Handbook on Energy-Aware and Green Computing

List of Figures

Contents

Chapter 1

Providing a Green Framework for Cloud Data Centers

Andrew J. Younge

Indiana University

Gregor von Laszewski

Indiana University

Lizhe Wang

Indiana University

Geoffrey C. Fox

Indiana University

1.1 Abstract

Cloud computing is emerging as the prominent new paradigm used in distributed systems today. One of the features that makes Clouds attractive is their ability to provide advanced services to users cost-effectively by taking advantage of the economies of scale. In this, large scale Cloud data centers have recently seen widespread deployment within both academia and industry. However, as the demand for such computational resources increases with the costs of using limited energy resources, there is a need to increase energy efficiency throughout the entire Cloud.

This manuscript presents a comprehensive system-level framework for identifying ways to integrate novel green computing concepts into tomorrow's Cloud systems. This framework includes components for advanced scheduling algorithms, virtual machine management, efficient virtual machine image design, service level agreements, and sophisticated data center designs. While the research activities discussed in each component improve overall system efficiency with little or no performance impact on an individual level, it's the green framework which provides the foundation for such research to build upon and have a lasting impact on the way in which data centers operate in the future.

1.2 Introduction

For years visionaries in computer science have predicted the advent of utility-based computing. This concept dates back to John McCarthy's vision stated at the MIT centennial celebrations in 1961.

> "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility... The computer utility could become the basis of a new and important industry."

Only recently has the hardware and software become available to support the concept of utility computing on a large scale.

The concepts inspired by the notion of utility computing have combined with the requirements and standards of Web 2.0 [11] to create Cloud computing [20, 36, 14]. Cloud computing is defined as, "A large-scale distributed computing paradigm that is driven by economies of

scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet." This concept of Cloud computing is important to Distributed Systems because it represents a true paradigm shift [47] within the entire IT infrastructure. Instead of adopting the in-house services, client-server model, and mainframes, Clouds push resources out into abstracted services hosted *en masse* by larger organizations. This concept of distributing resources is similar to many of the visions of the Internet itself, which is where the "Clouds" nomenclature originated, as many people depicted the internet as a big fluffy cloud one connects to.

While Cloud computing is changing IT infrastructure, it also has had a drastic impact on Distributed Systems itself. Gone are the IBM Mainframes of the 80's which dominated the enterprise landscape. While some mainframes still exist, they are used only for batch related processing tasks and are relatively unused for scientific applications as they are inefficient at Floating Point Operations. As such, they were replaced with Beowulf Clusters [59] of the 90's and 00's. The novelty of Supercomputing is that instead of just one large machine, many machines are connected together and used to achieve a common goal, thereby maximizing the overall speed of computation. Clusters represent a more commodity-based supercomputer, where off the shelf CPUs are used instead of the highly customized and expensive processors in Supercomputers. Supercomputers and Clusters are best suited for large scale applications such as particle physics, weather forecasting, climate research, molecular modeling, bioinformatics, and physical simulations, to name a few. These applications are often termed "Grand Challenge" applications and represent the majority of scientific calculations done on Supercomputing resources today. However, recently these scientific Clusters and Supercomputers are being subverted by the Cloud computing paradigm itself.

Many scientists are realizing the power that Clouds provide on demand, and are looking to harness the raw capacity for their own needs without addressing the daunting task of running their own Supercomputer. While many production-level Grid computing systems have looked to provide similar services for the past 10 years through services such as the Open Science Grid [55] and TeraGrid [21], the success of such Grid systems has been mixed. Currently we are on the cusp of a merger between the distributed Grid middleware and cutting-edge Cloud technologies within the realm of scientific computing. Recent projects such as the NSF FutureGrid Project [2] and the DOE Magellan Project [3] aim to leverage the advances of Grid computing with the added advantages of Clouds (discussed in detail in Chapter 2). Yet these projects

are at a research-only stage and in their infancy; the success of Cloud computing within the scientific community is still yet to be determined.

1.2.1 Motivation

As new distributed computing technologies like Clouds become increasingly popular, the dependence on power also increases. Currently it is estimated that data centers consume 0.5 percent of the world's total electricity usage [32]. If current demand continues, it is projected to quadruple by 2020. In 2005, the total energy consumption for servers and their cooling units was projected at 1.2% the total U.S. energy consumption and doubling every 5 years [45, 4]. The majority of the energy used in today's society is generated from fossil fuels which produce harmful CO_2 emissions. Therefore, it is imperative to enhance the efficiency and potential sustainability of large data centers.

One of the fundamental aspects of virtualization technologies employed in Cloud environments is resource consolidation and management. Using hypervisors within a cluster environment allows for a number of standalone physical machines to be consolidated to a virtualized environment, thereby requiring less physical resources than ever before. While this improves the situation, it often is inadequate. Large Cloud deployments require thousands of physical machines and megawatts of power. Therefore, there is a need to create an efficient Cloud computing system that utilizes the strengths of the Cloud while minimizing its energy and environmental footprint.

In order to correctly and completely unify a Green aspect to the next generation of Distributed Systems, a set of guidelines is needed. These guidelines must represent a path of sustainable development that can be integrated into data center construction and management as a whole. While the framework provided in this paper represents many promising ways to reduce power consumption, true sustainable development also depends on finding a renewable and reliable energy source for the data center itself. When combined, many of today's limits in the size of data centers will begin to deteriorate.

1.2.1.1 Economic Costs

Many supercomputers and large scale data centers are operated at a power envelope on the scale of Megawatts or even tens of Megawatts. Throughout the past, many of these resources were operated at an institutional level, where such power concerns were not dealt with directly by those that operate or administer such a data center. However recently these energy requirements have grown so large that institutions

are starting to feel the economic burden. As such, the time of the energy "blank check" is over, and energy efficiency for the sake of economic stability is now one of the top concerns across all institutions and resource providers alike.

This recent trend is not only being realized in small institutions trying to minimize costs during an economic recession, but also in large national-scale laboratories operating multi-Megawatt facilities. Many Supercomputing and chip manufacturers are now focused on performance per dollar, not just power.

1.2.1.2 Environmental Concerns

While the economic costs of operating the world's data centers can be extremely staggering as a whole, there is also another important consideration; the Environment. As shown in figure 1.1, nearly 70% of the U.S.'s energy comes from non-renewable fossil fuels [10]. As current data center energy consumption is estimated at 2.4%, the overall CO_2 emissions due to the data centers represents sobering reality to the environmental impact the industry has created.

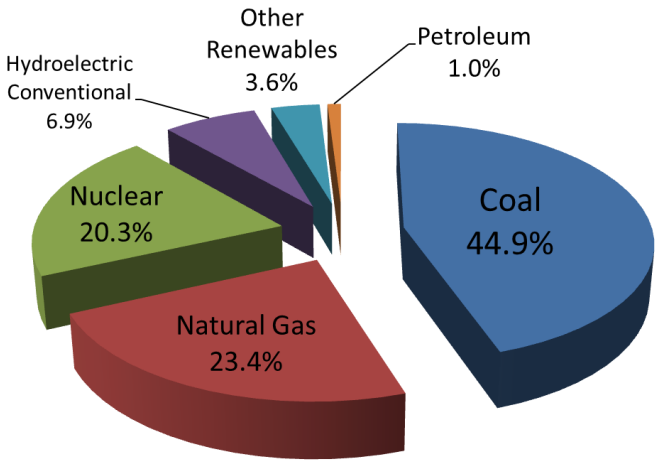

FIGURE 1.1: Sources of Energy in the U.S in 2009

To properly address the sustainability of data centers, it would make sense to focus primarily on the sources of energy. However achieving the goal of producing 70% of the U.S.'s current power generation from renewable energy sources is a task that will take at least a generation, if not many generations. In the meantime, it is imperative to focus on

the data centers themselves to improve efficiency, not only for economic reasons, but also for the environment.

Therefore, improving the energy efficiency within data centers does not look to revolutionize their sustainability, but instead to improve upon an already existing infrastructure. This work is not meant to remove the need for renewable resources, but instead to try to alleviate the burdening energy demands during this long transition period away form fossil fuels. As defined by Thomas Seager's Sustainability Spectrum [57], the work introduced in this manuscript lies within the *Reliability* phase of sustainability. Nevertheless, it is important to make what energy efficiency enhancements we can in order to minimize the global climate impact.

FIGURE 1.2: Seager's Sustainability Spectrum

Cloud computing also makes sense environmentally from a whole inter-level viewpoint. Much of what Cloud computing represents is the consolidation of resources from a intra standpoint to an inter standpoint, thereby pushing data and services from individual sites to a more centralized system. While there are numerous scaling and privacy issues Cloud computing must address, the advantages are clear from an environmental perspective. Typically each individual data center site is relatively small in size and will therefore have more traditional means of cooling. These small data center cooling systems are often just large air conditioning setups that result in a very high data center Power Usage Effectiveness (PUE) [17], resulting in more energy used overall.

Many commercial Cloud deployments provide a large scale data center operations. This allows each site to be take advantage of the economies of scale [60] which are not available to smaller-sized data centers. These larger Cloud centers will be able to implement advanced cooling solutions which keep the PUE low, such as advanced chiller towers or advanced water-cooling setups. Currently, Google has implemented a number of these large data centers and is achieving a PUE value between 1.1 and 1.3, compared to the national average of 2.1 [37]. If the IT industry continues to shift the overall computing resources from many

small data centers to a smaller number of larger data centers, the cooling savings alone will have a drastic impact on the environment, just by taking advantage of the basic principals of scaling.

1.3 Related Research

In order to accurately depict the research presented in this article, the topics within Cloud computing, Grid computing, Clusters and Green IT will be reviewed.

1.3.1 Cloud Computing

Cloud computing is one of the most explosively expanding technologies in the computing industry today. However it is important to understand where it came from, in order to figure out where it will be heading in the future. While there is no clear cut evolutionary path to Clouds, many believe the concepts originate from two specific areas: Grid Computing and Web 2.0.

Grid computing [35, 34], in its practical form, represents the concept of connecting two or more spatially and administratively diverse clusters or supercomputers together in a federating manner. The term "the Grid" was coined in the mid 1990's to represent a large distributed systems infrastructure for advanced scientific and engineering computing problems. Grids aim to enable applications to harness the full potential of resources through coordinated and controlled resource sharing by scalable virtual organizations. While not all of these concepts carry over to the Cloud, the control, federation, and dynamic sharing of resources is conceptually the same as in the Grid. This is outlined by [36], as Grids and Clouds are compared at an abstract level and many concepts are remarkably similar. From a scientific perspective, the goals of Clouds and Grids are also similar. Both systems attempt to provide large amounts of computing power by leveraging a multitude of sites running diverse applications concurrently in symphony. The only significant differences between Grids and Clouds exist in the implementation details, and the reproductions of them, as outlined later in this section.

The other major component, Web 2.0, is also a relatively new concept in the history of Computer Science. The term Web 2.0 was originally coined in 1999 in a futuristic prediction by Dracy DiNucci [26]: "The Web we know now, which loads into a browser window in essentially static screenfulls, is only an embryo of the Web to come. The first glimmerings

of Web 2.0 are beginning to appear, and we are just starting to see how that embryo might develop. The Web will be understood not as screenfuls of text and graphics but as a transport mechanism, the ether through which interactivity happens. It will [...] appear on your computer screen, [...] on your TV set [...] your car dashboard [...] your cell phone [...] hand-held game machines [...] maybe even your microwave oven." Her vision began to form, as illustrated in 2004 by the O'Riley Web 2.0 conference, and since then the term has been a pivotal buzz word among the internet. While many definitions have been provided, Web 2.0 really represents the transition from static HTML to harnessing the Internet and the Web as a platform in of itself.

Web 2.0 provides multiple levels of application services to users across the Internet. In essence, the web becomes an application suite for users. Data is outsourced to wherever it is wanted, and the users have total control over what they interact with, and spread accordingly. This requires extensive, dynamic and scalable hosting resources for these applications. This demand provides the user-base for much of the commercial Cloud computing industry today. Web 2.0 software requires abstracted resources to be allocated and relinquished on the fly, depending on the Web's traffic and service usage at each site. Furthermore, Web 2.0 brought Web Services standards [13] and the Service Oriented Architecture (SOA) [46] which outline the interaction between users and cyberinfrastructure. In summary, Web 2.0 defined the interaction standards and user base, and Grid computing defined the underlying infrastructure capabilities.

A Cloud computing implementation typically enables users to migrate their data and computation to a remote location with minimal impact on system performance [**?**]. This provides a number of benefits which could not otherwise be realized. These benefits include:

- *Scalable* - Clouds are designed to deliver as much computing power as any user needs. While in practice the underlying infrastructure is not infinite, the cloud resources are projected to ease the developer's dependence on any specific hardware.
- *Quality of Service (QoS)* - Unlike standard data centers and advanced computing resources, a well-designed Cloud can project a much higher QoS than traditionally possible. This is due to the lack of dependence on specific hardware, so any physical machine failures can be mitigated without the prerequisite user awareness.
- *Specialized Environment* - Within a Cloud, the user can utilize customized tools and services to meet their needs. This can be to utilize the latest library, toolkit, or to support legacy code within new infrastructure.

- *Cost Effective* - Users finds only the hardware required for each project. This reduces the risk for institutions potentially want build a scalable system, thus providing greater flexibility, since the user is only paying for needed infrastructure while maintaining the option to increase services as needed in the future.

- *Simplified Interface* - Whether using a specific application, a set of tools or Web services, Clouds provide access to a potentially vast amount of computing resources in an easy and user-centric way. We have investigated such an interface within Grid systems through the use of the Cyberaide project [67, 64].

Many of the features noted above define what Cloud computing can be from a user perspective. However, Cloud computing in its physical form has many different meanings and forms. Since Clouds are defined by the services they provide and not by applications, an integrated as-a-service paradigm has been defined to illustrate the various levels within a typical Cloud, as in Figure 1.3.

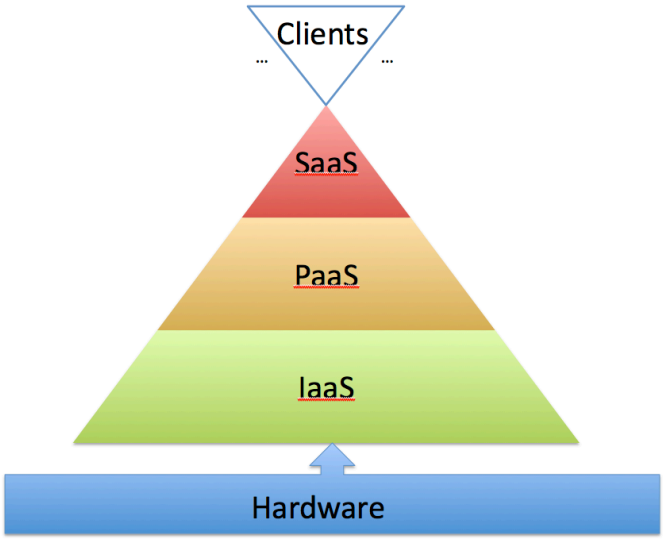

FIGURE 1.3: View of the Layers within a Cloud Infrastructure

- *Clients* - A client interacts with a Cloud through a predefined, thin layer of abstraction. This layer is responsible for communicating the user requests and displaying data returned in a way that is simple and intuitive for the user. Examples include a Web Browser or a thin client application.

- *Software-as-a-Service (SaaS)* - A framework for providing applications or software deployed on the Internet packaged as a unique service for users to consume. By doing so, the burden of running a local application directly on the client's machine is removed. Instead all the application logic and data is managed centrally and to the user through a browser or thin client. Examples include Google Docs, Facebook, or Pandora.

- *Platform-as-a-Service (PaaS)* - A framework for providing a unique computing platform or software stack for applications and services to be developed on. The goal of PaaS is to alleviate many of the burdens of developing complex, scalable software by proving a programming paradigm and tools that make service development and integration a tractable task for many. Examples include Microsoft Azure and Google App Engine.

- *Infrastructure-as-a-Service (IaaS)* - A framework for providing entire computing resources through a service. This typically represents virtualized Operating Systems, thereby masking the underlying complexity details of the physical infrastructure. This allows users to rent or buy computing resources on demand for their own use without needing to operate or manage physical infrastructure. Examples include Amazon EC2, Eucalyptus, and Nimbus.

- *Physical Hardware* - The underlying set of physical machines and IT equipment that host the various levels of service. These are typically managed at a large scale using virtualization technologies which provide the QoS users expect. This is the basis for all computing infrastructure.

When all of these layers are combined, a dynamic software stack is created to focus on large scale deployment of services to users.

1.3.1.1 Virtualization

There are a number of underlying technologies, services, and infrastructure-level configurations that make Cloud computing possible. One of the most important technologies is the use of virtualization [15, **?**]. Virtualization is a way to abstract the hardware and system resources from a operating system. This is typically performed within a Cloud environment across a large set of servers using a Hypervisor or Virtual Machine Monitor (VMM) which lies in between the hardware and the Operating System (OS). From here, one or more virtualized OSs can be started concurrently as seen in Figure 1.4, leading to one of the key advantages of Cloud computing. This, along with the advent of multi-core

processing capabilities, allows for a consolidation of resources within any data center. It is the Cloud's job to exploit this capability to its maximum potential while still maintaining a given QoS.

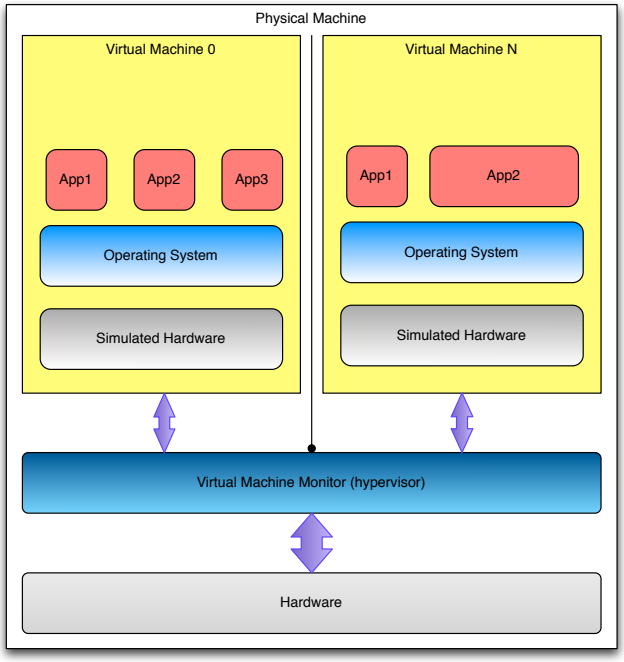

FIGURE 1.4: Virtual Machine Abstraction

Virtualization is not specific to Cloud computing. IBM originally pioneered the concept in the 1960's with the M44/44X systems. It has only recently been reintroduced for general use on x86 platforms. Today there are a number of Clouds that offer IaaS. The Amazon Elastic Compute Cloud (EC2) [12], is probably the most popular of which and is used extensively in the IT industry. Eucalyptus [54] is becoming popular in both the scientific and industry communities. It provides the same inter-

face as EC2 and allows users to build an EC2-like cloud using their own internal resources. Other scientific Cloud specific projects exist such as OpenNebula[31], In-VIGO [9], and Cluster-on-Demand [23]. They provide their own interpretation of private Cloud services within a data center. Using a Cloud deployment overlaid on a Grid computing system has been explored by the Nimbus project [44] with the Globus Toolkit [33]. All of these clouds leverage the power of virtualization to create an enhanced data center. The virtualization technique of choice for these Open platforms has typically been the Xen hypervisor, however more recently VMWare and the Kernel-based Virtual Machine (KVM) have become commonplace.

1.3.1.2 Workload Scheduling

While virtualization provides many key advancements, this technology alone is not sufficient. Rather, a collective scheduling and management for virtual machines is required to piece together a working Cloud. Let us consider a typical usage for a Cloud data center that is used in part to provide computational power for the Large Hadron Collider at CERN [22], a global collaboration from more than 2000 scientists of 182 institutes in 38 nations. Such a system would have a small number of experiments to run. Each experiment would require a very large number of jobs to complete the computation needed for the analysis. Examples of such experiments are the ATLAS [50] and CMS [1] projects, which (combined) require Petaflops of computing power on a daily basis. Each job of an experiment is unique, but the application runs are often the same.

Therefore, virtual machines are deployed to execute incoming jobs. There is a file server which provides virtual machine templates. All typical jobs are preconfigured in virtual machine templates. When a job arrives at the head node of the cluster, a correspondent virtual machine is dynamically started on acertain compute node within the cluster to execute the job (see Figure 1.5).

While this is an abstract solution, it is important to keep in mind that these virtual machines create an overhead when compared to running on "bare metal." Current research estimates this the overhead for CPU bound operations at 1 to 15% depending on the hypervisor, however more detailed studies are needed to better understand this overhead. While the hypervisor introduces overhead, so does the actual VM image being used. Therefore, it is clear that slimming down the images could yield an increase in overall system efficiency. This provides the motivation for the minimal Virtual Machine image design discussed in Section **??**.

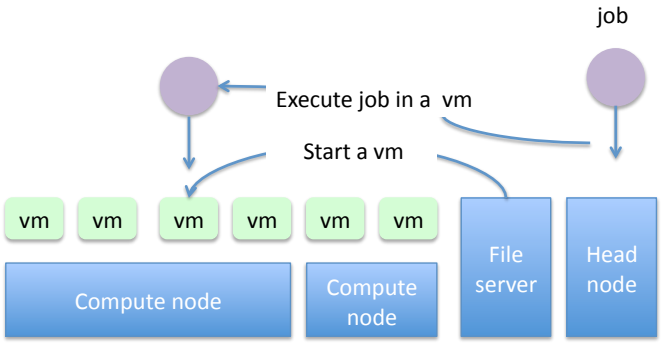

FIGURE 1.5: Example of Job Scheduling in a Virtualized Environment

1.3.2 Green Information Technology

The past few years have seen an increase in research on developing efficient large computational resources. Supercomputer performance has doubled more than 3000 times in the past 15 to 20 years, the performance per watt has increased 300 fold while performance per square foot has only doubled 65 times [24] in the same period of time. This lag in Moore's Law over such an extended period of time in computing history has created the need for more efficient management and consolidation of data centers. This can be seen in figure 1.6 [68].

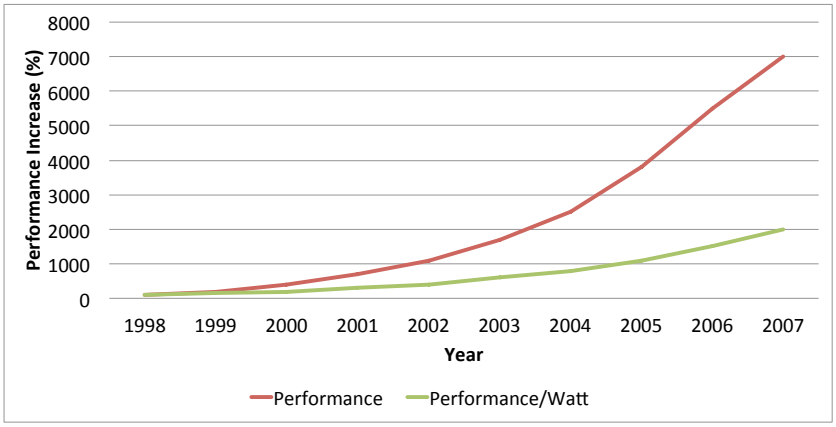

FIGURE 1.6: Performance increases much faster than performance per watt of energy consumed.

Much of the recent work in Green computing focuses on Super-

computers and Cluster systems. Currently the fastest Supercomputer in the world is the IBM Roadrunner at Los Alamos National Laboratory [16, 27], which was fundamentally designed for power efficiency. However, Roadrunner consumes several Megawatts of power [58] (not including cooling) and costs millions of dollars to operate every year. The second fastest Supercomputer is Jaguar at Oak Ridge National Laboratory. While Jaguar also has a number of power saving features developed by Sandia, Oak Ridge and Cray [48] such as advanced power metering at the CPU level, 480 volt power supplies, and an advanced cooling system developed by Cray. However, the system as a whole still consumes almost 7 Megawatts of power.

1.3.2.1 Dynamic Voltage and Frequency Scaling

One technique being explored is the use of Dynamic Voltage and Frequency Scaling (DVFS) within Clusters and Supercomputers [42, 43]. By using DVFS one can lower the operating frequency and voltage, which results in decreased power consumption of a given computing resource considerably. High-end computing communities such as cluster computing and supercomputing in large data centers, have applied DVFS techniques to reduce power consumption and achieve high reliability and availability [38, 29, 28, **?**]. A power-aware cluster is defined as a compute cluster where compute nodes support multiple power/performance modes, for example, processors with frequencies that can be turned up or down. This technique was originally used in portable and laptop systems to conserve battery power, and has since migrated to the latest server chipsets. Current technologies exist within the CPU market such as Intel's SpeedStep and AMD's PowerNow! technologies. These dynamically raise and lower both frequency and CPU voltage using ACPI P-states [19]. In [30], DVFS techniques are used to scale down the frequency by 400Mhz while sustaining only a 5% performance loss, resulting in a 20% reduction in power.

A power-aware cluster supports multiple power and performance modes, allowing for the creation of an efficient scheduling system that minimizes power consumption of a system while attempting to maximize performance. The scheduler performs the energy-performance trade-off within a cluster. Combining various power efficiency techniques for data centers with the advanced feature set of Clouds could yield drastic results; however, currently no such system exists. This is the premise for much of the work described in Section 1.5.1.

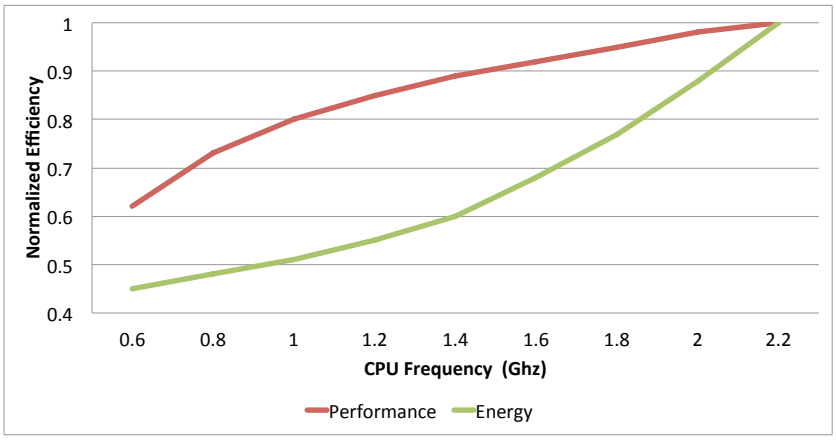

FIGURE 1.7: Possible energy to performance trade-off.

1.3.2.2 Cooling Systems

While there have been numerous reports focused on the Green computing aspects of DVFS scheduling, there is also the other side of the coin: cooling solutions. The typical data center cooling layout consists of rows or rack with under floor cold air distribution. These rows are typically laid back to back, where the hot exhaust air point to each other, such as outlined in Figure 1.8. This forms "cold aisles", where cold air comes from the computer room air conditioning (CRAC) unit under the raised flooring system, and "hot aisles" where the hot air exhausts back around to the CRAC unit.

While the hot & cold aisle cooling approach to a data center improves efficiency compared to a uniform setup, it is far from ideal. With hot and cold air cooling, thermal imbalances are common and interfere with the cooling operations. These imbalances, or "hotspots" can exceed the normal operating temperatures of the servers. This can eventually lead to decreased performance and a high rate of hardware failure. The Arrhenius time-to-fail model [39] describes this, stating that every 10°C increase of temperature leads to a doubling of the system failure rate. Therefore, objectives of thermal aware workload scheduling are to reduce both the maximum temperature for all compute nodes and the imbalance of the thermal distribution in a data center. In a data center, the thermal distribution and computer node temperatures can be obtained by deploying ambient temperature sensors, on-board sensors [63, 61], and with software management architectures like the Data Center Observatory [40].

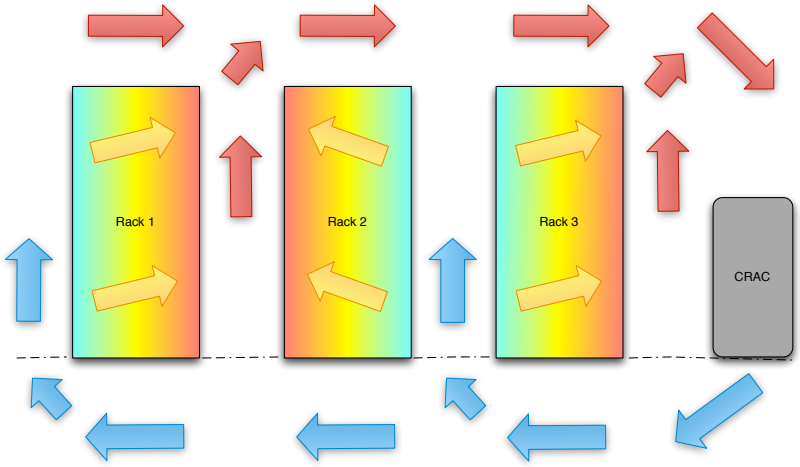

FIGURE 1.8: Data center cooling system

Many recent data center cooling designs have been created to ease the cooling needs. Many of these designs try to break the standard raised floor hot-cold isle designs. The first mainstream alternative design was the Sun Blackbox project [5], now known as the Modular Data Center. As seen in Figure 1.9, the design consists of using a trucking cargo holder as a modular server room. By having such a finely controlled system, the overall cooling requirements can be reduced substantially. This was realized by Google, who has since deployed large warehouses of these containers or "pods", reaching a sustained PUE of under 1.3 [8].

1.4 A Green Cloud

There is a pressing need for an efficient yet scalable Cloud computing system. This is driven by the ever-increasing demand for greater computational power countered by the continual rise in use expenditures, both economical and environmental. Both business and institutions will be required to meet these needs in a rapidly changing environment in order to survive and flourish in the long term. As such, a systems level design guideline is needed to outline areas of exploration to change efficient Cloud data centers from fiction into reality.

FIGURE 1.9: Sun Modular Datacenter

1.4.1 Framework

This manuscript presents a novel Green computing framework which is applied to the Cloud paradigm in order to meet the goal of reducing power consumption, as introduced in [69]. The framework is meant to define efficient computing resource management and Green computing technologies can be adapted and applied to Cloud systems, but many of the concepts are applicable to various other data center usages. The focus here is on Cloud computing data centers for several reasons. First, the Cloud is a relatively new concept, one that is able to accept input and be defined most readily. Second, it is a technology that is on the rise with exponential growth, thereby yielding significant gains. Finally, Cloud computing's underlying technologies finally allow for the flexibility and precision needed to add in efficiency that makes a difference.

Figure 1.10 illustrates a comprehensive *Green Cloud framework* for maximizing performance per watt within a Cloud. Within the framework, there are two major areas which can lead to widespread improvements in efficiency: Virtualization and machine system level designs. The framework tries to outline ways to expand upon the baseline functioning of virtual machines in a cloud environment. This is first done with deriving a more efficient scheduling system for VMs. The Scheduling section addresses the placement of VMs within the Cloud infrastructure in a way in which maximizes the work capacity while simultaneously mini-

mizing the operating costs of the Cloud itself. This is typically achieved by optimizing either power of the server equipment itself or the overall temperature within the data center. Due to the inherent disposability and mobility of stateless VMs within a semi-homogeneous data center, we can leverage the ability to move and manage the VMs to further improve efficiency. Furthermore, intelligent image management can attempt to control and manipulate the size and placement of VM images in various ways to conserve power and reduce the size of images. Through this, the design of the virtual machine images can also lead to a drastic power savings, if architected correctly.

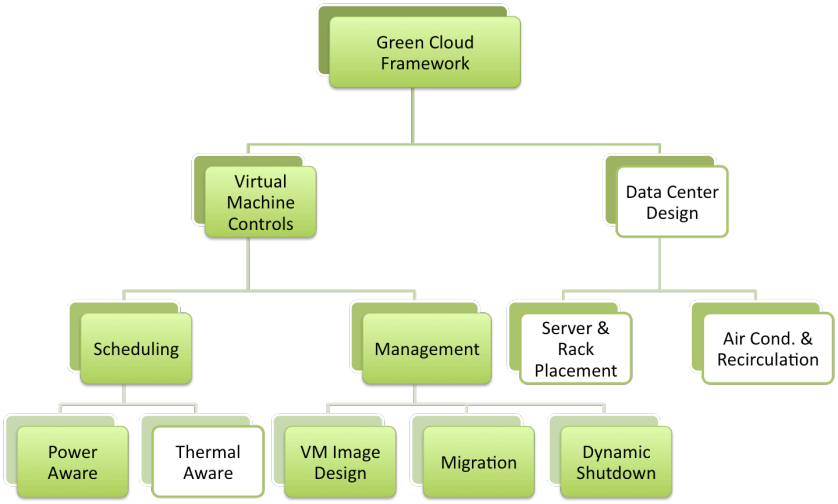

FIGURE 1.10: Green Cloud Framework. Shaded items represent topics discussed in this paper.

While these operational and runtime chances can have a drastic impact, however more static data center level design decisions are also vital for improving energy efficiency. Using more efficient air conditioning units, employing exterior "free" cooling, using completely separated hot and cold isles, or simply picking more efficient power supplies for the servers can lead to incremental but substantial improvements. These best practices can be further enhanced with more radical design, such as a cylindrical or spiral data center design that brings cool air form the outside in, and exhausts it up through a center chimney and out the top. Also, the excess heat energy of data centers should not go to waste. Instead, the exhaust could be used to heat water or provide ambient heat for surrounding workspaces. Although the potential is great, combining

the factors together in such a unified framework and deploying it to a large scale Cloud poses many challenges.

While the last technique may be outside the scope of this manuscript, the integrated components of the Green Cloud framework in Figure 1.10 provide a sustainable development platform which show the largest potential impact factor to drastically reduce power requirements within a Cloud data center. This framework is not meant to be a solution in and of itself, but rather, a set of paths, or guidelines, to reach a solution. As such, only a subset of these tasks are addressed head-on throughout the remaining sections. The hope is that this work will lead to a growing amount of research in this new field to address the growing energy demands within our newest and greatest data centers.

1.5 Scheduling & Management

While Supercomputer and Cluster scheduling algorithms are designed to schedule individual jobs and not virtual machines, some of the concepts can be translated to the Cloud. We have already conducted such research in [65]. In many service oriented scientific Cloud architectures, new VMs are created to perform some work. The idea is similar to sand boxing work within a specialized environment.

1.5.1 DVFS-enabled Scheduling

As outlined in Section 1.3.2.1, DVFS has proven to be a valuable tool when scheduling work within a cluster. When looking to create a DVFS scheduling system for a Cloud data center, there are a few rules of thumb to build a scheduling algorithm which schedules virtual machines in a cluster while minimizing the power consumption:

1. Minimize the processor supply voltage by scaling down the processor frequency.
2. Schedule virtual machines to processing elements with low voltages and try not to scale PEs to high voltages.

Based on the performance model defined above, Rule 1 is obvious as the power consumption could be reduced when supplied voltages are minimized. Then Rule 2 is applied: to schedule virtual machines to processing elements with low voltages and try not to operate PEs with high voltages to support virtual machines.

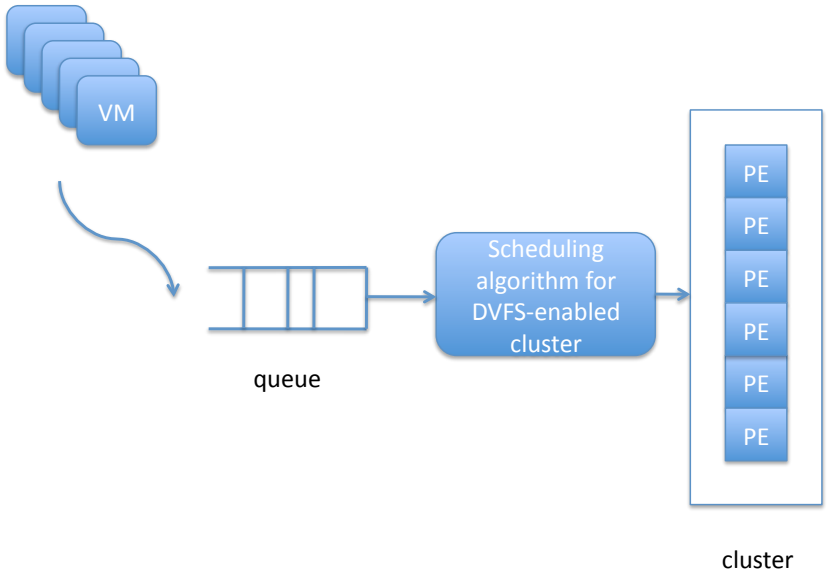

FIGURE 1.11: Working scenario of a DVFS-enabled cluster scheduling

As shown in Figure 1.11, incoming virtual machine requests arrive at the cluster and are plased in a sorted queue. This system has been modeled, created and described in [66], but this section will explain in detail this scheduling mechanism.

The algorithm in Appendix **??** shows the scheduling algorithm for virtual machines in a DVFS-enabled cluster. A scheduling algorithm runs as a daemon in a cluster with a predefined schedule interval, *INTERVAL*. During the period of scheduling interval, incoming virtual machines arrive at the scheduler and will be scheduled at the next schedule round. The algorithm in Appendix **??** is used to schedule incoming virtual machine requests in a certain schedule round defined by the previous algorithm. For each virtual machine (VM), the algorithm checks the processing element (PE) operating point set from low voltage level to high voltage level. The PE with lowest possible voltage level is found; if this PE can fulfill the virtual machine requirement, the VM can be scheduled on this PE. If no PE can schedule the VM, a PE must be selected to operate with higher voltage. This is done using the processor with the highest potential processor speed, which is then incremented to the lowest speed that fulfills the VM requirement, and the VM is then scheduled on that PE. After one schedule interval passes, some virtual machines may have finished their execution; thus the algorithm in Ap-

pendix **??** attempts to reduce a number of PE's supply voltages if they are not fully utilized.

1.5.2 Power-Aware Multi-core Scheduling

Currently, there are two competing types of Green scheduling systems for Supercomputers; power-aware and thermal-aware scheduling. In thermal-aware scheduling [62], jobs are scheduled in a manner that minimizes the overall data center temperature. The goal is not always to conserve the energy used to the servers, but instead to reduce the energy needed to operate the data center cooling systems. In power-aware scheduling [43], jobs are scheduled to nodes in such a way to minimize the server's total power. The largest operating cost incurred in a Cloud data center is in operating the servers. As such, we concentrate on power-aware scheduling in this paper.

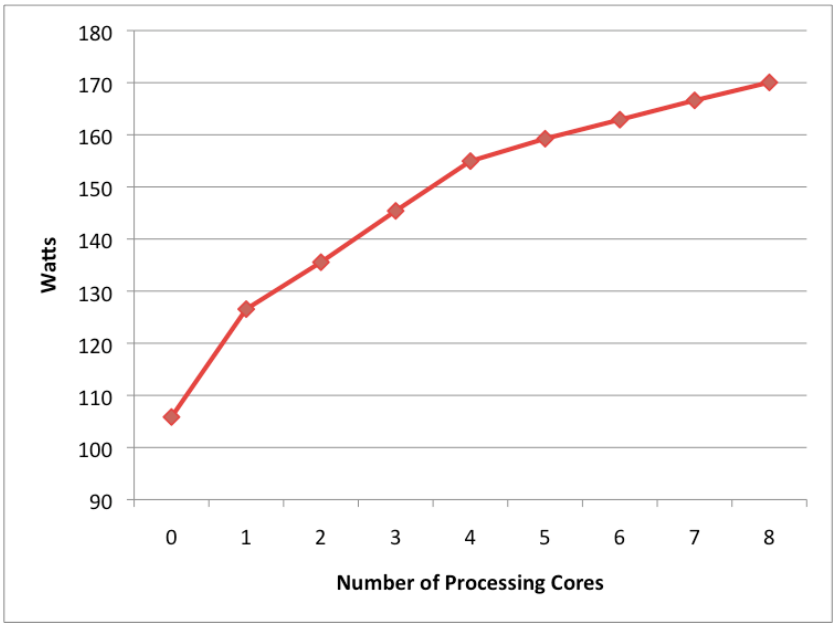

FIGURE 1.12: Power consumption curve of an Intel Core i7 920 CPU

Figure 1.12 illustrates the motivation behind power-aware VM scheduling. This graphic documents our recent research findings regarding watts of energy consumed verses the number of processing cores in use. The power consumption curve illustrates that as the number of processing cores increases, the amount of energy used does not increase pro-

portionally. When evaluating using only one processing core, the change in power consumption incurred by using a second processing core is over 20 watts. The change from 7 processing cores to all 8 processing cores results in an increase of only 3.5 watts.

The impact of this finding is substantial. In a normal round robin VM scheduling system like the one in Eucalyptus, the load of VMs is distributed evenly to all servers within the data center. While this may be a fair scheduler, in practice it is very inefficient. The result is that each time the scheduler distributes VMs to a processor, the power consumption increases by its greatest potential. In contrast, this research demonstrates that if the scheduler distributes the VMs with the intent to fully utilize all processing cores within each node, the power consumption is decreased dramatically. Therefore, there is a large need for an advanced scheduling algorithm which incorporates the findings in Figure 1.12. To meet this need we propose Algorithm 1, a new greedy-based algorithm to minimise power consumption.

Algorithm 1 Power based scheduling of VMs

```
FOR i = 1 TO i ≤ |pool| DO
  pe_i = num cores in pool_i
END FOR

WHILE (true)
  FOR i = 1 TO i ≤ |queue| DO
    vm = queue_i
      FOR j = 1 TO j ≤ |pool| DO
        IF pe_j ≥ 1 THEN
          IF check capacity vm on pe_j THEN
            schedule vm on pe_j
            pe_j − 1
          END IF
      END IF
    END FOR
  END FOR
  wait for interval t
END WHILE
```

Algorithm 1 is a VM scheduling algorithm that minimizes power consumption within the data center. This task is accomplished by continually loading each node with as many VMs as possible. In Algorithm 1 the pool acts as a collection of nodes and remains static after initialization. While not in the algorithm, the pool can be initialized by

a priority based evaluations system to either maximize performance or further minimize power consumption. At a specified interval t the algorithm runs through each waiting VM in the queue to be scheduled. The first node in the priority pool is selected and evaluated to see if it has enough virtual cores and capacity available for the new VM. If it does, it is scheduled and the pe_i is decremented by one; and this processes is continued until the VM queue is empty. When a VM finishes its execution and terminates, it reports it back the scheduler and pe_i is increased by one to signify a core of machine i is freed.

1.5.3 Virtual Machine Management

Another key aspect of a Green Cloud framework is virtual machine image management. By using virtualization technologies within the Cloud, a number of new techniques become possible. Idle physical machines in a Cloud can be dynamically shut down and restarted to conserve energy during low load situations. A similar concept was achieved in Grid systems though the use of the Condor Glide-In [56, 49] add-on to Condor, which dynamically adds and removes machines from the resource pool. This concept of shutting down unused machines will have no effect on power consumption during peak load as all machines will be running. However in practice Clouds almost never run at full capacity as this could result in a degradation of the QoS. Therefore by design, fast dynamic shut down and startup of physical machines could have a drastic impact on power consumption, depending on the load of the Cloud at any given point in time.

The use of live migration features within Cloud systems [25] is a recent concept. Live migration is presently used for proactive fault tolerance by seamlessly moving VMs away from failing hardware to stable hardware without the user noticing a change [53] in a virtualized environment. Live migration can be applied to Green computing in order to migrate away machines. VMs can be shifted from low load to medium load servers when needed. Low load servers are subsequently shutdown when all VMs have migrated away, thus conserving the energy required to run the low load idle servers. When using live migration, the user is completely unaware of a change and there is only a 60 to 300ms delay, which is acceptable by most standards.

This process of dynamically allocating and deallocating physical machines is complimentary to our scheduling system outlines in Algorithm 1. As the scheduling algorithm executes, it will leave a number of machines idling, potentially for long periods of time. At this point these machines shut down these machines when they are unused. When load increases, we use Wake on LAN (WOL) to start them back up. This

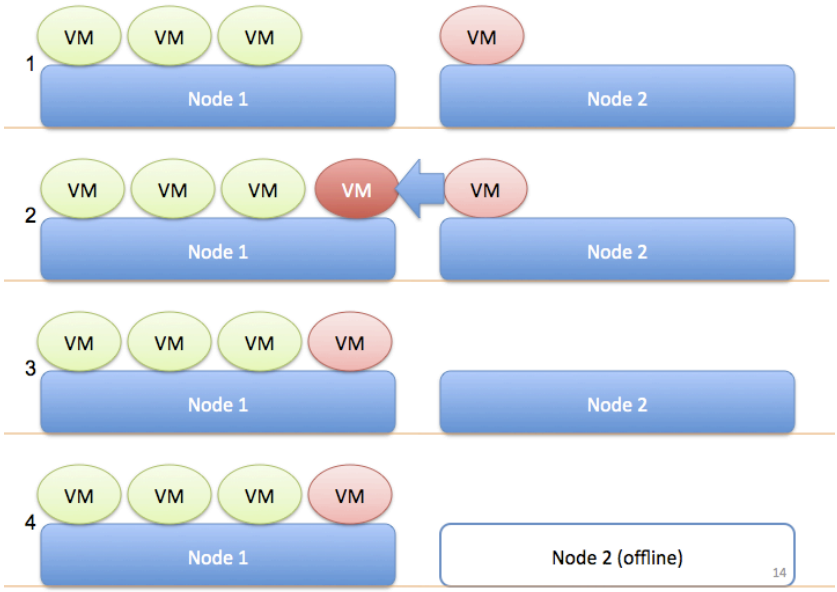

FIGURE 1.13: Virtual Machine management dynamic shutdown technique

control can be easily monitored and implemented as a daemon running on the Cloud head node or scheduler. An illustration of this is presented in Figure 1.13.

1.5.4 Performance Analysis

OpenNebula [31] is an open source distributed virtual machine manager for dynamic allocation of virtual machines in a resource pool. The OpenNebula core components illustrated in Figure 1.14 accept user requirements via the OpenNebula interface, and then place virtual machines in compute nodes within the cluster.

The OpenNebula scheduler is an independent component that provides policies for virtual machine placement. The OpenNebula project was chosen because of this compartmentalized design as it allows for integration of our custom scheduling algorithm. The default scheduler provides a scheduling policy based on rank, which allocates compute resources for virtual machines. Scheduling algorithm 1 is implemented by modifying the OpenNebula scheduler to reflect the desired hypothesis that DVFS scheduling leads to a higher performance per unit of energy.

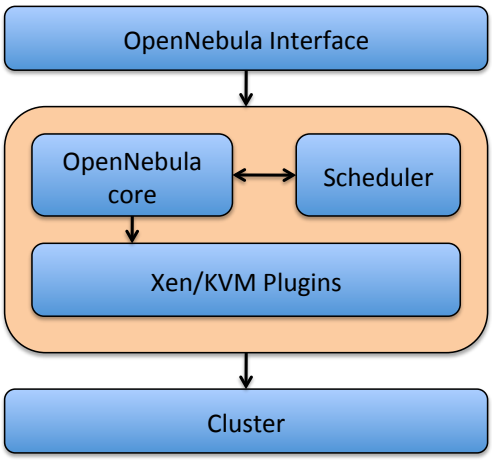

FIGURE 1.14: OpenNebula Software Architecture

In order to test our design, we created a two-node experimental multi-core cluster consisting of Intel Nehalem quad-core processors with Hyperthreading (providing 8 virtual cores). The Nehalem-based CPUs allow for each core to operate on its own independent P-state, thereby maximizing the frequency scaling flexibility. The compute nodes are installed with Ubuntu Server 8.10 with Xen 3.3.4-unstable. The head node consists of a Pentium 4 CPU installed with Ubuntu 8.10, OpenNebula 1.2 and a NFS server to allow compute nodes access to OpenNebula files and VM images. For this experiment, we schedule all virtual machines to the compute nodes and run the nBench [6] Linux Benchmark version 2.2.3 to approximate the system performance. The nBench application is an ideal choice as it is easily compiled in Linux, combines a number of different mathematical applications to evaluate performance, and it provides a comparable Integer and Floating Point Index that can used to evaluate overall system performance. The operating frequency of each core can be set to 1.6Hz, 1.86Ghz, 2.13GHz, 2.53GHz, or 2.66Ghz, giving the processor a frequency range of over 1.0Ghz.

Figure 1.15 shows the largest observed power consumption on a WattsUp power meter [7] during the execution of 2, 4, and 8 VMs at each frequency while computing the nBench Linux Benchmark. Here the benchmark effectively simulates a CPU-intensive job running within a VM and provides valuable information on the performance of each VM.

A number of things can be observed from Figure 1.15. First, while scheduling more virtual machines on a node raises power consumption, it seems to consume far less power than operating two separate nodes.

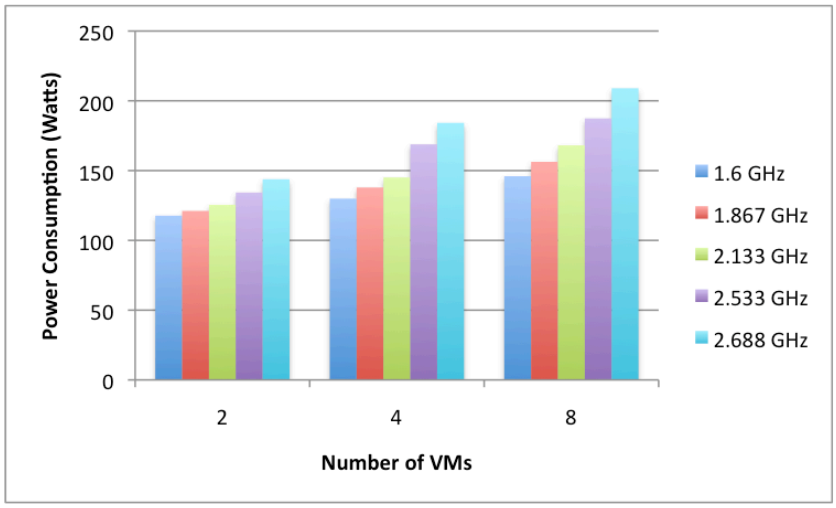

FIGURE 1.15: Power consumption variations for a Intel Nehalem Quad-core processor

Therefore, it seems logical for a scheduler to run as many virtual machines on a node as possible until all available virtual CPUs are taken. Second, when the frequency is dynamically reduced, the difference between running nBench on 2 VMs versus 8 VMs at 1.6GHz is only 28.3 Watts. When running the benchmark at 2.668 GHz (the maximum frequency available), this difference grows to 65.2 Watts, resulting in a larger VM power consumption difference and also a larger overall power consumption of 209 Watts.

It would be desirable to run each core at its lowest voltage 100% of the time to minimize power consumption, however one must consider the performance impact of doing so. In Figure 1.16 the average nBench Integer calculation Index is illustrated with the number of VMs per node and operating frequency dynamically varied for each test.

Figure 1.16 illustrates how the performance degradation due to operating frequency scaling is a linear relationship. This eliminates any question of unexpected slowdowns in performance when running at frequencies lower than the maximum, such as 1.6GHz. Another interesting observation is the node's performance running 8 VMs. Due to Intel's Hyperthreading technology, the CPU reports as 8 virtual cores within the host OS (Dom0) even though there are really only 4 cores per node. In our case of running nBench on virtual machines, theres appears to be an overall increase in throughput when using 8 VMs instead of just 4. While the performance of each individual VM is only approximately

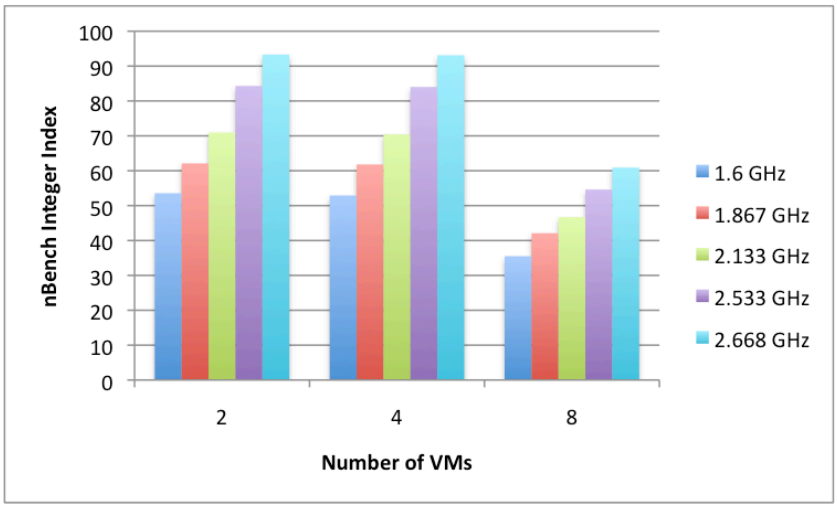

FIGURE 1.16: Performance impact of varying the number of VMs and operating frequency

67% as fast when using 8 VMs instead of 4, there are twice as many VMs contributing to an overall performance improvement of 34%, which is consistent with previous reports [52] of optimal speedups when using Hyperthreading. Therefore its even more advisable to schedule as many virtual machines on one physical node because it maximizes not only power consumption per VM, but also overall system performance.

To evaluate the energy savings of Algorithm 1, we consider the following small OpenNebula pool of just 4 servers. Each server within the pool is a 2.6Ghz Intel Core i7 920 with 12GB of RAM. We assume each server can hold 8 VMs as it has 8 virtual cores. At idle, they consume 105 Watts of power and under 100% load they consume 170 Watts (see Figure 1.12). If we execute the default OpenNebula scheduler to schedule 8 virtual machines, each server would gain 2 VMs and would consume 138 Watts with a total pool power consumption of 552 Watts. However when Algorithm 1 is used, all the VMs are scheduled to the first machine in the pool. This one machine operates at the full 170 Watts, however all other machines idle at 105 Watts, resulting in a pool power consumption of 485 Watts. Therefore, using our power based scheduling algorithm, we conserve 12% of the system's power on only 4 machines on a normal load, as seen in Figure 1.17. If the live migration and shutdown strategy is also deployed, some servers could be dynamically shutdown to further conserve energy, leading to further energy savings.

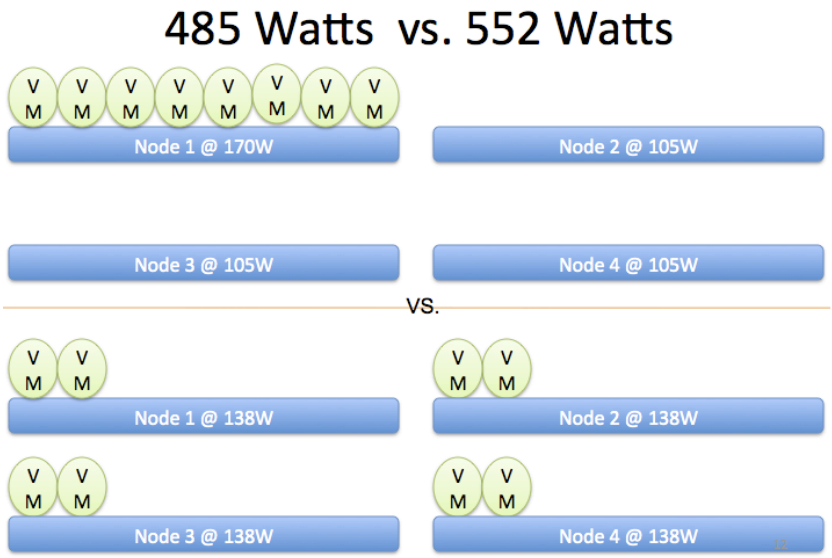

FIGURE 1.17: Illustration of Scheduling power savings

1.6 Virtual Machine Images

While scheduling and management of virtual machines within a private Cloud environment is important, one must realize what is actually being scheduled. In a normal Cloud environment like the Amazon's EC2 [12], full Operating System VMs are scheduled, often to carry out specific tasks in mass. These VM instances contain much more than they need to in order to support a wide variety of hardware software and varying user tasks. While this is ideal for a desktop based environment, it leads to wasted time and energy in a server based solution. A hypervisor provides the same virtualized hardware to each VM and each VM is typically designed for a specific task. In essence, we want the OS within the VM to act only as a light wrapper which supports a few specific but refined tasks or services, and not an entire desktop/application suite. In order to accomplish this task, we need to concentrate on two areas: VM image size and boot time.

Normal x86 hardware can vary widely, so most modern operating systems are able to detect various hardware and load modules on the fly

upon startup. It is common for bootstrap to spend 15 seconds running modprobe to load only a single module. This is not an issue with a virtual machine environment since the hardware is standardized and known in advance. The modules in the system and many of the time consuming probing functions can be reduced upon bootup within a VM environment. In [18] considerable amount of time is saved by changing the IDE delay times for probing new hardware.

Another technique for reducing the boot time is to orchestrate the boot sequence in a more efficient way. Often many daemons and applications are loaded for general use which (in the case of a lightweight VM instance) aren't needed and can be removed. This includes standalone server applications like Window managers and the X11 windowing system. This would also remove the system's disk footprint considerably saving valuable hard drive space in distributed file systems as well as network traffic when migrating the machines.

Boot time can be further improved by creating a new order which maximizes both the CPU utilization and I/O throughput. The use of bootchart [51] can profile where bootup system inefficiencies occur and to allow for optimization of the boot sequence. Another useful tool is readahead [41]. Readahead profiles the system startup sequence and uses file pre-fetching techniques to to load files into memory before they are requested. Therefore an application reads directly from system memory and does not have to wait for disk seek-time.

1.6.1 Virtual Machine Image Analysis

In order to evaluate the performance of our VM image design, we must create a prototype. There are two paths available to build such a VM OS image. The first is a bottom up approach where a basic Linux kernel is built upon to reach the minimal feature set needed. This requires developing an entirely new distribution from scratch. While this may be the "cleanest" way, it would require a large development team and is therefore infeasible for this project. The other option involves a top-down approach of taking a common distribution and removing certain components from it, making for a lighter and faster sub-distribution. This route is more practical as it does not require reinventing the wheel, and the option to keep components such as a package management system and a large distribution library are maintained.

Following the second approach, a custom Linux image was created to illustrate the possibility of a fast and lightweight VM OS. Starting with Ubuntu Linux version 9.04, all unnecessary packages were removed, including the Gnome window manager and X11. By removing this multitude of packages, the system image is reduced from 4Gb to only 636Mb.

This minimization speeds up migration of the image from one server to another as there is less network traffic during the movement phase. A number of other packages, libraries and boot level daemons were also removed from the startup process. At the final stage, the image is a minimal Linux installation with only absolutely necessity components. One thing that was left in was the Synaptic package management system, so if any tools or libraries are needed it is a trivial process to have them installed on the system. While the package management system does take up some room, it is well worth the extendability it provides to the system. A number of kernel modules were also removed from the 2.6.28-11 kernel to speed up the kernel init and modprobe processes as much as possible.

To test the speed of the custom image, both it and a basic Ubuntu 9.04 installation were moved to a VMWare server with 2.5Ghz Intel Core 2 Duo and 4GB of ram. The standard Ubuntu image booted from BIOS in 38 seconds. With our custom VM image, boot time was reduced dramatically to just 8 seconds. By comparing the boot charts in figures 1.18 and 1.19, we can see there is a drastic change in boot time, resulting in a boot time decrease of 30 seconds. Instead of a large amount of I/O blocking, all disk I/O is done at once towards the beginning, allowing for much higher utilization of the CPU. While a boot time of 8 seconds is a considerable improvement, we can do better. The kernel still takes a full 2 seconds to load; with some additional improvements a second or more could possibly be saved.

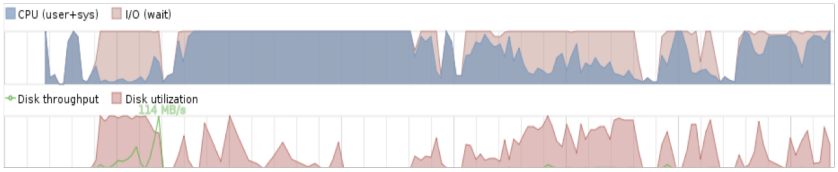

FIGURE 1.18: Bootup chart of the default Ubuntu Linux VM image

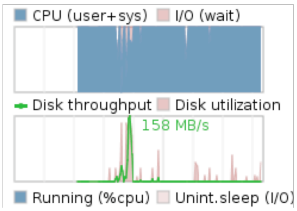

FIGURE 1.19: Bootup chart of Minimal Linux VM image

1.6.1.1 Minimal VM Discussion

Consider a VM which is started on a machine that requires 250 watts of power. Spending 30 seconds on booting up the VM results in 2.08 wh or .002 Kwh of energy used. While this saving of .002Kwh or 30 seconds doesn't seem like much, its effects are actually quite significant. In just a small private Cloud system, it is common for 100 VMs to be created every hour, depending on system load and utilization. As such, over 1750 Kwh will be wasted per year. Thus these changes in the VM image may lead to hundreds or thousands of dollars in savings. Furthermore, the power savings realized through using lightweight VM images on a 10 Megawatt facility where thousands of VMs are started every minute equate to tens or hundreds of thousands of dollars a year.

1.7 Conclusion

As the prevalence of Cloud computing continues to rise, the need for power saving mechanisms within the Cloud also increases. In this paper we have presented a novel Green Cloud framework for improving system efficiency in a data center. To demonstrate the potential of our framework, we have presented new energy efficient scheduling, VM system image, and image management components that explore new ways to conserve power. Though our research presented in this paper, we have found new ways to save vast amounts of energy while minimally impacting performance.

As the prevalence of Cloud computing continues to rise, the need for power saving mechanisms within the Cloud also increases. In this paper we have presented a novel Green Cloud framework for improving system efficiency in a data center. To demonstrate the potential of our framework, we have presented new energy efficient scheduling, VM system image, and image management components that explore new ways to conserve power. Though our research presented in this paper, we have found new ways to save vast amounts of energy while minimally impacting performance.

Acknowledgment

This document was developed with support from the National Science Foundation (NSF) under Grant No. 0910812 to Indiana University for FutureGrid: An Experimental, High Performance Grid Test-bed. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

Bibliography

[1] CMS. Web Page.

[2] Futuregrid project. Web Page.

[3] The magellan project. Webpage.

[4] Report to congress on server and data center energy efficiency - public law 109-431. Webpage, July 2007.

[5] Webpage, May 2008.

[6] Webpage, May 2008.

[7] Wattsup power meters. Website, 2009.

[8] Webpage, 2010.

[9] S. Adabala, V. Chadha, P. Chawla, R. Figueiredo, J. Fortes, I. Krsul, A. Matsunaga, M. Tsugawa, J. Zhang, Mi. Zhao, L. Zhu, and X. Zhu. From virtualized resources to virtual computing Grids: the In-VIGO system. *Future Generation Comp. Syst.*, 21(6):896–909, 2005.

[10] U.S Energy Information Administration. Net generation by energy source: Total (all sectors). Webpage, March 2010.

[11] B. Alexander. Web 2.0: A New Wave of Innovation for Teaching and Learning? *Learning*, 41(2):32–44, 2006.

[12] Amazon. Elastic Compute Cloud.

[13] Assaf Arkin, Sid Askary, Scott Fordin, Wolfgang Jekeli, Kohsuke Kawaguchi, David Orchard, Stefano Pogliani, Karsten Riemer, Susan Struble, Pal Takacsi-Nagy, Ivana Trickovic, and Sinisa Zimek. Web Services Choreography Interface, June 2002. Version 1.0.

[14] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. Above the clouds:

A berkeley view of cloud computing. Technical report, University of California at Berkeley, February 2009.

[15] 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 *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, pages 164–177, New York, U. S. A., Oct. 2003.

[16] K.J. Barker, K. Davis, A. Hoisie, D.J. Kerbyson, M. Lang, S. Pakin, and J.C. Sancho. Entering the petaflop era: the architecture and performance of Roadrunner. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*. IEEE Press Piscataway, NJ, USA, 2008.

[17] Christian Belady. The Green Grid Data center Efficiency Metrics: PUE and DCIE. Technical report, The Green Grid, Feb. 2007.

[18] T.R. Bird. Methods to improve bootup time in Linux. In *Proc of the Ottawa Linux Symp*, 2004.

[19] Deva Bodas. Data Center Power Management and Benefits to Modular Computing. In *Intel Developer Forum*, 2003.

[20] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC-08, IEEE CS Press, Los Alamitos, CA, USA)*, pages 5–13, 2008.

[21] C. Catlett. The philosophy of TeraGrid: building an open, extensible, distributed TeraScale facility. In *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2002*, pages 8–8, 2002.

[22] CERN. LHC Computing Grid Project. Web Page, December 2003.

[23] JS Chase, DE Irwin, LE Grit, JD Moore, and SE Sprenkle. Dynamic virtual clusters in a grid site manager. In *12th IEEE International Symposium on High Performance Distributed Computing, 2003. Proceedings*, pages 90–100, 2003.

[24] Wu chun Feng and Kirk W. Cameron. The Green500 List: Encouraging Sustainable Supercomputing. *IEEE Computer*, 40(12):50–55, 2007.

[25] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In *In Proceedings of the 2nd ACM/USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, May 2005.

[26] D. DiNucci. Fragmented future. *AllBusiness–Champions of Small Business*, 1999.

[27] J.J. Dongarra, H.W. Meuer, and E. Strohmaier. Top 500 supercomputers. website, November 2008.

[28] W. Feng and K.W. Cameron. The Green500 List: Encouraging Sustainable Supercomputing. pages 50–55. IEEE Computer Society, 2007.

[29] W. Feng, A. Ching, C.H. Hsu, and V. Tech. Green Supercomputing in a Desktop Box. In *IEEE International Parallel and Distributed Processing Symposium, 2007. IPDPS 2007*, pages 1–8, 2007.

[30] W. Feng, X. Feng, and R. Ge. Green Supercomputing Comes of Age. *IT PROFESSIONAL*, 10(1):17, 2008.

[31] J. Fontan, T. Vazquez, L. Gonzalez, R. S. Montero, and I. M. Llorente. OpenNEbula: The Open Source Virtual Machine Manager for Cluster Computing. In *Open Source Grid and Cluster Software Conference*, San Francisco, CA, USA, May 2008.

[32] William Forrest. How to cut data centre carbon emissions? Website, December 2008.

[33] I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputer Applications*, 11(2):115–128, 1997. ftp://ftp.globus.org/pub/globus/papers/globus.pdf.

[34] I. Foster, C. Kesselman, et al. The Physiology of the Grid:An Open Grid Services Architecture for Distributed Systems Integration. Technical report, Argonne National Laboratory, Chicago, January 2002.

[35] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *Intl. J. Supercomputer Applications*, 15(3), 2001.

[36] I. Foster, Y. Zhao, I. Raicu, and S. Lu. Cloud Computing and Grid Computing 360-Degree Compared. In *Grid Computing Environments Workshop, 2008. GCE'08*, pages 1–10, 2008.

[37] Google. Data center efficiency measurements. Webpage.

[38] I. Gorton, Greenfield, P., Szalay, A., and R. Williams. Data-Intensive Computing in the 21st Century. *IEEE Computer*, 41(4):30–32, 2008.

[39] P. W. Hale. Acceleration and time to fail. *Quality and Reliability Engineering International*, 2(4):259–262, 1986.

[40] Evan Hoke, Jimeng Sun, John D. Strunk, Gregory R. Ganger, and Christos Faloutsos. InteMon: continuous mining of sensor data in large-scale self-infrastructures. *Operating Systems Review*, 40(3):38–44, 2006.

[41] Harald Hoyer and Karel Zak. Readahead. Webpage.

[42] Chung hsing Hsu and Wu chun Feng. A feasibility analysis of power awareness in commodity-based high-performance clusters. In *Proceedings of IEEE International Conference on Cluster Computing*, pages 1–10, Los Alamitos, CA, USA, 2005. IEEE Computer Society.

[43] C. Hsu and W. Feng. A power-aware run-time system for high-performance computing. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. IEEE Computer Society Washington, DC, USA, 2005.

[44] K. Keahey, I. Foster, T. Freeman, X. Zhang, and D. Galron. Virtual Workspaces in the Grid. *Lecture Notes in Computer Science*, 3648:421–431, 2005.

[45] J.G. Koomey. Estimating total power consumption by servers in the US and the world. *Final report. February*, 15, 2007.

[46] D. Krafzig, K. Banke, and D. Slama. *Enterprise SOA: Service-Oriented Architecture Best Practices (The Coad Series)*. Prentice Hall PTR Upper Saddle River, NJ, USA, 2004.

[47] T.S. Kuhn. *The structure of scientific revolutions*. University of Chicago press Chicago, 1970.

[48] J.H. Laros III, K.T. Pedretti, S.M. Kelly, J.P. Vandyke, K.B. Ferreira, C.T. Vaughan, and M. Swan. Topics on Measuring Real Power Usage on High Performance Computing Platforms. In *IEEE Cluster*, 2009.

[49] M. J. Litzkow, M. Livny, and M. W. Mutka. Condor - A Hunter of Idle Workstations. In *Proceedings of the 8th International Conference on Distributed Computing Systems (ICDCS)*, pages 104–111, San Jose, California, June 1988. IEEE Computer Society.

[50] J. Luo, A. Song, Y. Zhu, X. Wang, T. Ma, Z. Wu, Y. Xu, and L. Ge. Grid supporting platform for AMS data processing. *Lecture notes in computer science*, 3759:276, 2005.

[51] Ziga Mahkovec. Bootchart. Webpage, 2005.

[52] D.T. Marr, F. Binns, D.L. Hill, G. Hinton, D.A. Koufaty, J.A. Miller, and M. Upton. Hyper-threading technology architecture and microarchitecture. *Intel Technology Journal*, 6(1):4–15, 2002.

[53] Arun Babu Nagarajan, Frank Mueller, Christian Engelmann, and Stephen L. Scott. Proactive fault tolerance for hpc with xen virtualization. In *ICS '07: Proceedings of the 21st annual international conference on Supercomputing*, pages 23–32, New York, NY, USA, 2007. ACM.

[54] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The Eucalyptus Open-source Cloud-computing System. *Proceedings of Cloud Computing and Its Applications*, 2008.

[55] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, F. W
"urthwein, et al. The open science grid. In *Journal of Physics: Conference Series*, volume 78, page 012057. Institute of Physics Publishing, 2007.

[56] S. Sarkar and I. Sfiligoi. GlideCNAF: A Purely Condor Glide-in Based CDF Analysis Farm. Technical report, CDF/DOC/COMP UPG/PUBLIC/7630P, 2005.

[57] T.P. Seager. The sustainability spectrum and the sciences of sustainability. *Business Strategy and the Environment*, 17(7):444–453, 2008.

[58] Sushant Sharma, Chung-Hsing Hsu, and Wu chun Feng. Making a case for a green500 list. In *IEEE International Parallel and Distributed Processing Symposium (IPDPS 2006)/ Workshop on High Performance - Power Aware Computing*, 2006.

[59] T.L. Sterling. *Beowulf cluster computing with Linux*. The MIT Press, 2001.

[60] G.J. Stigler. Economies of Scale, The. *JL & Econ.*, 1:54, 1958.

[61] Qinghui Tang, Sandeep K. S. Gupta, and Georgios Varsamopoulos. Thermal-aware task scheduling for data centers through minimizing heat recirculation. In *CLUSTER*, pages 129–138, 2007.

[62] Qinghui Tang, Sandeep K. S. Gupta, and Georgios Varsamopoulos. Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Data Centers: A Cyber-Physical Approach. *IEEE Trans. Parallel Distrib. Syst.*, 19(11):1458–1472, 2008.

[63] Qinghui Tang, Tridib Mukherjee, Sandeep K. S. Gupta, and Phil Cayton. Sensor-Based Fast Thermal Evaluation Model For Energy Efficient High-Performance Datacenters. In *Proceedings of the Fourth International Conference on Intelligent Sensing and Information Processing*, pages 203–208, Oct. 2006.

[64] Gregor von Laszewski, Fugang Wang, Andrew Younge, Xi He, Zhenhua Guo, and Marlon Pierce. Cyberaide JavaScript: A JavaScript Commodity Grid Kit. In *GCE08 at SC'08*, Austin, TX, Nov. 16 2008. IEEE.

[65] Gregor von Laszewski, Lizhe Wang, Andrew J. Younge, and Xi He. Power-aware scheduling of virtual machines in dvfs-enabled clusters. In *IEEE Cluster 2009*, New Orleans, Louisiana, Aug 2009. IEEE, IEEE.

[66] Gregor von Laszewski, Lizhe Wang, Andrew J. Younge, and Xi He. Power-Aware Scheduling of Virtual Machines in DVFS-enabled Clusters. In *IEEE Cluster 2009*, New Orleans, 31 Aug. – Sep. 4 2009. IEEE.

[67] Gregor von Laszewski, Andrew Younge, Xi He, Kumar Mahinthakumar, and Lizhe 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.

[68] Mike Wagner. The efficiency challenge: Balancing total cost of ownership with real estate demands. Technical report, Cherokee International, 2008.

[69] Andrew J. Younge, Gregor von Laszewski, Lizhe Wang, Sonia Lopez-Alarcon, and Warren Carithers. Efficient resource management for cloud computing environments. In *WIPGC in International Green Computing Conference*. IEEE, August 2010.