Second International Green Computing Conference, Work-in-Progress, Orlando, FL, July 2011.

CACM:Current-aware Capacity Management in Consolidated Server Enclosures

Hui Chen, Meina Song, Junde Song
School of Computer
Beijing University of Posts and Telecommunications
Beijing, China 100876

Ada Gavrilovska, Karsten Schwan, Mukil Kesavan
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332

***Abstract*—Using virtualization to consolidate servers is a routine method for reducing power consumption in data centers. Current practice, however, assumes homogeneous servers that operate in a homogeneous physical environment. Experimental evidence collected in our mid-size, fully instrumented data center challenges those assumptions, by finding that chassis construction can significantly influence cooling power usage. In particular, the multiple power domains in a single chassis can have different levels of power efficiency, and further, power consumption is affected by the differences in electrical current levels across these two domains. This paper describes experiments designed to validate these facts, followed by a proposed current-aware capacity management system (CACM) that controls resource allocation across power domains by periodically migrating virtual machines among servers. The method not only fully accounts for the influence of current difference between the two domains, but also enforces power caps and safety levels for node temperature levels. Comparisons with industry-standard techniques that are not aware of physical constraints show that current-awareness can improve performance as well as power consumption, with about 16% in energy savings. Such savings indicate the utility of adding physical awareness to the ways in which IT systems are managed.**

***Keywords*-Energy Efficiency, Capacity Management, Electrical Current Balance**

I. Introduction

Reductions in data center power consumption are an inherently multi-disciplinary research endeavour. Mechanical engineers look at developing more efficient ways to cool the data center, electrical engineers are constructing more power efficient chips and power delivery systems, and computer science researchers are continuing to develop computer architectures, systems, and software to reduce power waste and enhance systems' performance-power ratios.

At data center level, server consolidation based on virtualization has been recognized as a key ingredient for enhancing power efficiency and resource utilization in cloud computing infrastructures. Research appearing in [1], [2], [3] attempts to determine optimal resource allocations that maximize power saving while attempting to meet the SLA requirements of customers. The management methods being used include DVFS (Dynamic Voltage and Frequency Scaling), turning off idle machines, and virtual machine migration. These methods, however, assume machines to be identical in terms of VM migration decisions, except for [4], which also takes into account machine heterogeneities and [5], which uses predictive methods to improve load balancing effectiveness and overheads. This paper takes a step beyond such work by presenting experimental results that show that even with homogeneous machines like the IBM BladeCenters resident in our instrumented datacenter facility, there can be distinct differences in power consumption depending on 'where' load is placed. To the best of or knowledge, this is the first study to consider the influence of chassis structure on power consumption, and in response to this study, we suggest the need for 'location-aware' methods for load placement and migration.

This paper makes the following contributions. (1) It determines that the two power domains in a single chassis have different power efficiency and in addition, chassis power consumption is increased when there are signficant differences in the electrical current draw by these two domains. (2) The paper presents a power balance controller and models the relationship between workload, current, and temperature. (3) The controller is implemented in a current-aware capacity management system (CACM), which optimizes resource allocation among blades in ways that can outperform methods that do not have information about and consider such physical machine characteristics, i.e., non-aware methods.

The remainder of the paper is organized as follows. Section 2 introduces background information about the structure of the chassis used in our data center, then proposes the hypothesis that power savings could be obtained from improved balance in electrical current. In Section 3, the current balance control model is presented. The algorithms used in these controllers are described in Section 4. In Section 5, we validate the hypothesis proposed in Section 2, and also compare the CACM implementation with methods that lack the information used by CACM. We conclude in Section 6.

II. Background

This section presents relevant background about how blade servers are constructed and about the experimental results that provide the motivation for our work.

A. Blade Server

Blade servers are commonly used in today's data centers as validated in a survey [6] that shows 76% of data centers use

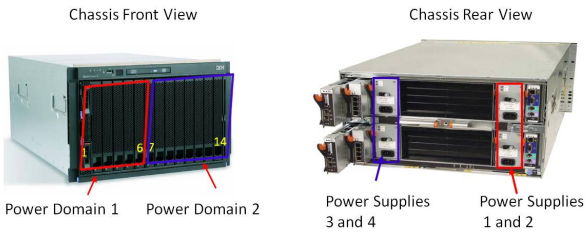

Fig. 1. IBM BladeCenter E Power Components Design

blade servers produced by companies like IBM, HP, or Dell, amongst others.

The experimental results presented in this paper are attained on an IBM 8677-3XU BladeCenter [7], as shown in Figure 1. It has a total of forteen hot-swap blade server bays and a media tray in the front. The rear of the chassis contains up to 4 power modules, 2 blowers, 2 management modules, and 4 switch (I/O) modules. These components connect to the blades in the front of the chassis through the midplane, into which all major components are plugged. The power supplies in bays 1 and 2 provide redundant power to all the BladeCenter modules and blade bays 1 through 6. The BladeCenter unit supports a second pair of power modules in power bays 3 and 4 that provide redundant power to blade bays 7 through 14.

We denote the two power domains – 1 and 2 – as PDA and PDB in this paper. In our configuration, PDB supplies power to 8 blade servers, i.e., 2 more servers than PDA. Although the PDA includes other BladeCenter modules such as Midplane, Blowers, I/O modules, etc., these modules comsume mostly static power when the chassis power is on, which is about 390 Watts in our environment. As a result, PDA typically comsumes more power than PDB when workloads are low, but when all servers reach their capacity limits, then PDB's power consumption will surpass PDA. These differences in power consumption will lead to variations in the current draw across the entire chassis. This constitutes a challenge to the three phase balance in electrical current and in general, to chassis power management.

B. Motivation

To explore the effects of current imbalance in a BladeCenter, we run VMware's ESXi server on each of its blade servers. We use VMware's vSphere [8] platform to manage this BladeCenter as a virtualized datacenter, which makes it easy to to create and migrate Virtual Machines (VMs) among blades and processors. In order to explore the relationship between current balance and the entire BladeCenter's power consumption, we conduct three experiments, as follows.

- **Scenario 1**: we place 6 virtual machines onto the blade servers of PDA, then run a CPU-intensive microbenchmark that is capable of generating a specified amount of CPU load for a given time interval. The experiment is scripted to produce a step-wise increase in CPU load from 10% to 100% at an interval of three minutes in every virtual machine being run. We collect the power, current, and average CPU usage information of PDA and PDB during the 30 minutes experiment time.
- **Scenario 2**: we place 3 VMs in each power domain, and then re-run the same experiment as those described in Scenari 1 above. In this situation, the workloads of these two domains are balanced.
- **Scenario 3**: all 6 VMs are placed into PDB, and we again run the experiments described in Scenario 1.

Experimental results are shown in Figure 2. The first figure shows the whole chassis power consumption for all three experiments, the second and third figures present the current and cpu load differences between the two power domains.

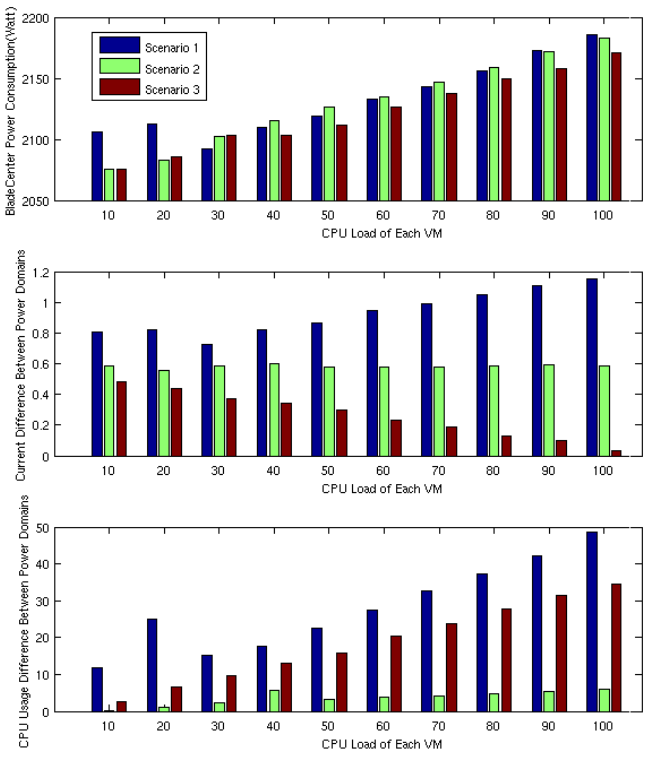

Fig. 2. Relationship Between Current Balance and Power Consumption

Experimental results lead to the following observations. First, comparing Scenarios 1 and 3, we find that under the same workload, the power usage of the entire BladeCenter is lower when the current difference between two power domains is smaller. Second, both Scenarios 1 and 3 illustrate that the same workload in PDA consumes more power than in PDB, which indicates that PDB is more power efficient. Third, load balancing between two domains does not save power, as evident from the results in Scenario 2, where the same workloads are generated in both domains, with an average CPU usage difference close to 0, but power consumption is still higher than in Scenario 3.

From this small scale experiment, we can infer that maintaining an electrical current balance across the two power domains of our BladeCenter has the potential of reducing power consumption at chassis level. A potential reason for these savings is the three-phase current balance when we

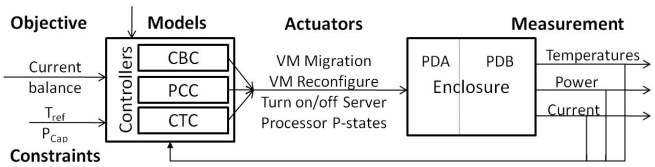

Fig. 3. CACM Control System

achieve an electrical current balance between the two power domains. Achieving such a three-phase current balance is also important to equipment maintenance in that it can extend the lifetime of the power provisioning infrastructure used in the bladecenter.

III. The Current Balance Control Model

This section discusses the CACM (Current-aware Capacity Management) system architecture and its control model, the goal of which is to achieve current balance between the two power domains. Figure 3 illustrates the key components of the CACM control system.

A. Objective Functions and Constraints

As stated in the previous section, current balance between the two domains could help reduce BladeCenter power consumption. So, in this control system, our objective function is to minimize the current difference between PDA and PDB.

$$min\left(|C_{PDA} - C_{PDB}|\right) \tag{1}$$

where C_{PDA} is the current of PDA and C_{PDB} is the current of PDB.

For regulation methods that reduce the current difference between these two domains, there are two constraints that must be obeyed during the actuation process. One is the temperature reference point for the thermal safety requirement of the CPUs; the other is to maintain some power cap for the entire chassis. There are multiple reasons for imposing such caps, but in our case, we do so in order to maintain a cap on the total thermal load the chassis imposes on the surrounding environment, and with such a cap, we can then be certain that chassis temperatures remain within safe limits given the HVAC system's cooling capacity. We use T_{ij} to denote the ith CPU temperature of blade j,

$$T_{ij} \leq T_{ref}, \textit{for each CPU } i \textit{ in blade } j \tag{2}$$

where T_{ref} is the reference threshold that depends on the tolerance level of the CPU, which is usually $70°C \sim 80°C$ and varies according to the model of the CPU being used.

P_j is used to present the power consumption of domain j, and it should satisfy the following equation,

$$P_j \leq P_{cap}, j = 1, 2 \tag{3}$$

where P_{cap} is the power cap for the power domain. For the BladeCenter used in our experiment, this value is 2000W.

The above objective function (Equation 1) and constraints (Equations 2-3) can be used to formulate an optimization problem. Before doing so and then presenting a solution method for such a problem, we first comment on the measurements and actuators used in our work.

B. Measurements and Actuators

There are two components of measurement data being gathered: one part includes data regarding the physical equipment, such as CPU temperature, power usage, and current footprint, which are collected from many sensors located in and around the BladeCenter. The other is information about the IT system, such as host and virtual machine (VM) CPU usage and requirement, memory usage and requirement, and the resource allocation among those VMs. In our system, the latter monitoring data is obtained from the ESXi server through the vSphere client API, and the former is obtained from an environmental measurement and monitoring system we have constructed for our experimental data center facility [**?**].

The following actions can serve as control system actuators: VM migration and reconfiguration, blades and VMs on/off states, processor P-states, etc. They may be used by controllers aiming to reduce current differences, to regulate power usage, to assure CPU safe operating temperature, etc.

C. Controllers

We present three controllers: CBC (Current Balance Controller), PCC (Power Capping Controller), and CTC (CPU Temperature Controller). To curtain environmental system monitoring overheads, these controllers work at different time scales, determined by corresponding event rates. For example, the power overspend situation happens rarely, so we can schedule PCC to run only every few hours, whereas current imbalance occurs more often, so that CBC should run more frequently, every few minutes. The same applies to CTC.

The actuators mentioned above differ in the overheads they impose on the system and their impacts on application/VM performance. Therefore, controllers assign different 'priorities' to different actuators. For instance, consider CBS operating in a situation where the current difference between the two BladeCenter domains is above a preset threshold value. The CBS must choose to take an appropriate action. Given that VM migration or reconfiguration will lead to degradation of service, the controller first checks whether there are any idle blades in the domain with bigger current draw. If so, then it invokes the actuator, which lowers the P-states of the idle blades, or even turns them off. Else, if no idle blades exist, the controller selects VM migration and load reconfiguration as an actuator, so as to migrate part of the workload from the more intensive domain to the less loaded one. Because VM migration and reconfiguration usually takes a longer time than other methods and can affect the quality of running services, it is being considered after other actuation techniques, which have lower overheads or less impact on runtime performance.

D. Models

After deciding on the actuation method appropriate for a controller in particular circumstances, we need additional

information to further specify the actuator's behavior, in response to questions such as the following. How many blades should be turned off? Which VM should be migrated to another blade? Which blade should receive the migrated VM? In order to generate answers to such questions, and to optimize the target control objectives under the given constraints, we rely on models that capture the relationship between CPU utilization and CPU temperature, power domain power, or current draw. We will present two such models, which correlate the domain current and CPU temperature with resource utilization. These models are then used to determine how much workload needs to be migrated when balancing the current or when maintaining the safe operating temperature for the CPUs.

Based on the experimental data obtained in our instrumented data center and using the linear regression method, we find the following relationship between ΔC and ΔU:

$$\Delta C = \Delta\alpha * \Delta U, \qquad \Delta\alpha \in [0.005, 0.007] \tag{4}$$

From Equation 4, we can quickly predict how much CPU workload (ΔC/$\Delta\alpha$) needs to be migrated when the current difference ΔC surpasses the threshold. Next, we can select the VM(s) whose total CPU utilization approximates this value, and migrate them to other blades.

We can also determined from these experiments the blade's CPU temperatures at certain workloads, which we observe to be a linear function. We can then use this function to set some safe workload threshold for the CPU useful for the CTC controller.

IV. CACM:Current-aware Capacity Management

This section discusses the CACM system's implementation details and the rationale behind our design choices to achieve energy efficiency. We again note that the three controllers introduced in last section run at different time intervals; they are run inside the resource allocation controller that is responsible for resource provisioning, adjustments, and additions. In this paper, we do not consider capacity additions, however, but deal only with new VM arrivals and with periodic resource adjustments among existing VMs.

A. VM Resource Allocation

There are two resource allocation policies:

1) prefer to use the blades in PDB when the two domains have the same current levels or are at their initial states; or
2) place more workload in PDB when it is impossible to achieve balanced electrical currents.

All information about hosts and virtual machines is updated periodically by the ESXi hypervisor, and it is accessible via the VI Java API [9]. The controller algorithms used in our work typically first read the updated host and VMs information, and then use this uptodate information to support their control decisions.

In our experiments, the resource controller assumes that all VM images are available in NFS storage, so that there is no need to create a new image upon VM arrival. This reduces response time and improves service quality. Further, since all of our VMs are configured identically, the system can be started by randomly selecting some VMs to run. When starting a VM, the host with minimum available resources among those that can satisfy the request will be selected, the idea being to increase resource utilization and server consolidation. Three controller threads periodically check the event trigger to balance the current and ensure proper system execution.

Algorithm 1 Current Balance Controller

```
1:  while(true)
2:  {
3:    ΔC=CompareDomainCurrent();
4:    UpdateVMs(VMs);
5:    UpdateHosts(Hosts);
6:    if( ΔC ≥ C_ref) {
7:     workload = ΔC/Δα;
8:     workload^save_vms = selectOffVMs(workload);
9:     if(workload^save_vms < workload) {
10:      Δworkload_vm = workload − workload^save_vms;
11:      workload^save_hosts=selectOffHosts(Δworkload_vm);
12:      if(workload^save_hosts < Δworkload_vm) {
13:      Δworkload_host = Δworkload_vm − workload^save_hosts;
14:       vms=selectMigrateVMs(Δworkload_host);
15:       hosts=selectDestinationHosts(domainNum);
16:       Migrate(vms, hosts);
17:      } } }
18:    sleep(5 minutes);
19: }
```

B. Periodic Resource Adjustment

As stated in the previous section, there are three controllers that run at different time scales, but they all use the same actuators to accomplish their control objectives. We next introduce the algorithm run in the CBC controller, which is the main controller used to reduce power usage; the other two controllers differ only in that they use different trigger events.

As seen in Algorithm 1, the first priority action is to turn off VMs, then turn off hosts, and the last choice is to migrate VMs. A VM or host is considered idle if its time period of zero CPU usage exceeds some value and in that case, it can be turned off to reduce current and power levels. If there is no idle VM or host, then the only choice is to migrate VMs, and this is the most common situation in our experiment because of the workloads being used. The domainNum used in the algorithm is the domain that has smaller current. The candidate idle VMs and hosts are all selected from the domain that has the larger current level.

V. Experimental Evaluation

A. Experiment Environment

The experiment testbed is depicted in Figure 4. There are four main parts of the system: IBM BladeCenter, control server, PI server, and NFS storage.

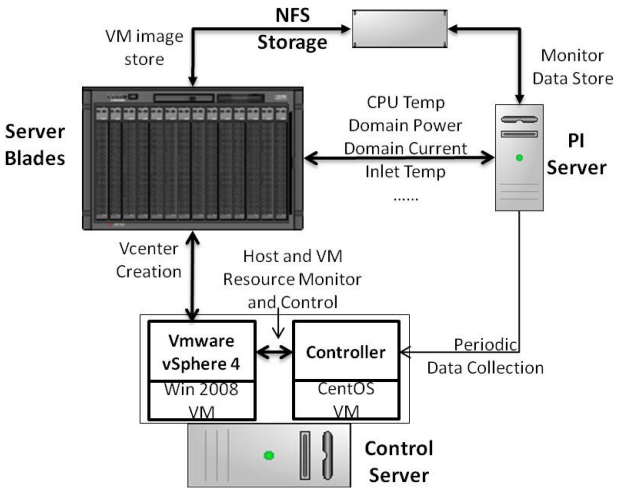

Fig. 4. Experimental Setup

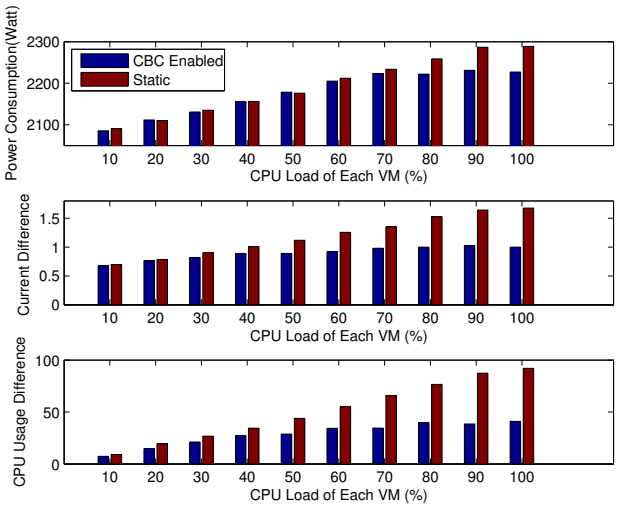

Fig. 5. Validation Result

- **BladeCenter**: its configuration and structure are described in the background section, the only additional information of note being that ESXi 4.0 is installed on these blades, which are grouped as a virtual datacenter by the VMware vSphere client used in the control server.
- **Control Server**: the control server accomodates two virtual machines, one running Windows 2008 to install the VMware vSphere Client. This VM is used to collect VM- and host-level information, and it also supports the execution of control commands such as VM migration, on/off,etc. The other VM runs CentOS and executes the resource allocation controller. The controller server not only collects the cyber information, but also gathers the physical information such as CPU temperature, Inlet temperature, chassis power and current, which are fetched from the PI server used as a collection point for physical infrastructure data.
- **PI server**: this server collects environmental information for our datacenter, such as the inlet temperature of each BladeCenter, CPU temperature of each CPU, the power, current and voltage information of each power strip outlet, PDU outlet, etc. The PI system is a commercial product provided by OSIsoft company[10].
- **NFS Storage**: to enable 'hot' VM migration, NFS storage is used to store all virtual machine images and also part of the monitoring data produced by the PI server. This storage is accessible to the control server.

B. Validation

In this section, we run the microbenchmarks introduced earlier to generate workload in each VM. We use 12 VMs, all of which initially run in the PDA domain, and we then compare the same metrics between CBC-enabled and physically unaware controls.

In Figure 5, when controllers are unaware of current levels, we find that the current and CPU usage differences between PDA and PDB increase with increasing workload. The larger CPU usage difference reaches almost 100 percent when there is no workload in PDB. In the CBC-enabled case, we find that the current and CPU usage differences are significantly lower between these two domains, and are nearly at the same levels during the last several samples. Most importantly, the whole system power consumption is reduced substantially, up to 3% of the static situation for the last experiment steps. This validation experiment proves our previous hypothesis that current balance can reduce total power usage for the same workload. We next test our CACM system in an actual cluster production situation to compare it with physically unaware management methods.

C. Comparison with Unaware Controls

1) Unaware Controls: For unaware controls, we use VMware's native Distributed Resource Scheduler (DRS), which aggregates compute capacity across a collection of servers into logical resource pools and intelligently allocates available resources among the virtual machines based on pre-defined rules that reflect business needs and changing priorities. VMware's Distributed Power Management (DPM), included with VMware DRS, is also enabled. DPM's purpose is to reduce power consumption across the collection of servers in a VMware DRS cluster [11] by turning off idle machines.

2) Analysis of Eperimental Results: In this experiment, we run 20 VMs in the BladeCenter, each VM configured with 2 vCPUs running the same workload as in the previous experiment. Each vCPU workload increases from 10% to 100% in 30 minutes, and workload increases 10% in every 3 minutes. First, we place all of these VMs and blades into a DRS- and DPM-enabled virtual cluster, allocating the VMs equally across the two domains, with each domain hosting 10 VMs, and every 2 VMs sharing a blade. We then start the workload generating VMs as well as monitoring data collection. To conduct the comparative experiment, we then move all of the blades and VMs out of the vcluster, and set up VMs as in the first experiment, again start workload

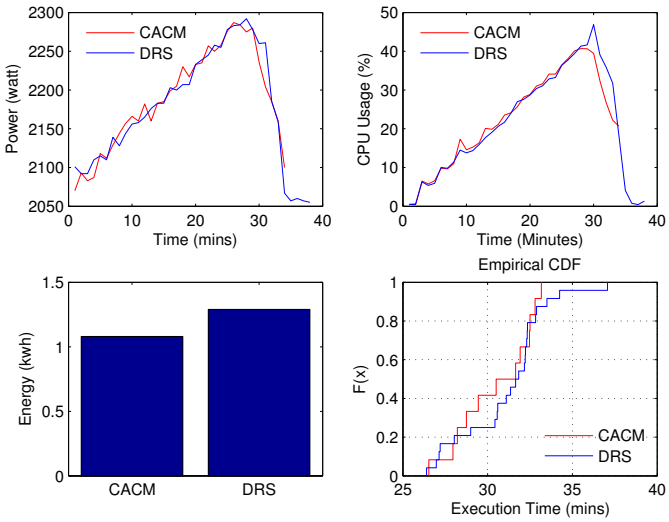

Fig. 6. Power Usage:CACM compare with VMware's DRS

generation, the current balance controller, and monitoring components. The results are presented in Figure 6.

There are four graphs in the figure, which show the power consumption, CPU usage, energy cost, and benchmark execution time distribution of the CACM- vs. the DRS-based experiments. From the last graph, we find that most benchmark execution times surpass the specified 30 minute experiment duration. The longest execution time of the CACM-based experiment is 34 minutes, compared with 38 minutes for the DRS-based experiment. This difference is due to VM migration during the experiment, and there is also some resource contention among the VMs if three or more VMs run on the same blade server (due to limited memory capacity on our servers). In summary, we find that 40% of the VMs finish their specified task within 30 minutes when using the CACM-based physically aware controls, compared with 24% in the DRS non-physically aware control experiment. In other words, CACM-based capacity management has the potential of improving application performance and perhaps more importantly, physically aware capacity management need not degrade the performance of distributed applications running on cluster systems. Finally, under fixed workload conditions, when applications finish early and machines then go into idle states or are turned off, power savings can result, up to 16% for the experimental results shown in the figure.

With these encouraging results, a next useful step for this work is not to replace DRS, which is known to perform well in terms of achieving cluster-level load balance for consolidated server systems, but instead, to add to it CACM's notion of physical awareness. We have not yet been able to do so because the rule interface exported by ESXi does not support us in taking that step.

VI. Conclusions and Future Work

This paper proposes a current-aware capacity management method to optimize blade server power consumption. The method is motivated by the discoveries that (1) the different power domains of BladeCenter chassis can have different power efficiencies and (2) differences in electrical current levels between these domains can contribute to additional power usage for the chassis. In response, we devise and experimentally demonstrate viable a current balance control policy that can improve performance and power usage compared to unaware policies.

Our conclusios is that physical information about the data center can be and should be used in IT management policies, if they are to obtain the reduced levels of power consumption required for future energy-efficient data center systems.

Our future work will attempt to enhance existing methods for IT system management with physical information, so as to benefit from such methods' sophisticated management abilities.

Acknowledgment

The authors would like to thank Chad Huneycutt and Vikneshan Sundaralingam who helped build the monitoring and experiment infrastructure. This work is partially supported by the National Key project of Scientific and Technical Supporting Programs of China (Grant Nos.2008BAH24B04, 2008BAH21B03, 2009BAH39B03); the National Natural Science Foundation of China(Grant No.61072060); the Program for New Century Excellent Talents in University (No.NECET-08-0738); Engineering Research Center of Information Networks; and the Chinese Ministry of Education. Data center instrumentation and measurement infrastructures have been constructed with support from the U.S. National Science Foundation. We also thank the anonymous reviewers who helped improve the quality of the paper.

References

[1] G. Jung, M. A. Hiltunen, K. R. Joshi, R. D. Schlichting, and C. Pu, "Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures," in *ICDCS*, 2010, pp. 62–73.

[2] R. Urgaonkar, U. C. Kozat, K. Igarashi, and M. J. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *NOMS*, 2010, pp. 479–486.

[3] I. Goiri, F. Julia, R. Nou, J. L. Berral, J. Guitart, and J. Torres, "Energy-aware scheduling in virtualized datacenters," *Cluster Computing, IEEE International Conference on*, vol. 0, pp. 58–67, 2010.

[4] R. Nathuji, C. Isci, E. Gorbatov, and K. Schwan, "Providing platform heterogeneity-awareness for data center power management," *Cluster Computing*, vol. 11, no. 3, pp. 259–271, 2008.

[5] S. Kumar, V. Talwar, V. Kumar, P. Ranganathan, and K. Schwan, "vmanage: loosely coupled platform and virtualization management in data centers," in *Proceedings of the 6th international conference on Autonomic computing*, ser. ICAC '09. New York, NY, USA: ACM, 2009, pp. 127–136.

[6] (2008) Data center users' group survey results. [Online]. Available: http://datacenterug.org

[7] IBM BladeCenter E Chassis. [Online]. Available: http://www-03.ibm.com/systems/bladecenter/hardware/chassis/bladee/index.html?

[8] VMware vSphere 4. [Online]. Available: http://www.vmware.com/products/vsphere/

[9] VMware VI (vSphere) Java API. [Online]. Available: http://vijava.sourceforge.net/

[10] The PI System. [Online]. Available: http://www.osisoft.com/Default.aspx

[11] "VMware Distributed Resource Scheduler (DRS)," White Paper, VMware, 2009.