Thermal-aware Resource Management Framework

Xi He
Golisano College of Computing and Information Sciences, Rochester Institute of Technology
102 Lomb Memorial Drive, Rochester, NY 14623-5608
e-mail: hexi111@gmail.com

***Abstract*—Large energy consumption in data centers has become a challenging problem with the emergence of cloud computing and large scale data centers. In this paper, we present an architectural framework for thermal-aware resource management while considering energy efficiency. The framework consists of a layered architecture and integrates an set of easy-to-use client tools and a thermal-aware task management middleware to schedule tasks based on thermal conditions within a cluster and among different data centers. As part of this paper we focus on the development of a thermal-aware task scheduling component for a single data center. This component is fundamental to our future activities, while considering to balance the temperature distribution in a single data center, thus implicitly minimizing energy cost in data centers.**

I. Introduction

In recent years, many large scale data centers have been deployed with high density computing clusters and server farms to support high performance scientific applications. However, besides the scientific challenges, many operational issues in data centers need to be addressed. One of those issues is the large energy consumption in data centers. According to U.S. Environmental Protection Agency (EPA), 61 billion kilowatt-hours of power was consumed in data center in 2006, that is 1.5 percent of all US electricity consumption costing around $4.5 billion [1]. In fact, the energy consumption in data centers doubled between 2000 and 2006 and EPA estimates that the energy usage will double again by 2011.

Data centers, as the typical computing resources, their management is very sophisticated and complex. How can the end users with little computer knowledge easily utilize the computing power in data centers? How can the scientists collaborate with other scientists and seamlessly integrate the computing ability provided by data centers into their research? Since there is more concern about the energy consumption in data centers, the data center administrators are interested with the thermal and energy consumption situation in data centers so that they can take efficient measures to increase the system reliability and reduce the energy consumption.

Much effort has been spend towards addressing the power crisis in data centers. For example, large amount of research work has been conducted on smart cooling techniques [2], [3], [4], [5] to improve energy-efficiency in data centers. In [6], [7] dynamic voltage and frequency scaling (DVFS) is proposed to reduce processor power dissipation with modest performance loss. Server virtualization [8] is another way to address the large power consumption problem. Most servers and desktops today are in use only 5-15% of the time they are powered on, yet most x86 hardware consumes 60-90% of the normal workload power even when idle. With virtualization technique, all the VMs share the same physical resources and the utilization of the physical server can reach up to 80% with ignorable overhead [9]. Thus, although using virtualization results typically in a performance loss using the available hardware more efficiently provides great contribution in regards to the utilization. In addition, a wide variety of task scheduling algorithms are reported to minimize the energy consumption in data centers [10], [11], [12], [13], [14].

Temperature is considered as an important physical metrics in data centers [15]. First, efficient thermal management can decrease the cooling costs in data centers. For example, in [13] it is pointed out that a data center that can run the same computational workload and cooling configuration, but maintain an ambient room temperature that is 5°C cooler through intelligent thermal management, can lower computer room air conditioning (CRAC) power consumption by 20%-40% for a $1-$3 million savings in annual cooling costs. Second, efficient thermal management can also increase hardware reliability. Some studies [16], [17] show that a 15°C rise increases hard disk drive failure rates by a factor of two.

A recent research work [12] indicates that computing nodes' temperature distribution will affect the energy cost of cooling system in data centers. It also shows that minimizing the maximal temperature of all the nodes will minimize the cooling energy cost of a data center. Assume a set of tasks consume a fixed amount of energy in data centers, an appropriate temperature distribution of computing nodes will reduce the maximal temperature of all the nodes and thus significantly conserve cooling energy consumption. There exists many available metrics for measuring temperature distribution, using the minimization of the standard deviation is one possible strategy that we will exploit.

In our study, we present a thermal-aware resource management framework (TARMF) for managing resource in a energy efficient fashion. As part of our long term research effort, we are currently focusing on thermal-aware task scheduling, which is an important component in TARMF. The contribution of this paper is two-fold: first, we present a novel framework to solve resource management problem. Second, we developed a thermal-aware task scheduling for data centers, which will save an amount of cooling energy cost. We employ standard deviation of computing nodes' temperature as our metrics.

The reminder of this paper is organized as follows. First we introduce requirement of thermal-aware resource management framework and the multi-layer architecture of this framework in Section 2,3. Then we spent the rest of the sections discussing thermal-aware task scheduling, which is an

important component in thermal-aware resource management framework. We review related works and present an example to illustrate our thermal-aware task scheduling in Section 4 and 5. In Section 6, 7, we model data centers and their components, define our problem and present our solution to the problem. In the last section, we conclude the paper and propose our future work.

II. Requirements

As mentioned in Section 1, resource management is an important and complex research topic and involves a wide variety of issues ranging from resource organization, resource discovery, resource access to resource integration and security. It is essential that all issues are addressed in a distributed system in that system end users, system administrators and resources are geographically dispersed. The system is preferred to be flexible so that it is convenient to update some of the system's functionality. For example, data centers used to adopt Backfilling Algorithm [18] to schedule tasks with the aim of increasing system utilization. Yet in this paper we presents a novel thermal-aware task scheduling for the purpose of saving energy cost. Replacing the legend algorithm with our algorithm should not lead to significant change of the system.

In addition, the system is expected to be scalable. For example, different kinds of client tools can be easily developed and integrated into the system to meet different end users' requirement. Furthermore, the system must be easy to use and integratable in other frameworks. This consequently results in some features. One of these required feature is the command-line based client tool. Many scientists get accustomed to Unix system and they need the command-line based tool so that they can easily get access to computing resources. However, for less advanced users, they may require GUI based client tools. Another required feature is cross-platform and Internet-based security scheme. This scheme can provide discrete authority and authentication and can be easily integrated into other system components.

III. Thermal-aware resource management framework

To address our requirments we are designing a thermal-aware resource management framework (TARMF). As this framework is quite extensive we have chosen a layered architecture design that allows us to gradually expand upon it and implement it in phases. In Figure 1 we show our TARMF while focusing on the 3-layer architecture design with a resource, mediator and client layer. This design addresses explicitly the requirements that we have identified in Section II. We describe the functionality of each layer next.

- The *resource layer* contains a number of advanced cyberinfrastructure resources. For our research we focus on clusters that are housed in data centers. Most importantly, it also contains a service that maps tasks onto a cluster while considering thermal properties.
- The *client layer* is composed of a set of easy-to-use client tools designed to facilitate the use of computing resources without putting much burden on the end users. However it is important to note that we provide a variety of different interfaces for different end users. Programmers are able to use APIs, administrators are exposed through scripts, and all of them can in addition use graphical user interfaces to interact with TARMF. This helps especially those that may just use TARMF as a monitoring tool.
- The *mediator layer* provides all services that establish the connection between the client and the resource layer. Important features of this layer include: the secure connection between clients an resources, the availability of a super scheduling service to distribute tasks not only on one data center, but across data centers while considering thermal properties. Furthermore, the mediator layer must have a sophisticated workflow engine included that allows interfacing with Grids and clouds easily and provides scripting capabilities.

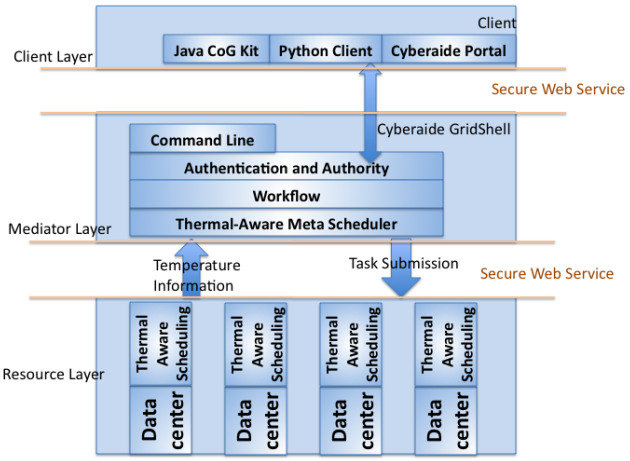

Fig. 1. The 3-layered architecture of our TARMF.

The most typical and important computing resources in the *resource layer* are data centers which house computer system, storage system and other associated components and provide powerful computing ability for task execution. Each data center is equipped with a thermal-aware task scheduling component, all of which are integrated into an overarching thermal scheduling component that are together integrated in our framework controlled, monitored, and accessed through the mediator layer. In accordance to prior work in Grid computing on meta scheduling [19] we also refer to it as *thermal-aware meta scheduler*.

Besides the *thermal-aware meta scheduler*, the *mediator layer* also provides such functionality as task organization, security and client interface API. We have designed a sophisticated mediator service Cyberaide Shell [20]. Cyberaide Shell provides a set of services such as task management service, authentication and authority service, task submission service to allow users to get access to complex cyberinfrastructure using its command line interface. Cyberaide Shell also exposes its functionality to a wide variety of frameworks and programming languages so that other tools in the client layers can be developed based on Cyberaide Shell.

The client layer is targeted to offer users who are unfamiliar with the complex Grid and data center infrastructure, a tool to decrease the learning curve of using sophisticated computing resources across different data centers, complex remote resource that are independently managed by differnently organizations, a key concept of Grids. To further assist in providing access to non computer scientists, we have also developed a Cyberaide Portal [21] framework that allows accessing data centers through a web browser with a sophisticated window based user interface and integrate Web Services, Javascript and other Web 2.0 technology. We are working on adding new components for auditing and monitoring thermal-aware task scheduling across data centers.

IV. Related Work

Thermal-aware task scheduling [22], [23], [24], [25], [26], [27] has been one of important research topics in the area of embedded systems because efficient power management is critical in those system in which there is no dedicated energy source. Recently, with the emergence of cloud computing and the development of large scale data centers, a number of power-aware task scheduling [10], [11] and thermal-aware task scheduling [12], [13], [14] have been presented.

In [10], [11], power reduction is achieved by the power-aware task scheduling on DVS-enabled commodity systems which can adjust the supply voltage and support multiple operating points. In [12], [13] thermodynamic formulation of steady state hot spots and cold spots in data centers is examined and based on the formulation several task scheduling algorithms are presented to reduce the cooling energy consumption. In [14] the researchers study the task characteristics and temperature profiles for a subset of SPEC'2K benchmarks and exploit these profiles to suggest several scheduling algorithms aimed at achieving lower cluster temperature. However, these algorithms are not aimed at energy consumption and they are intuition based algorithms that cannot guarantee energy efficiency.

In our work, we study different types of task temperature profiles and the relationship between temperature distribution and cooling energy cost in data centers. Based on these studies, we mathematically define and develop our task scheduling targeted at conserving energy consumption and increasing system reliability.

V. A Motivational Example

Given certain compute processors and steady ambient temperature, a task-temperature profile shows the temperature increase along with the task execution. It has been observed that different types of computing tasks generate different amount of heat, therefore featuring with distinct task-temperature profiles [14].

A task-temperature profile can be obtained by using profiling tools. As the task temperature is related to processor type and ambient environment and normal profiling tools can only get task-termperture in a standard environment, the task-temperature is also termed as general task-temperature profile. Figure 2 shows an overall task-temperature profiles in the

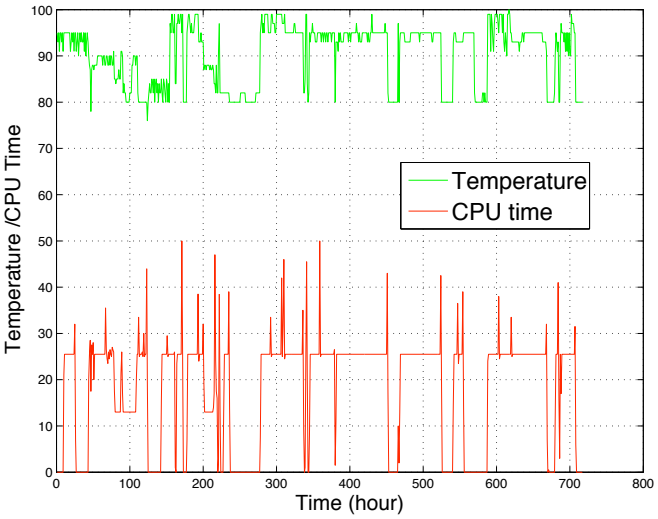

Fig. 2. Task-temperature profiles in buffalo data center

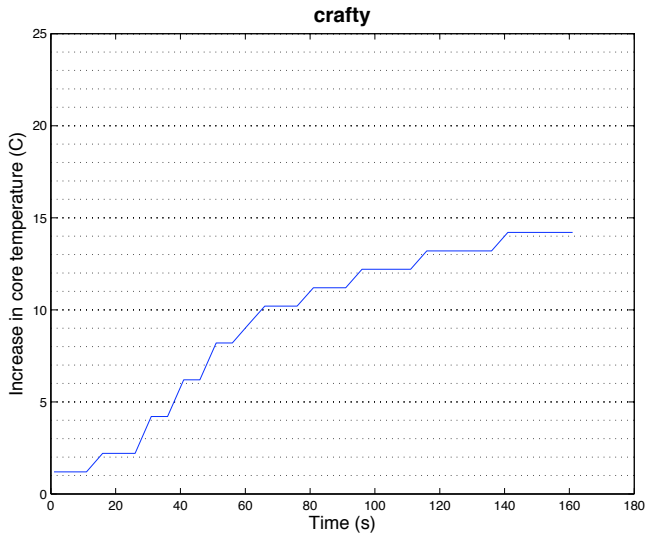

Fig. 3. Task-temperature profile of SPEC2000 (crafty)

Center for Computational Research at State University of New York at Buffalo [28]. The X-axis is the time, and the Y-axis gives two values: the upper line is the task execution time (CPU time), the lower line shows computing node temperature. Figure 2 indicates the task and temperature correlation: as compute loads in term of task CPU time augment, computing node temperatures increase incidentally. Their correlation coefficient is 0.65. To be able to design a scheduling algorithm, we are also interested in explicit task-temperature profiles to derive some elementary characteristic that we like to model. Figure 3 [14] shows a task-temperature profile, which is obtained by running SPEC 2000 benchmark (crafty) on a IBM BladeCenter with 2 GB memory and Red Hat Enterprise Linux AS 3.

Now we present a simple example to illustrate thermal-aware task scheduling which is based on the mentioned observation and experiments. Suppose there are 2 tasks, job_1 and job_2, with task-temperature profiles, $f(job_1)$, $f(job_2)$

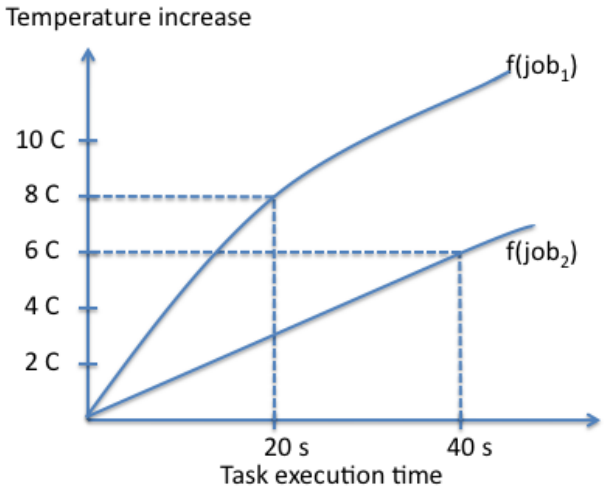

Fig. 4. Two types of tasks' task-temperature profiles

shown in Figure 4. Now job_1 and job_2 are modeled as follows:

$$job_1 = (0, 2, 20, f(job_1))$$

Where, job_1 starts at 0 second, needs 2 processors; Its execution time is 20 seconds, and task-temperature profile is $f(job_1)$.

$$job_2 = (0, 1, 40, f(job_2))$$

Where, job_2 starts at 0 second, needs 1 processors; Its execution time is 40 seconds, and task-temperature profile is $f(job_2)$.

Suppose we have 4 identical computing nodes, with current temperature:

$node_1 = 40\ ^\circ C$
$node_2 = 32\ ^\circ C$
$node_3 = 34\ ^\circ C$
$node_4 = 32\ ^\circ C$

The goal of our task scheduling is to minimize the temperature profile deviation of computing nodes after tasks are executed, and try to minimize the maximal temperature of computing nodes (mathematically, these two objectives are the same). Now a thermal-aware optimized scheduling of job_1, job_2 is as follows:

$Schedule_1$
$job_1 \rightarrow node_2, node_4$
$job_2 \rightarrow node_3$

After task execution, the temperature of these nodes will become:

$node_1 = 40\ ^\circ C$
$node_2 = 40\ ^\circ C$
$node_3 = 40\ ^\circ C$
$node_4 = 40\ ^\circ C$
$Max\ temperature = 40\ ^\circ C$
$\sigma = 0$

Where, the highest temperature among these nodes are 40 $^\circ C$. Standard deviation of these temperature is 0.

If the tasks are allocated with another schedule, for example:

$Schedule_2$
$job_1 \rightarrow node_1, node_2$
$job_2 \rightarrow node_3$

After task execution, the temperature of these nodes will become:

$node_1 = 48\ ^\circ C$
$node_2 = 40\ ^\circ C$
$node_3 = 40\ ^\circ C$
$node_4 = 32\ ^\circ C$
$Max\ temperature = 48\ ^\circ C$
$\sigma = 5.6$

Where, the highest temperature among these nodes are 48 $^\circ C$. Standard deviation of these temperature is 5.6.

Therefore, $schedule_1$ is better than $schedule_2$.

VI. System Model

This sections provides the formal description of data centers, nodes, jobs and the problem, which are employed as basis of the scheduling algorithm presented in Section VII.

We define a data center as:

$$Data\_Center = \{node_i\}, i \in [1, N] \tag{1}$$

Where, $node_i$ indicates the ith node in the data center. There are N nodes in $Data\_Center$.

We define the node as:

$$node_i = (temp(t)) \tag{2}$$

Where, each node has a temperature-time profile that indicates the node's temperature value over time.

We define the jobs to be scheduled in the data center as:

$$job_k = (t_{start}, t_{exe}, node_{num}, f_{temp}(t)) \tag{3}$$

Where, t_{start} indicates the starting time of job; The job needs $node_{num}$ processors and lasts t_{exe}; $f_{temp}(t)$ is a function that indicates temperature rise caused by the execution of the job based on the execution time of the job.

$$(job_k) \xrightarrow{map} (node_i) \tag{4}$$

The task scheduling in the data center is actually the mapping of job_k to $node_i$, and the starting time t_{start} of job_k is specified when the mapping is conducted. As a job runs on the node, it will cause $\Delta Temp$ temperature increase on the node.

$$\Delta Temp(job_k) = job_k.f_{temp}(job_k.t_{exe}) \tag{5}$$

Where, $\Delta Temp(job_k)$ can be calculated through the job_k's function f_{temp}.

Now we define our problem as : given a set of jobs, find an optimal schedule to minimize computing nodes' temperature deviation. We use the deviation because it has the potential to balance the overall temperature in the data centers and thus

decreasing the cost for air conditioning needs as pointed out in the introduction of this paper. Hence,

$$u(t) = \frac{\sum_{i=1}^{N}(node_i.temp(t))}{N} \quad (6)$$

$$\sigma(t) = \sqrt{\frac{\sum_{i=1}^{N}(node_i.temp(t) - u(t))^2}{N}} \quad (7)$$

As defined in equation 6 and 7, we use standard deviation as our metric for measuring temperature distribution. The less the value of standard deviation is, the more even temperature distribution is.

VII. Thermal-Aware Task Scheduling Algorithm

Next we define a thermal-aware task scheduling algorithm.

Algorithm 1 Task Scheduling Algorithm

1: FOR $i = 1$ TO N DO
2: Initiate $node_i.temp(t)$
3: END FOR

4: $t = 0$
5: WHILE (¬ finished) DO
6: Schedule the set of upcoming tasks with Algorithm 2
7: $t = t + interval$
8: END WHILE

Algorithm 1 shows thermal-aware task scheduling pseudo code to highlight its functionality. The scheduling algorithm runs as a daemon in a data center with a predefined scheduling interval, $interval$. First the scheduler initiates each node's temperature-time profile. A node's temperature-time is the node's temperature value over time. Then the scheduler iterates to schedule tasks (line 6,7) with predefined interval value, $interval$ from starting time 0; In each scheduling round, the scheduler places upcoming tasks in the queue to the nodes with Algorithm 2. During the period of scheduling interval, incoming tasks arrive at the scheduler and will be scheduled at the next schedule round.

Algorithm 2 One-Round Scheduling Algorithm

1 Set the threshold for the node's temperature
2: Select the node which has the lowest "current" temperature
3: Sort jobs in descending order of the temperature rise they cause
4: FOR EACH job_k
5: Assign the job to the selected node
6: Update the node's temperature-time profile.
7: Select the node which has the lowest "current" temperature
8: END FOR EACH

In Algorithm 2, a scheduler assigns jobs to the nodes in a one-round scheduling. In line 2, the scheduler iterates each node's temperature profile and identifies the node which has the lowest "current" temperature. We demonstrate it in a small example. Figure 5 is the node's temperature-time profile. As shown in Figure 5, current time is t3 and the node's current temperate is $26°C$. But the scheduled jobs on the node will run till t4 and at the time the node's temperature is 30 $°C$. So the node's "current" temperature is 30 °C. In line 3, the scheduler sorts the jobs based on the temperature rise they will cause. From line 4 to line 8, the scheduler iterates to assign the unscheduled job to the node with lowest "current" temperature, update the node's temperature-time profile and then select the node which has the lowest "current" temperature.

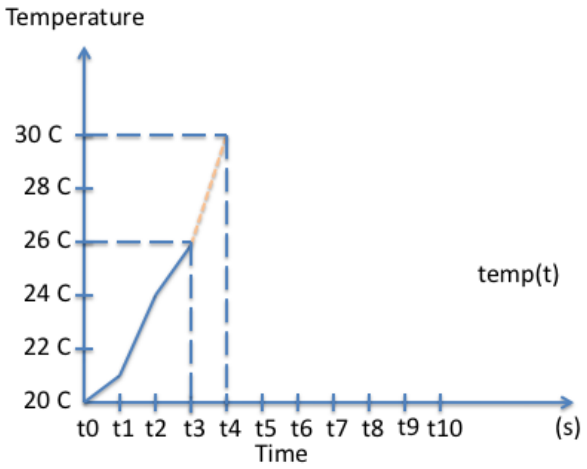

Fig. 5. a node's temperature-time profile

Let us calculate the running time for Algorithm 2. Assume there is L jobs to be scheduled. The running time of selecting the node with the lowest "current" temperature is $O(N)$. Sorting the jobs takes $O(L\lg L)$ time. Assigning the job to the selected node and updating node's temperature-time profile both take $O(1)$ time. Since the algorithm needs L times of selecting operations, the total running time for Algorithm 2 is $O(L * N) + O(L\lg L)$.

VIII. Conclusion and Future Work

To manage computing resources in a more energy efficient fashion, we present our thermal-aware resource management framework which has a layered architecture and can address a series of resource management related problems. Based on our observation that different types of task-temperature profiles and the relationship between temperature distribution and cooling energy consumption, we also developed a thermal-aware task scheduling as part of our framework with a goal of conserving cooling energy cost and increasing system reliability.

In the future, We would further our study and investigate other thermal characteristic of data centers. In addition, we would continue the development and improvement of our layered thermal-aware resource management framework.

REFERENCES

[1] "Report to Congress on Server and Data Center Energy Efficiency ." [Online]. Available: http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf

[2] C. Patel, R. Sharma, C. Bash, and A. Beitelmal, "Thermal considerations in cooling large scale high compute density data centers," in *Thermal and Thermomechanical Phenomena in Electronic Systems, 2002. ITHERM 2002. The Eighth Intersociety Conference on*, 2002, pp. 767–776.

[3] A. Beitelmal and C. Patel, "Thermo-fluids provisioning of a high performance high density data center," *Distributed and Parallel Databases*, vol. 21, no. 2, pp. 227–238, 2007.

[4] J. Moore, J. Chase, K. Farkas, and P. Ranganathan, "Data center workload monitoring, analysis, and emulation," in *Eighth Workshop on Computer Architecture Evaluation using Commercial Workloads*, 2005.

[5] J. Moore, J. Chase, and P. Ranganathan, "Weatherman: Automated, Online and Predictive Thermal Mapping and Management for Data Centers," in *IEEE International Conference on Autonomic Computing, 2006. ICAC'06*, 2006, pp. 155–164.

[6] C. Hsu and W. Feng, "A feasibility analysis of power awareness in commodity-based high-performance clusters," *Cluster 2005*, 2005.

[7] C.-H. Hsu and W. chun Feng, "A power-aware run-time system for high-performance computing," in *SC*, 2005, p. 1.

[8] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. L. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *SOSP*, 2003, pp. 164–177.

[9] L. M. Silva, J. Alonso, P. Silva, J. Torres, and A. Andrzejak, "Using virtualization to improve software rejuvenation," in *NCA*, 2007, pp. 33–44.

[10] K. H. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters," in *CCGRID*, 2007, pp. 541–548.

[11] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters," in *SC*, 2005, p. 34.

[12] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Thermal-aware task scheduling for data centers through minimizing heat recirculation," in *CLUSTER*, 2007, pp. 129–138.

[13] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, "Making scheduling "cool": Temperature-aware workload placement in data centers," in *USENIX Annual Technical Conference, General Track*, 2005, pp. 61–75.

[14] D. C. Vanderster, A. Baniasadi, and N. J. Dimopoulos, "Exploiting task temperature profiling in temperature-aware task scheduling for computational clusters," in *Asia-Pacific Computer Systems Architecture Conference*, 2007, pp. 175–185.

[15] H. Hamann, M. Schappert, M. Iyengar, T. van Kessel, and A. Claassen, "Methods and techniques for measuring and improving data center best practices," in *Thermal and Thermomechanical Phenomena in Electronic Systems, 2008. ITHERM 2008. 11th Intersociety Conference on*, 2008, pp. 1146–1152.

[16] D. Anderson, J. Dykes, and E. Riedel, "More than an interface - scsi vs. ata," in *FAST*, 2003.

[17] G. Cole, "Estimating drive reliability in desktop computers and consumer electronics systems," *Seagate Technology Paper TP*, vol. 338, 2000.

[18] A. M. Weil and D. G. Feitelson, "Utilization, predictability, workloads, and user runtime estimates in scheduling the ibm sp2 with backfilling," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 6, pp. 529–543, 2001.

[19] J. Heilgeist, T. Soddemann, and H. Richter, "Distributed meta-scheduling for grids," in *Mardi Gras Conference*, 2008, p. 25.

[20] G. von Laszewski, A. Younge, X. He, K. Mahinthakumar, and L. Wang, "Experiment and Workflow Management Using Cyberaide Shell," in *4th International Workshop on Workflow Systems in e-Science (WSES 09) in conjunction with 9th IEEE International Symposium on Cluster Computing and the Grid*. IEEE, 2009. [Online]. Available: http://cyberaide.googlecode.com/svn/trunk/papers/09-gridshell-ccgrid/vonLaszewski-ccgrid09-final.pdf

[21] G. von Laszewski, F. Wang, A. Younge, X. He, Z. Guo, and M. Pierce, "Cyberaide JavaScript: A JavaScript Commodity Grid Kit," in *GCE08 at SC'08*. Austin, TX: IEEE, Nov. 16 2008. [Online]. Available: http://cyberaide.googlecode.com/svn/trunk/papers/08-javascript/vonLaszewski-08-javascript.pdf

[22] R. Mukherjee, S. O. Memik, and G. Memik, "Temperature-aware resource allocation and binding in high-level synthesis," in *DAC*, 2005, pp. 196–201.

[23] R. Jayaseelan and T. Mitra, "Temperature aware task sequencing and voltage scaling," in *ICCAD*, 2008, pp. 618–623.

[24] A. K. Coskun, T. S. Rosing, and K. Whisnant, "Temperature aware task scheduling in mpsocs," in *DATE*, 2007, pp. 1659–1664.

[25] R. Jayaseelan and T. Mitra, "Temperature aware scheduling for embedded processors," in *VLSI Design*, 2009, pp. 541–546.

[26] A. Arani, "Online thermal-aware scheduling for multiple clock domain CMPs," in *2007 IEEE International SOC Conference*, 2007, pp. 137–140.

[27] J. Yang, X. Zhou, M. Chrobak, Y. Zhang, and L. Jin, "Dynamic thermal management through task scheduling," in *ISPASS*, 2008, pp. 191–201.

[28] "the Center of Computational Research." [Online]. Available: http://www.ccr.buffalo.edu/display/WEB/Home