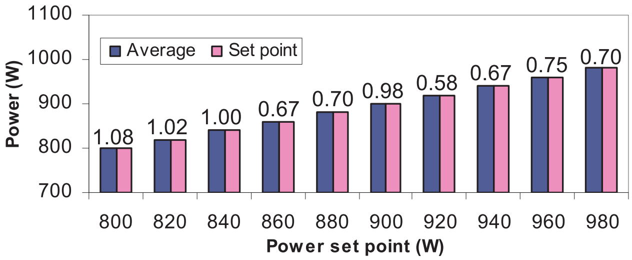
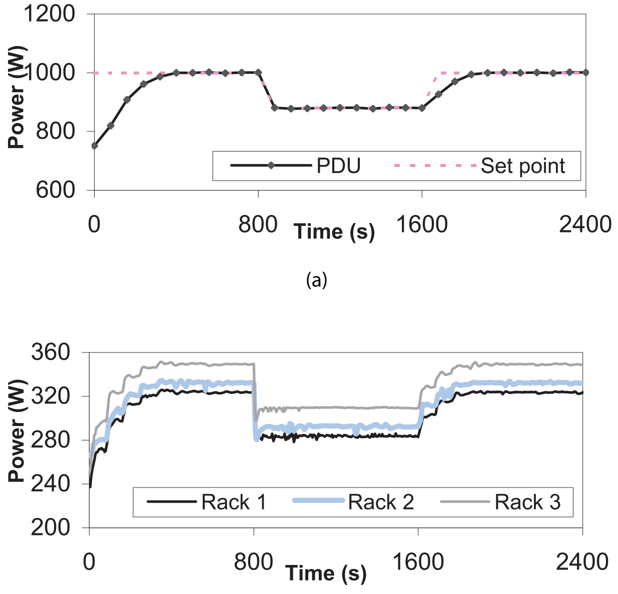


Fig. 4. Average power consumption of the emulated PDU under different power set points (with standard deviations above the bars).

6.3 Power Budget Reduction at Runtime

In this experiment, we stress test the hierarchical control solution in a scenario that is important to data centers. In this scenario, the power set point of the PDU needs to be reduced at runtime due to various reasons, such as failures of its cooling systems or its power supply systems. The set point is then raised back after the problem is fixed. A power controller designed for today's data center must be able to handle online power budget reduction because it is commonly infeasible to shut down, and then, restart all the servers with a new power set point.

As shown in Fig. 5a, the power set point is reduced from 1,000 W at time 800 s to 880 W in the next control period. As a result, the PDU-level controller reduces the power of the PDU by lowering the budgets allocated to the three racks. The racks then achieve the lowered budgets by stepping down the CPU frequency levels of their servers, as shown in Fig. 5b. Consequently, the power of the PDU converges to the new set point within in one control period of the PDU-level control loop. At time 1,600 s, the power set point is



(a)

(b)

Fig. 5. A typical run of the hierarchical solution when the power set point is reduced at runtime. (a) Power consumption of the PDU. (b) Power consumptions of the three racks.

raised back to 1,000 W. The PDU-level controller then increases the power budgets of the racks to achieve the new set point. This experiment demonstrates that SHIP can provide robust power control despite power budget reduction at runtime.

7 Related Work

Power is one of the most important design constraints for enterprise servers. Much of the prior work has attempted to reduce power consumption by improving the energy-efficiency of individual server components [16]. There has been some work on system-level power and thermal management [23], [24], [25]. For example, Nathuji and Schwan have proposed heuristic solutions for power budgeting in virtualized environments [26]. In contrast to existing work, which relies on heuristic-based control schemes, we adopt a rigorous design methodology that features a *control-theoretic* framework for systematically developing control strategies with analytical assurance of control accuracy and system stability.

Several research projects [27], [8], [28] have successfully applied control theory to explicitly control power or temperature of a single enterprise server. Some recent work has proposed heuristic-based control strategies at the rack level [4], [29]. Control-theoretic solutions have also been designed to control rack-level power consumption for optimized system performance [5]. However, those solutions cannot be directly applied to control a PDU or an entire data center because the overhead of their centralized control schemes becomes prohibitive when the system size increases to a certain extent. In contrast, our hierarchical control architecture is highly scalable for large-scale data centers.

A recent study [6] indicates the possibility of having a general group power manager that can be extended to control a data center. Our work is different in three aspects: 1) our control scheme is designed specifically based on data centers' three-level power supply hierarchy, 2) our solution features a MIMO control strategy with rigorous stability analysis, and 3) our work is evaluated on a physical testbed, while only simulation results are presented in [6]. In addition, we also present simulation results in large-scale data centers with a trace file of 5,415 servers while only 180 servers are simulated in [6]. At the PDU level, Govindan et al. [30] propose statistical profiling-based techniques to provision servers under a power constraint. At the data center level, Fan et al. [2] investigate the aggregate power usage characteristics of a warehouse-sized data center. In contrast, we dynamically control the power consumption of an entire data center and optimize system performance by shifting power among racks and PDUs. Pelley et al. propose a method of distributing PDU power feeds to reduce the number of PDUs required to tolerate PDU failures [31]. Their technique requires a power capping component, such as SHIP, to prevent long-term overload conditions.

8 Conclusions

Power control for an entire data center has become increasingly important. However, existing server power control solutions are not scalable for large-scale data centers because they are designed for a single server or a rack enclosure. In this paper, we presented SHIP, a highly scalable hierarchical control architecture that controls the total power consumption of a large-scale data center to stay within a constraint imposed by its power distribution capacity. The control architecture is designed based on rigorous control theory for analytical assurance of control accuracy and system stability. Empirical results on a physical testbed show that our control solution can provide precise power control, as well as power differentiations for optimized system performance and desired server priorities. In addition, our extensive simulation results based on a real trace file demonstrate the efficacy of our control solution in large-scale data centers composed of thousands of servers.

Acknowledgments

This is a significantly extended version of a conference paper [1]. The authors at the University of Tennessee are supported by National Science Foundation (NSF) under a CSR grant CNS-0720663 and an NSF CAREER Award CNS-0845390, and by Microsoft Research under a power-aware computing award in 2008.

References

[1] X. Wang, M. Chen, C. Lefurgy, and T.W. Keller, "SHIP: Scalable Hierarchical Power Control for Large-Scale Data Centers," *Proc. 18th Int'l Conf. Parallel Architectures and Compilation Techniques (PACT '09)*, 2009.

[2] X. Fan, W.-D. Weber, and L.A. Barroso, "Power Provisioning for a Warehouse-Sized Computer," *Proc. 34th Ann. Int'l Symp. Computer Architecture (ISCA '07)*, 2007.

[3] S. Gorman, "Power Supply Still a Vexation for the NSA," The Baltimore Sun, June 2007.

[4] P. Ranganathan, P. Leech, D. Irwin, and J.S. Chase, "Ensemble-Level Power Management for Dense Blade Servers," *Proc. 33rd Ann. Int'l Symp. Computer Architecture (ISCA '06)*, 2006.

[5] X. Wang, M. Chen, and X. Fu, "MIMO Power Control for High-Density Servers in an Enclosure," *IEEE Trans. Parallel and Distributed Systems*, vol. 21, no. 10, pp. 1412-1426, Oct. 2010.

[6] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, "No Power Struggles: Coordinated Multi-Level Power Management for the Data Center," *Proc. 13th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS '08)*, 2008.

[7] X. Wang, D. Jia, C. Lu, and X. Koutsoukos, "DEUCON: Decentralized End-to-End Utilization Control for Distributed Real-Time Systems," *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 7, pp. 996-1009, July 2007.

[8] C. Lefurgy, X. Wang, and M. Ware, "Power Capping: A Prelude to Power Shifting," *Cluster Computing*, vol. 11, no. 2, pp. 183-195, 2008.

[9] T. Horvath, T. Abdelzaher, K. Skadron, and X. Liu, "Dynamic Voltage Scaling in Multi-Tier Web Servers with End-to-End Delay Control," *IEEE Trans. Computers*, vol. 56, no. 4, pp. 444-458, Apr. 2007.

[10] Y. Chen et al., "Managing Server Energy and Operational Costs in Hosting Centers," *Proc. ACM SIGMETRICS*, 2005.

[11] Y. Wang, X. Wang, M. Chen, and X. Zhu, "Power-Efficient Response Time Guarantees for Virtualized Enterprise Servers," *Proc. Real-Time Systems Symp. (RTSS, 08)*, 2008.

[12] P. Bohrer, E.N. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell, and R. Rajamony, "The Case for Power Management in Web Servers," *Power Aware Computing*, Kluwer Academic Publishers, 2002.

[13] J.M. Maciejowski, *Predictive Control with Constraints.* Prentice Hall, 2002.

[14] X. Fu et al., "Dynamic Thermal and Timeliness Guarantees for Distributed Real-Time Embedded Systems," *Proc. 15th IEEE Int'l Conf. Embedded and Real-Time Systems Symp. (RTCSA '09)*, 2009.

[15] Nat'l Fire Prevention Assoc. "NFPA 70: National Electrical Code," 2008.

[16] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T.W. Keller, "Energy Management for Commercial Servers," *IEEE Computer,* vol. 36, no. 12, pp. 39-48, Dec. 2003.

[17] Rockwell Automation, "Bulletin 1489 Circuit Breakers Selection Guide, Publication 1489-SG001B-EN-P," Jan. 2007.

[18] SPEC, "Power and Temperature Measurement Setup Guide SPECpower v1.1," 2010.

[19] M. Ware et al., "Architecting for Power Management: The POWER7 Approach," *Proc. IEEE 16th Int'l Symp. High Performance Computer Architecture (HPCA '10),* 2010.

[20] Intel Corporation, "Voltage Regulator Module (VRM) and Enterprise Voltage Regulator-Down (EVRD) 11.1 Design Guidelines," Sept. 2009.

[21] AMD, "White Paper Publication 26094: BIOS and Kernel Developer's Guide for AMD Athlon 64 and AMD Opteron Processors, Revision 3.30," Feb. 2006.

[22] W. Kim, M. Gupta, G.-Y. Wei, and D. Brooks, "System Level Analysis of Fast, Per-Core DVFS Using On-Chip Switching Regulators," *Proc. IEEE 14th Int'l Symp. High Performance Computer Architecture (HPCA '08),* 2008.

[23] H. Zeng et al., "ECOSystem: Managing Energy as a First Class Operating System Resource," *Proc. 10th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS '02),* 2002.

[24] Y.-H. Lu, L. Benini, and G.D. Micheli, "Operating-System Directed Power Reduction," *Proc. Int"l Symp. Low Power Electronics and Design (ISLPED '00),* 2000.

[25] D. Brooks and M. Martonosi, "Dynamic Thermal Management for High-Performance Microprocessors," *Proc. Seventh Int'l Symp. High Performance Computer Architecture (HPCA '01),* 2001.

[26] R. Nathuji and K. Schwan, "Vpm Tokens: Virtual Machine-Aware Power Budgeting in Data centers," *Proc. 17th Int'l Symp. High Performance Distributed Computing (HPDC '08),* 2008.

[27] R.J. Minerick, V.W. Freeh, and P.M. Kogge, "Dynamic Power Management Using Feedback," *Proc. Workshop on Compilers and Operating Systems for Low Power (COLP '02),* Sept. 2002.

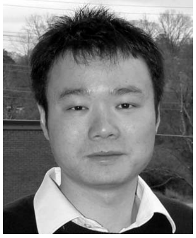
[28] K. Skadron, T. Abdelzaher, and M.R. Stan, "Control-Theoretic Techniques and Thermal-RC Modeling for Accurate and Localized Dynamic Thermal Management," *Proc. Eighth Int'l Symp. High Performance Computer Architecture (HPCA '02),* 2002.

[29] M.E. Femal and V.W. Freeh, "Boosting Data Center Performance through Non-Uniform Power Allocation," *Proc. Second Int'l Conf. Automatic Computing (ICAC '05),* 2005.

[30] S. Govindan, J. Choi, B. Urgaonkar, A. Sivasubramaniam, and A. Baldini, "Statistical Profiling-Based Techniques for Effective Provisioning of Power Infrastructure in Consolidated Data Centers," *Proc. Fourth ACM European Conf. Computer Systems (EuroSys '09),* 2009.

[31] S. Pelley et al., "Power Routing: Dynamic Power Provisioning in the Data Center," *Proc. 15th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS '10),* 2010.

Xiaorui Wang received the PhD degree from Washington University in St. Louis in 2006. He is an associate professor in the Department of Electrical and Computer Engineering at The Ohio State University. He is the recipient of the US Office of Naval Research (ONR) Young Investigator (YIP) Award in 2011, the US National Science Foundation (NSF) CAREER Award in 2009, the Power-Aware Computing Award from Microsoft Research in 2008, and the IBM Real-Time Innovation Award in 2007. He also received the Best Paper Award from the 29th IEEE Real-Time Systems Symposium (RTSS) in 2008. He is an author or coauthor of more than 60 refereed publications. From 2006 to 2011, he was an assistant professor at the University of Tennessee, Knoxville, where he received the EECS Early Career Development Award, the Chancellor's Award for Professional Promise, and the College of Engineering Research Fellow Award in 2008, 2009, and 2010, respectively. In 2005, he worked at the IBM Austin Research Laboratory, designing power control algorithms for high-density computer servers. From 1998 to 2001, he was a senior software engineer and then a project manager at Huawei Technologies Co. Ltd., China, developing distributed management systems for optical networks. His research interests include power-aware computer systems and architecture, real-time embedded systems, and cyber-physical systems. He is a member of the IEEE and the IEEE Computer Society.



Ming Chen received the BEng and MEng degrees in electrical engineering from Northwestern Polytechnic University, Xian, China, in 2002 and 2005, respectively. He received the PhD degree in computer engineering from the Department of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville in 2010, under the supervision of Dr. Xiaorui Wang. He is a performance engineer at the NetApp, Inc. in Sunnyvale, CA. From 2005 to 2006, he worked as a software engineer at ZTE Corp., China, developing embedded multimedia software. His research interests include power-aware computing.

Charles Lefurgy received the PhD degree in computer science and engineering from the University of Michigan. He is a research staff member at the IBM Austin Research Laboratory. His research interests include power management for servers and data centers and contributed to the development of the IBM active energy management product. He is a member of the ACM, IEEE, and IEEE Computer Society.

Tom W. Keller received the PhD degree in computer sciences from the University of Texas. He is a IBM's ranking technical executive in the area of power-aware computing and he is a distinguished engineer in IBM Research, Austin, where research in energy-efficient data centers, microprocessors, and computing systems is conducted. He has previously worked at the UT Computation Center, MCC and the Los Alamos National Laboratory.