J Grid Computing (2012) 10:447–473
DOI 10.1007/s10723-012-9219-2

Energy-Efficient Thermal-Aware Autonomic Management of Virtualized HPC Cloud Infrastructure

Ivan Rodero · Hariharasudhan Viswanathan · Eun Kyung Lee · Marc Gamell · Dario Pompili · Manish Parashar

Received: 1 October 2011 / Accepted: 3 July 2012 / Published online: 27 July 2012

Abstract Virtualized datacenters and clouds are being increasingly considered for traditional High-Performance Computing (HPC) workloads that have typically targeted Grids and conventional HPC platforms. However, maximizing energy efficiency and utilization of datacenter resources, and minimizing undesired thermal behavior while ensuring application performance and other Quality of Service (QoS) guarantees for HPC applications requires careful consideration of important and extremely challenging tradeoffs. Virtual Machine (VM) migration is one of the most common techniques used to alleviate thermal anomalies (i.e., hotspots) in cloud datacenter servers as it reduces load and, hence, the server utilization. In this article, the benefits of using other techniques such as voltage scaling and pinning (traditionally used for reducing energy consumption) for thermal management over VM migrations are studied in detail. As no single technique is the most efficient to meet temperature/performance optimization goals in all situations, an autonomic approach that performs energy-efficient thermal management while ensuring the QoS delivered to the users is proposed. To address the problem of VM allocation that arises during VM migrations, an innovative application-centric energy-aware strategy for VM allocation is proposed. The proposed strategy ensures high resource utilization and energy efficiency through VM consolidation while satisfying application QoS by exploiting knowledge obtained through application profiling along multiple dimensions (CPU, memory, and network bandwidth utilization). To support our arguments, we present the results obtained from an experimental evaluation on real hardware using HPC workloads under different scenarios.

Keywords Cloud infrastructure · Virtualization · Thermal management · Energy-efficiency

I. Rodero (✉) · H. Viswanathan · E. K. Lee · M. Gamell · D. Pompili · M. Parashar
NSF Cloud and Autonomic Computing Center, Rutgers Discovery Informatics Institute, Rutgers University, 94 Brett Road, Piscataway, NJ 08854, USA
e-mail: irodero@cac.rutgers.edu

H. Viswanathan
e-mail: hari_viswanathan@cac.rutgers.edu

E. K. Lee
e-mail: eunkyung_lee@cac.rutgers.edu

M. Gamell
e-mail: mgamell@cac.rutgers.edu

D. Pompili
e-mail: pompili@cac.rutgers.edu

M. Parashar
e-mail: parashar@cac.rutgers.edu

1 Introduction

1.1 Motivation and Background

Virtualized datacenters and clouds provide the abstraction of nearly-unlimited computing resources through the elastic use of consolidated resource pools. These platforms are being increasingly considered for traditional High-Performance Computing (HPC) workloads that have typically targeted Grids and conventional HPC platforms. The scale and overall complexity of modern datacenters are growing at an alarming rate (current datacenters contain tens to hundreds of thousands of computing and storage devices running complex applications) and, hence, energy consumption, heat generation, and cooling requirements have become critical concerns both in terms of the growing operating costs as well as their environmental and societal impacts [1]. Addressing these concerns while balancing multiple requirements, including performance, Quality of Service (QoS), and reliability, is thus an important task for service providers.

Thermal awareness, which is the knowledge of temperature distribution inside a datacenter, is essential to maximize energy and cooling efficiency as well as to minimize server failure rates. Hence, in this article, we propose techniques to acquire and characterize the thermal behavior of a datacenter so to bestow self-protection and self-healing capabilities on datacenter management systems. Thermal-aware datacenter management is aimed at minimizing both the impact on the environment and the Total Cost of Ownership (TCO) of datacenters without any increase in the number of Service Level Agreement (SLA) violations.

Prior work in the field of thermal management explores efficient methods for improving heat extraction through cooling system optimization as in [20, 44–46] or methods for controlling heat generation [31, 48, 52] that focus on how to distribute and migrate workloads (if necessary) in such a way as to avoid undesired thermal behavior (e.g., overheating of computing equipment). However, maximizing energy efficiency and utilization of datacenter resources, minimizing undesired thermal behavior, and ensuring QoS guarantees for HPC application are conflicting goals. Careful consideration of extremely challenging tradeoffs among these goals is essential for addressing them simultaneously.

1.2 Architecture and Goals

The long-term goal of our approach is to autonomically manage datacenters using the information from sensors and taking decisions at different levels (through controllers) based on the optimization goals (e.g., performance, energy efficiency, cost). To do this, we consider an architecture composed of layers belonging to different abstract components with different responsibilities but with the same common objectives. The architecture (see Fig. 1) is composed of four lay-

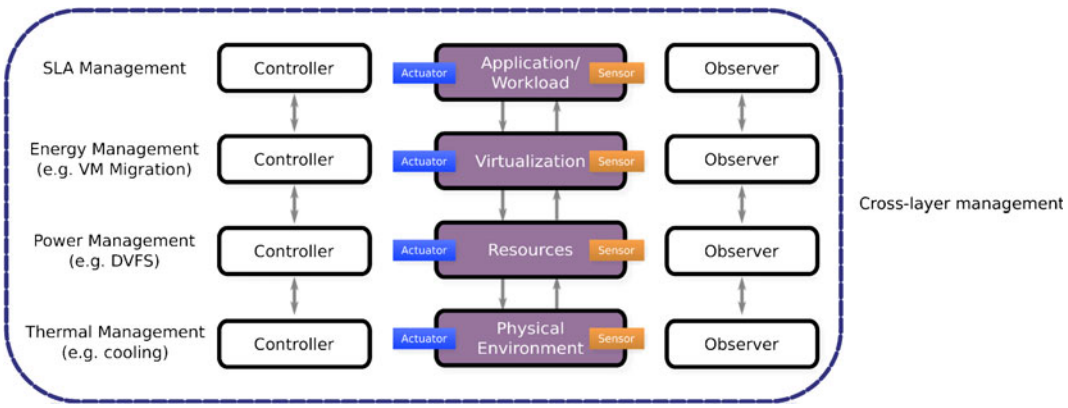


Fig. 1 Layered architecture model

ers: environment layer (which detects, localizes, characterizes, and tracks thermal hotspots using a hybrid sensing infrastructure composed of in-built as well as external scalar temperature and humidity sensors), physical resource layer (which manages the hardware and software components of servers), virtualization layer (which instantiates, configures, and manages VMs), and application layer (which is aware of the workload's and applications' characteristics and behavior). For autonomic reactive thermal management we focus on the possible interactions among all the four layers based on temperature, power consumption, and application QoS requirements. Note that the environment layer comprising of the hybrid sensing infrastructure monitors both the micro (chip and server level) as well as the macro (rack and datacenter level) phenomena, i.e., temperature distribution, in the datacenter. The proposed approach represents a significant and transformative shift towards cross-layer autonomics for datacenter management problems, which have so far been considered mostly in terms of individual layers.

1.3 Approach

One component of our proposed solution continuously monitors temperature changes and handles thermal emergencies (using VM migrations, voltage and/or frequency scaling, and pinning) while ensuring that application performance is not degraded. Another component strives to achieve energy efficiency while adhering to SLAs by co-locating compatible workloads (or VMs hosting the workloads) in physical machines when VM migration is chosen as the thermal-management strategy. Compatibility among workloads is determined through extensive HPC application profiling. It is evident that the first component involves cross-layer interactions among the virtualization, physical, and the environment layers while the second component requires interactions between the application and virtualization layers. These two components together help us realize our envisioned cross-layer thermal-aware autonomic management of datacenters through careful consideration of trade-offs among the requirements from energy, application QoS, and operating environment perspectives. Our approach is twofold:

(1) *Reactive thermal management:* VM migration is one of the most common techniques used to alleviate thermal anomalies (i.e., hotspots) in cloud datacenter's servers by reducing the load and, therefore, by decreasing the server utilization. Switching off idle servers is also a typical policy along with VM migration for power management. In this article, we explore ways to take actions at the server side before performing costly migrations of VMs, including techniques that can be used as temporary actions that may facilitate migrations without incurring in additional penalty in terms of server thermal behavior. We aim at optimizing the energy efficiency of datacenters while ensuring the QoS delivered to the users. Specifically, we focus on exploiting VM Monitor (VMM) configurations (such as pinning techniques, i.e., CPU affinity) in Xen platforms as well as Dynamic Voltage and Frequency Scaling (DVFS) at the physical resource layer to alleviate undesired thermal behavior of the datacenter's hardware components. However, not always the same mechanism is the most efficient to meet the desired optimization goals.
(2) *Application-centric VM allocation:* To address the problem of VM allocation—which arises in the following two situations: initial mapping of VMs to physical servers and migration of VMs from one physical server to another in response to (reactive) undesired thermal behavior and/or SLA violations—we propose an application-centric energy-aware VM allocation model. Our model can help reduce the number of SLA violations, avoid undesired thermal behavior, and minimize the energy costs by improving resource utilization.

In contrast to existing VM allocation solutions that are aimed at satisfying only the resource utilization requirement of an application along only one dimension (CPU utilization), our approach to VM consolidation considers the application's resource utilization requirements along multiple dimensions, i.e., CPU, memory, disk I/O, and

network subsystems. In addition, our approach is application centric as in [15, 17, 56] and, hence, enables co-location of compatible VMs on servers during consolidation. This co-location minimizes the usual adverse effects of resource contention and virtualization overhead on application performance while retaining the benefits of server consolidation. The aforementioned application awareness is acquired through HPC benchmarks profiling.

1.4 Contributions

The main contributions of this work are: (1) a study of different reactive thermal management techniques for virtualized and instrumented datacenters from the energy perspective towards the design of an autonomic approach, (2) a study of the tradeoffs between performance, energy efficiency, and thermal efficiency of the techniques for HPC workloads, (3) development of an autonomic reactive thermal-management solution that leverages the knowledge gained in the aforementioned studies to choose the most appropriate techniques for alleviating thermal anomalies, (4) creation of an empirical VM allocation model from data obtained by running HPC workloads extensively on a system with a general-purpose rack server configuration, and (5) development of a proactive application-centric VM allocation algorithm that uses this empirical model. To support the arguments of our approach, we present the results obtained from simulation and experimental evaluation on real hardware using HPC workloads under different scenarios.

To evaluate our proposed techniques for thermal management of HPC cloud infrastructure, we conducted the simulations and experiments on real hardware using parallel workload traces from real world HPC production systems. The results obtained from simulations using real world production HPC workload traces show that our proactive VM allocation solution significantly contributes to energy efficiency (12 % reduction in energy consumption) and/or optimization of the application performance (18 % reduction in execution time) depending on optimization goals. Our reactive thermal management solution in conjunction with the proactive consolidation approach outperforms other traditional thermal management approaches like load redistribution (VM migrations) and Temperature-Aware (TA) VM placement in terms of energy consumption (up to 9 % less than that of TA's), makespan (up to 12 % less than that of TA's), maximum server operating temperatures (up to 10 % less than that achieved by the load redistribution technique), and percentage of time for which servers operate in unsafe temperatures (up to 25 % less than that achieved by the load redistribution technique).

The rest of the article is organized as follows. In Section 2, we review related work. In Section 3, we discuss the temporal-spatial characteristics of hotspots and our empirical-model-based approach to reactive thermal management. In Section 4, we describe our empirical VM allocation model and present our VM allocation algorithm, which uses this model. In Section 5, we discuss our evaluation methodology as well as the results we obtained from simulations and empirical experiments on real hardware. Finally, in Section 6, we conclude the article and outline directions for future work.

2 Related Work

In this section, firstly, we review the state of the art in thermal management [38, 45] at the environment as well as application layers. Secondly, we review power management techniques (such as DVFS and pinning at the physical resource layer) that we propose to exploit for energy-efficient thermal management in place of costly VM migrations. Thirdly, we discuss prior work in the field of energy-efficient VM allocation, which is key for VM migration decisions in reaction to thermal anomalies. Finally, we review existing cross-layer solutions, which are characterized by pair-wise interactions between layers.

2.1 Thermal Management at the Environment Layer

Prior research efforts in this area focus on management of heat extraction in a datacenter [20, 44–46]. In [16], Greenberg et al. profiled and benchmarked the energy usage of 22 datacenters

and concluded that the key to energy efficiency is air circulation management (for effective and efficient cooling). Lee et al. [26] propose a proactive control approach that jointly optimizes the air conditioner compressor duty cycle and fan speed to prevent heat imbalance and to minimize the cost of cooling in data centers. As many datacenters employ raised floors with perforated tiles to distribute the chilled air to racks, researchers have tried to gain valuable insights into efficient airflow distribution strategies in such datacenter layouts [20, 44–46]. Other research efforts were aimed at improving the efficiency of cooling systems through thermal profiling [27, 28, 47], which is the extraction of knowledge about air and heat circulation using measurements from scalar sensors and mathematical models. However, capturing this complex thermodynamic phenomena using compute-intensive models [5, 37] (e.g., computational fluid dynamics) is prohibitive in terms of computational overhead.

2.2 Thermal Management at the Application Layer

Another popular approach to thermal management has been controlling the heat generation inside a datacenter [4, 18, 31, 32, 52] through thermal-aware workload placement. Moore et al. [31] propose the use of "offline experiments" to characterize the thermodynamic phenomena (heat recirculation) inside the datacenter and schedule workloads by taking the temperature distribution into account. In [32], Moore et al. eliminate the need for the aforementioned offline experiments and propose an online machine-learning-based method to model the thermal behavior of the datacenter. Bash et al. [4] propose a policy to place the workload in areas of a datacenter that are easier to cool, which results in cooling power savings. They use scalar temperature sensor measurements alone to derive two metrics that help decide whether to place workload on a server or not. Tang et al. [52] develop a linear, low-complexity process model to predict the equipment inlet temperatures in a datacenter given a server utilization vector and formalize (mathematically) the problem of minimizing the datacenter cooling cost as the problem of minimizing the maximal (peak) inlet temperature through task assignment. In [33], Mukherjee et al. explore a spatio-temporal thermal-aware job scheduling as an extension to spatial thermal-aware solutions like [18, 31, 32, 52].

2.3 Power Management at the Physical Resource Layer

Several research efforts propose methods to jointly manage power and performance at the physical resource layer. One of the most used techniques in the last decades to reduce power consumption is DVFS. Researchers have developed different DVFS scheduling algorithms and mechanisms to save energy while provisioning resources under deadline restrictions. Chen et al. [7] address resource provisioning and propose power management strategies with SLA constraints based on steady-state queuing analysis and feedback control theory. They use server turn on/off and DVFS for enhancing power savings. Ranganathan et al. [40] highlight the current issue of under utilization and over-provisioning of the servers and present a solution for peak power budget management across a server ensemble to avoid excessive over-provisioning considering DVFS and memory/disk scaling. Rusu et al. [43] propose a cluster-wide on/off policy based on dynamic reconfiguration and DVFS. They focus on power, execution time, and server capacity characterization to provide energy management. Rodero et al. [41] studied application-centric aggressive power management of data centers resources for HPC workloads considering power management mechanisms and controls available at different levels and for different subsystems (i.e., CPU, memory, disk, network). Kephart et al. [22] and Das et al. [9] address the coordination of multiple autonomic managers for power/performance tradeoffs by using a utility function approach in a non virtualized environment.

Virtual machine monitor configurations (e.g., within the Xen hypervisor) such as pinning (i.e., CPU affinity) have been proposed earlier (as in [50]) to optimize VM performance. In this article, we investigate the use of power management and performance optimization techniques (DVFS and pinning) for mitigating thermal anomalies before

considering costly VM migrations. To the best of our knowledge, none of the existing approaches have investigated and exploited pinning to mitigate the effects of thermal anomalies with energy efficiency and performance as optimization goals.

2.4 Energy Management at the Virtualization Layer

A large body of work in datacenter energy management addresses the problem of the request distribution at the VM management level in such a way that the performance goals are met and the energy consumption is minimized. VM consolidation techniques involve filling up physical servers with VMs (using heuristics like first fit, best fit, etc.) until high server subsystem (CPU, memory, disk storage, network interface) utilization is achieved while still ensuring that the individual VM's subsystem utilization requirements are met. Apparao et al. [3] present a study on the impact of consolidating several applications on a single server running Xen.

Server resource provisioning or VM allocation can be static or dynamic. It is static when a VM is being allocated physical resources for the first time and the problem of VM allocation reduces to a deterministic or statistical bin/vector-packing problem [2] depending on how the VM utilization requirement is characterized. In the dynamic case, VMs are first consolidated using any simple bin-packing heuristic and the variations in VM's utilization requirements are handled through live VM migrations [6, 19, 51, 53, 54] or through dynamic server resource provisioning [29, 34] whenever necessary.

2.5 SLA-Management at the Virtualization Layer

VM migrations are performed either reactively [42] or proactively [6] in such a way as to avoid equipment overheating and/or SLA violations. Kochut et al. [23] provide an estimate of the expected improvement in response time due to a migration decision and determines which VMs are best candidates to be placed together. In [19], Hermenier et al. determine the order in which the VM migrations should occur in addition to deciding which VMs to migrate so to minimize the impact on application performance in terms of execution time. Stoess et al. [51] developed a multi-tiered infrastructure that enables intra-node virtual CPU (vCPU) migration and inter-node live VM migration for workload consolidation and thermal balancing. Voorsluys et al. [55] present an evaluation on the effects of live migration of virtual machines on the performance of applications running inside Xen VMs.

On-demand server resource provisioning techniques monitor the workloads on a set of VMs and adjust the instantaneous resources availed by VMs. Song et al. [49] propose an adaptive and dynamic scheme for adjusting resources (specifically, CPU and memory) among virtual machines on a single server to share the physical resources efficiently. Menasce et al. [29] proposed an autonomic controller and showed how it can be used to dynamically allocate CPUs in virtualized environments with varying workload levels by optimizing a global utility function. Meng et al. [30] exploited statistical multiplexing of VMs to enable joint VM provisioning and consolidation based on aggregated capacity needs. However, all the aforementioned techniques are aimed at satisfying the resource utilization level guarantees and do not consider the application-level performance (execution time).

2.6 Cross-Layer Solutions For Power-, Thermal-, and QoS-Management

Nathuji et al. [35] investigate the integration of power management at the physical layer and virtualization technologies. In particular they propose VirtualPower to support the isolated and independent operation of virtual machines and control the coordination among virtual machines to reduce the power consumption. In [34], Nathuji et al. consider the heterogeneity of the underlying platforms (in terms of processor and memory subsystem architecture) to efficiently map the workloads to the best fitting platforms. Laszewski et al. [25] present a scheduling algorithm for VMs in a cluster to reduce power consumption using DVFS. Kumar et al. [24] present vManage, a practical coordination approach that loosely couples platform

and virtualization management aimed at improving energy savings and QoS and at reducing VM migrations.

Heath et al. [18] propose emulation tools (Mercury and Freon) for investigating the thermal implications of power management. In [39], Ramos et al. present C-Oracle, a software infrastructure that dynamically predicts the temperature and performance impact of different thermal management reactions (such as load redistribution and dynamic voltage and frequency scaling) into the future, allowing the thermal management policy to select the best reaction.

Recently, researchers have started to focus on application- or workload-aware VM consolidation that not only achieves all the objectives as its traditional resource-utilization- and energy-aware counterparts but also ensures minimum degradation to application performance due to resource multiplexing and virtualization overhead [15, 17, 56]. Application-aware VM allocation is not only aimed at energy-efficient VM consolidation but also at co-locating VMs that are "compatible" so that further gains can be achieved in terms of energy savings and overhead for virtualization. In [17, 56], the authors propose to consolidate VMs with similar memory content on the same hosts for higher memory sharing and Govindan et al. [15] propose to consolidate based on interprocess communication patterns. Zhu et al. [57] propose pSciMapper, a power-aware method for consolidation virtual machines that host scientific workflow tasks. They use a dimensionality reduction technique, Kernel Canonical Correlation Analysis (KCCA), to associate temporal features extracted from time series data about resource requirements of a workflow task with its power consumption and execution time (performance). Information about the tasks' power consumption and performance are exploited in an online consolidation algorithm.

All the aforementioned cross-layer solutions are characterized only by pair-wise interaction between two layers—physical and virtualization layers, physical and environment layers, or application and virtualization layers. In our proposed thermal-aware management solution interactions between the application and the virtualization layers are leveraged for proactive VM allocation while interactions among all the four layers are exploited for reactive thermal management. In other words, in our thermal management solution, the environment layer alerts the virtualization layer and physical layer about undesired thermal behavior. These two layers jointly decide whether to use power management techniques at the physical layer or VM migration with help from application layer to alleviate the thermal anomalies.

3 Reactive Thermal Management in Datacenters

In this section, we briefly discuss the characteristics of thermal hotspots that majorly cause thermal inefficiency in datacenters. Then, we introduce three different techniques to mitigate the effects of hotspots and a characterization of the different techniques based on empirical experimentation.

3.1 Characteristics of Thermal Hotspots

Hotspots can be detected using internal and external temperature sensors when the measurements cross specific temperature value defined as "hotspot threshold". Hotspots are difficult to localize accurately in space and are hard to predict in time: this is because the heat transfer via air circulation (convection) and through the server blades and racks (conduction) are phenomena

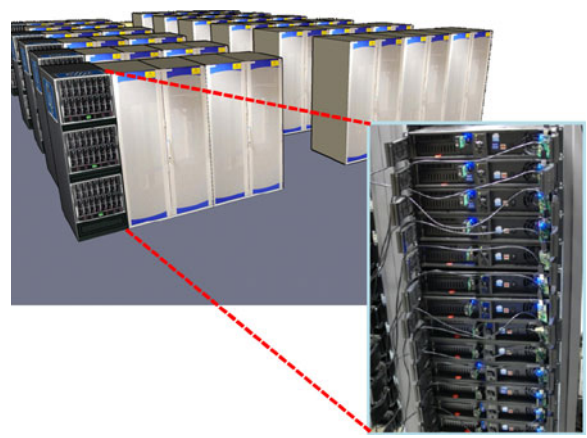
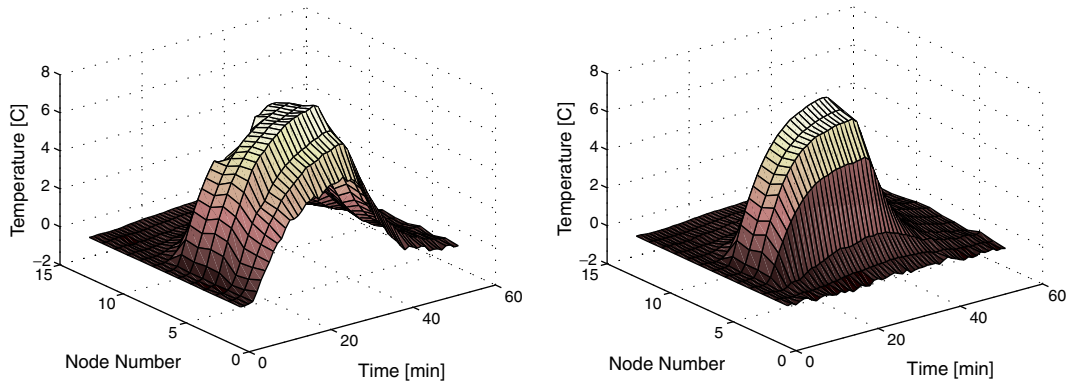


Fig. 2 26 TelosB motes deployed on a rack to observe the characteristics of hotspots



(a) Seven servers running (nodes 7-13) (b) Four servers running (nodes 7-10)

Fig. 3 Temporal and spatial measurement data (temperature) among 13 nodes

difficult to model. In addition, hotspots change their positions in time and space depending on several factors such as distribution and intensity of running workloads, server heat dissipation characteristics, airflow circulation in the datacenter, etc. Thus, it is crucial to understand the characteristics of hotspots for energy-efficient thermal management.

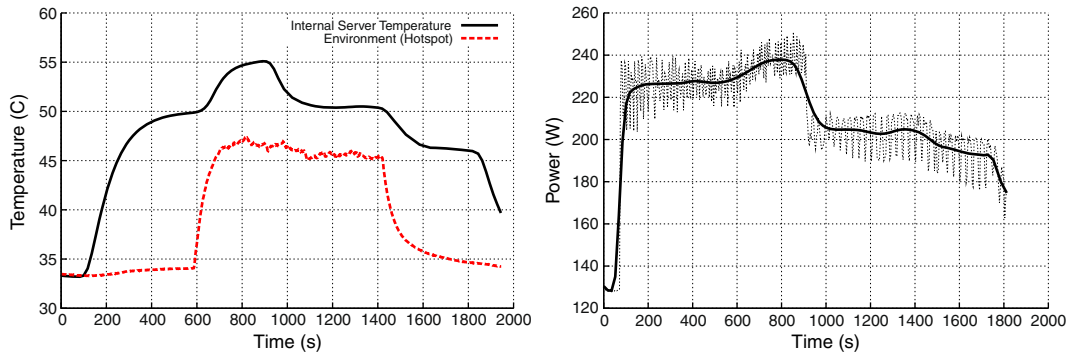
The temporal correlation of the measured temperature data collected from 13 TelosB sensor nodes (deployed in front of the outlet fans of each server in a dual rack system), each placed on 13 vertically arranged servers (as shown in Fig. 2) is depicted in Fig. 3. These TelosB sensor nodes are wirelessly connected and built with IEEE 802.15.4 compliant CC2420 radio (2.4 GHz) as well as with a few sensors (temperature, humidity, and light). Figure 3a and b correspond to the set of experiments of using TelosB nodes in which only servers 7–13 and 7–10 are running, respectively. These results show that some idle servers that are in the proximity of operational servers experience an increase in temperature much higher than that of other idle servers. These results show that a hotspot affecting a server may in fact be caused by another server running on the same rack or on a different one nearby. The increase in the temperature near the outlet fans of idle servers (due to conduction and convection) decreases the ability of the air around idle CPUs to extract heat efficiently when the idle servers become busy. This results in higher server operating temperatures, which in turn increases the risk of failures. The only solution to counter this effect is decreasing the CRAC outlet air temperature, which is not energy efficient.

Figure 4 shows the thermal behavior of a single server in the presence of an hotspot and the corresponding power consumption. It also shows the impact of VM migration on server's temperature and power consumption. The temperature sensors on the TelosB nodes placed near the server outlet fan (external) and near the CPU (internal) are used to observe heat propagation in space and time. The increase of the external temperature, which is controlled with a heat source (from second 600 to 1,400) results in an increase of the server's internal temperature. The figures illustrate the impact of migrating one VM on the server's temperature and power consumption (at second 900—reacting to the hotspot). In comparing the internal server temperature (Fig. 4a) and power consumption (Fig. 4b) we observe that there is a correlation between the two metrics.

3.2 Reactive Thermal Management Approaches

A typical mechanism used to reduce the server's temperature (e.g., to react to a hotspot) consists of decreasing the heat generated by the server, which, based on our measurements is highly correlated with the server's power consumption. As the CPU[1] is the most power consuming

[1] We use the term CPU to refer to each core of a multi-core processor.



(a) Temperature (b) Power Consumption

Fig. 4 Server's thermal and power behavior in the presence of a hotpost

component of a server, we consider CPU power (as a simplification) to describe the different techniques analyzed in this article. Equation 1 shows a simplified dynamic power dissipation model for a CPU, where C is the capacitance of the processor (that we consider fixed), α is an activity factor (also known as switching activity), and V and f are the operational voltage and frequency, respectively [21].

$$P_{\text{cpu}} \sim C \times \alpha \times V^2 \times f. \tag{1}$$

Therefore, the power consumption of a server can be decreased by either reducing the activity of CPUs or reducing the frequency/voltage of CPUs (via DVFS). In the following sections, we discuss different techniques that use this model to reduce server power consumption (and thus the heat generated). We do not take into account cooling or placement issues; rather, we focus on the energy efficiency of thermal management approaches on the servers considering HPC workloads and virtualized environments under the assumption that cooling and placement do not change.

3.2.1 VM Migration

This technique consists of moving a running VM, its guest OS, and all of its applications to a different server. Migrating VMs reduces the CPU activity α in (1) and, if a CPU is freed and the OS implements dynamic CPU power management, it can also reduce the frequency/voltage. We assume that the OS power management is enabled by default. When we need to migrate a VM or multiple VMs to react to a hotspot, one of the following four scenarios is possible:

- Another server is available to host the VM(s) that are being migrated: migration can be performed at the penalty of some overhead (energy, latency, bandwidth).
- A server has been powered down: we can perform the migration after powering the server up at the additional penalty of booting.
- All servers already have some load but some of them have higher thermal efficiency: we can perform the migration but with possible penalty due to resource sharing between migrated workload and the existing workload in the destination server.
- No other server is available to host new VM(s): in this case, we can only suspend the VM(s) until a server becomes available.

3.2.2 Processor DVFS

This is an effective technique to reduce processor power dissipation supported in most of the current processors. DVFS reduces the processor frequency/voltage but not the activity factor. However, CPU-intensive workloads running at low frequencies may experience a significant penalty on their execution time. Within a single server we can use DVFS in two ways:

- *By reducing the frequency/voltage of all CPUs simultaneously*: the workload execution may increase depending of the frequency/voltage reduction and workload characteristics (e.g.,

I/O-intensive workloads may not be significantly penalized).
– *By reducing the frequency/voltage of a subset of CPUs*: if different VMs are independent they may complete their workload at different times. If VMs are coupled (e.g., MPI parallel applications) the workload running in slower CPUs may penalize the workload running in faster CPUs. However, some architectures have restrictions (e.g., by paired CPUs).

3.2.3 VMM Configuration (Pinning)

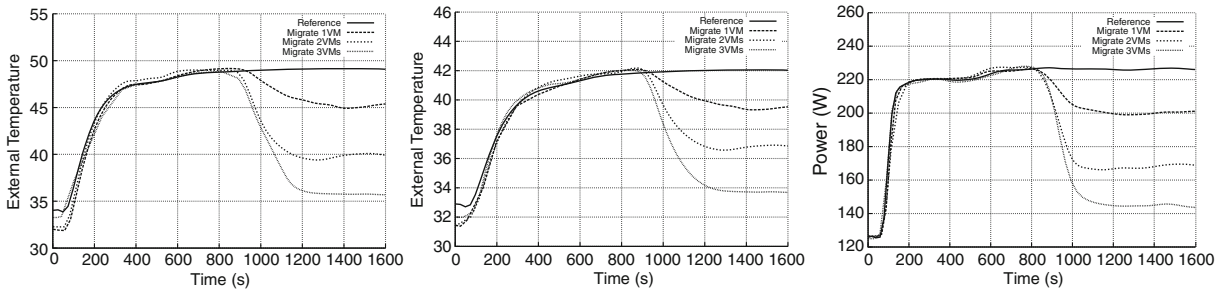
We propose to use this technique to react to hotspots as an alternative to VM migration and DVFS. VMMs may allow *virtual* CPUs (vCPUs) of VMs to be assigned to *physical* CPUs (pCPUs) using two different approaches: without and with affinity (pinning). In the former, the VMM determines how vCPUs are assigned to pCPUs; in the latter, the VMM allows hard assignments of the vCPUs to one or more pCPUs. Pinning techniques are typically used where the characteristics of the workload would benefit from execution on specific CPUs (e.g., cache locality).

We propose to reduce the activity of one or more CPUs by pinning the VMs to the other CPUs. As we assume that the OS performs by default dynamic CPU power management, when a CPU is freed from VMs activity, its frequency or voltage can also be reduced. However, the activity of the running CPUs may be increased, resulting in a penalty in the workload's execution performance caused by higher resource sharing.

3.3 Characterizing Thermal Management Techniques

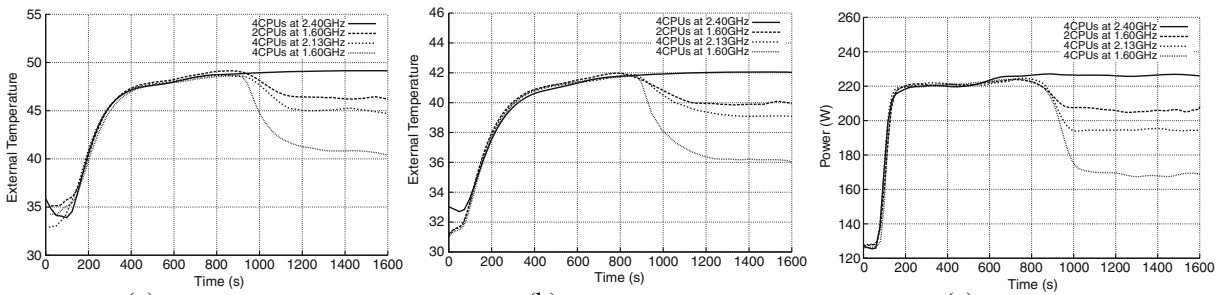
We evaluated the three VM management techniques discussed previously using two Dell servers, each with an Intel quad-core Xeon X3220 processor which operates at four frequencies ranging from 1.6 GHz to 2.4 GHz, 4 GB of memory, two hard disks, and two 1 Gb Ethernet interfaces. This is intended to represent a general-purpose rack server configuration, widely used in virtualized datacenters. The servers were racked one on top of the other and run CentOS Linux operating system with a patched 2.6.18 kernel with Xen version 3.1. To empirically measure the "instantaneous" power consumption of the servers we used a *"Watts Up? .NET"* power meter. This power meter has an accuracy of ±1.5 % of the measured power with sampling rate of 1 Hz. The meter was mounted between the wall power and the server. We estimate the consumed energy by integrating the actual power measures over time. We used built-in temperature sensors and Linux-based hardware monitoring tool 'lm-sensors' to measure CPU temperatures, and TelosB motes to measure both internal (sensors placed inside the chassis, near the CPU) and external (sensors places on the back of the servers) temperatures of the server as described in Section 3.1. We used the `CPUfreq` kernel module and its user-space tools to dynamically adjust the voltage and frequency pairs. We used a Sunbeam SFH111 heater (directed at the servers) in order to emulate a controlled thermal hotspot (e.g., potentially from other servers), which facilitated the experiments since the nodes were isolated with disproportionate cooling.

Figures 5, 6, and 7 show the thermal behavior and power consumption of a server when running a HPC workload with the different VM management techniques considered in this article. Specifically, we used the HPL Linpack benchmark, which is one of the most representative HPC benchmarks (uses intensively different resources such as CPU and memory) since the goal of this experiment was to get insights regarding typical behaviors and trends. All the techniques and configurations have been evaluated under the same conditions. The experiment consists of running HPL in 4 VM instances with the same configuration and applying a given technique 800 s after starting the experiment (which is long enough to reach the steady state). The figures focus on the initial part of the experiment to show better the trends and only plot the Bezier curves for readability. We obtained the external temperature of the server with a sensor placed in the back side of the server, and the internal temperature (in °C) of the server with a sensor placed inside the server. The internal sensor provides the average temperature of the server's components (not only CPU temperature, which can be obtained from its internal sensors).



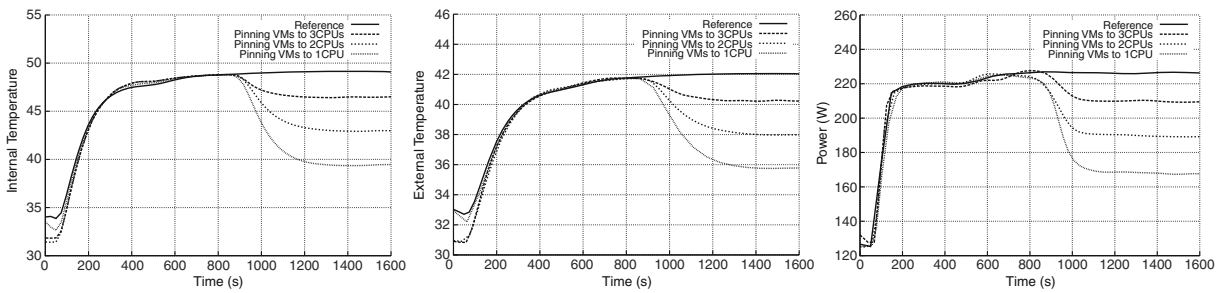
(a) Internal temperaure (b) External temperaure (c) Power Comsuption

Fig. 5 Thermal behavior and power consumption using VM migration



(a) Internal temperaure (b) External temperaure (c) Power comsuption

Fig. 6 Thermal behavior and power consumption using processor DVFS



(a) Internal temperaure (b) External temperaure (c) Power comsuption

Fig. 7 Thermal behavior and power consumption using pinning techniques (pinning 4 VMs to different number of CPUs—from 3 to 1 CPUs)

We can appreciate in the figures that internal and external temperatures are strongly correlated and follow similar trends. However, the internal temperature is almost 8 °C higher than the external temperature. Temperature and power are also correlated but variations in temperature are slower than in power. Overall, the higher reduction in temperature and power are obtained using VM migration. The higher the number of VMs migrated, the more significant the decrease of temperature and power. The reduction of temperature and power is more moderate using DVFS than using VM migration. The highest reduction in temperature and power is obtained operating all 4 CPUs at 1.60 GHz. This is consistent with (1), where higher the operational frequency higher the temperature and power consumption. However, running 2 CPUs at 1.60 GHz and 2 CPUs at 2.40 GHz obtains slightly worse results than running all CPUs at 2.13 GHz.

The reduction in temperature and power using pinning is lower than the one using VM migration but higher than the one using DVFS. The reduction in temperature and power is higher when the VMs are pinned to fewer CPUs. Although using different techniques we obtain different reductions in temperature and power, the plots obtained with the three different mechanisms follow similar patterns. Hence, we can conclude that all of them can reduce the temperature and power consumption effectively, however, we focus on the tradeoffs between performance, energy efficiency, and thermal efficiency (i.e., temperature reduction).

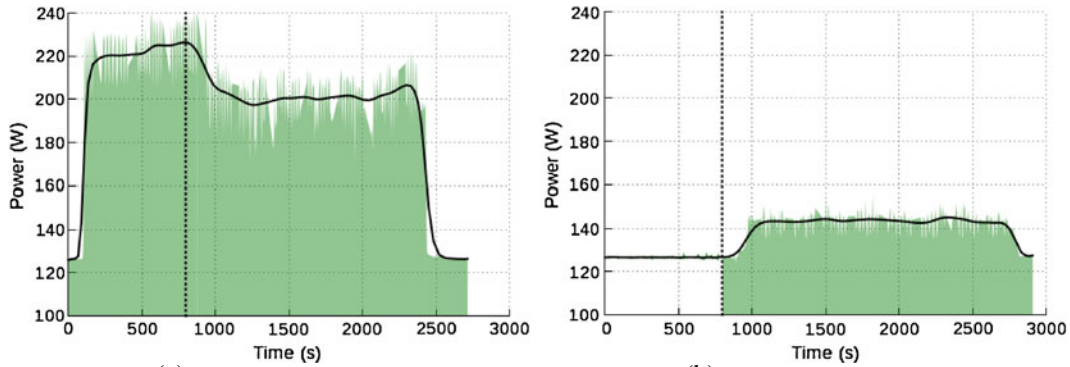
In order to measure the energy consumed by the servers in the experiments that perform VM migrations, we have considered the energy consumed by the original server during the whole execution of the experiment plus the energy consumed by the destination machine from when the VMs start migrating to the destination machine (i.e., shadowed area in Fig. 8a and b).

In the case that the destination machine already hosts other VMs, we take into account the energy consumed from the increase of power in relation to the power consumed by the server before the migration. Although we can find other intermediate scenarios, in our experiments we only consider these two scenarios as they provide simplified but meaningful performance bounds. Although VM migration seems better in terms of thermal efficiency, we look at the tradeoffs between thermal efficiency and other dimensions such as performance and energy efficiency.

Table 1 summarizes the experimental results. It shows i) the obtained makespan (the time needed to complete the workload, which may include the migration overhead), ii) the energy consumed, iii) the Energy Delay Product (EDP), which is a good metric for energy efficiency because it captures the effect of energy management on performance [14], and iv) the reduction of temperature ("Temp. ↓" in Table 1). For the reference execution (regular execution without any specific technique) we present absolute values and for the different techniques we present the results relative the reference execution (in the form of % increase), except for temperature reduction. The best results of each technique are shown in bold.

As we commented previously, VM migration is the technique that achieves highest reduction in temperature. At the price of some migration overhead, we also achieve shorter makespan by migrating two VMs. However, it does not achieve the highest energy efficiency. The configuration where the destination server is empty is better than when the destination server is full in term of makespan but is worse in terms of energy efficiency. The configuration where the destination is full is the best case in term of energy efficiency (except for migration of 2 VMs) but the worst in terms of makespan. However, when the destination server is switched off, the penalty of starting up the server on both makespan and energy efficiency is significant. When the destination server is full, the number of migrated VMs does not influence significantly either the makespan or the energy efficiency. However, the higher the number of migrated VMs, the higher the reduction of temperature. Hence, the potential increase of temperature and the penalty on existing VMs on the destination server should be analyzed in order to identify the tradeoffs between increasing the number of migrated VMs and the negative effects on the destination machine.

In most of the cases, DVFS obtains worse results than the other techniques, in contrast to



(a) Source server (running 4 VMs) (b) Destination server (initially idle)

Fig. 8 Power consumption of source/destination servers when migrating one VM. Migration starting time is depicted with the *dashed vertical line*

other approaches that only focus on DVFS mainly for non-virtualized environments. To obtain a similar reduction of temperature with DVFS, the penalty on both makespan and energy efficiency is higher. As we commented previously, running all CPUs at 2.13 GHz works better than running 2 CPUs at 1.60 GHz. In fact, if the workloads of the different VMs were dependent, the execution time of the workload when using 2 CPUs at 1.60 GHz and using 4 CPUs at 1.60 GHz would be similar. Pinning the VMs to 3 CPUs penalizes makespan by only 18.57 % (which is similar to the penalty when performing VM migration) but gives the highest energy efficiency. However, the reduction in temperature is lower than the one with VM migration. A higher reduction in temperature is achieved when pinning the VMs to 2 CPUs; however, in such scenario the makespan increases significantly and becomes comparable to running all CPUs at 2.13 GHz (which is the best case for DVFS). Pinning the VMs to only one CPU achieves a higher reduction in temperature but the penalty on both makespan and energy efficiency (due to resource sharing and the associated problems such as context switches) is not acceptable. Thus, the thresh-

Table 1 Experimental results using different VM management techniques

Technique - configuration	Makespan (s)	Energy (J)	EDP	Temp ↓
Reference (regular execution)	2,239 s	501,023 J	1,121,790,497	–
Migrate 1 VM – destination empty	+19.60 %	+52.82 %	+82.79 %	4 °C
Migrate 1 VM – destination full	+37.51 %	+31.92 %	+81.41 %	4 °C
Migrate 1 VM – destination off	+26.52 %	+57.40 %	+83.10 %	4 °C
Migrate 2 VMs – destination empty	**+14.15**%	**+51.86** %	**+73.36** %	**9 °C**
Migrate 2 VMs – destination full	+37.51 %	+30.92 %	+80.04 %	9 °C
Migrate 2 VMs – destination off	+21.08 %	+56.44 %	+73.68 %	9 °C
Migrate 3 VMs – destination empty	+16.43 %	+51.71 %	+76.65 %	15 °C
Migrate 3 VMs – destination full	+37.60 %	+25.89 %	+73.12 %	15 °C
Migrate 3 VMs – destination off	+23.35 %	+56.29 %	+76.96 %	15 °C
DVFS 4 CPUs @1.60 GHz	+78.38 %	+46.18 %	+160.76 %	8 °C
DVFS 4 CPUs @2.13 GHz	**+53.46** %	**+36.28** %	**+109.15** %	**4 °C**
DVFS 2 CPUs @1.60 GHz	+60.02 %	+51.14 %	+141.87 %	3 °C
Pinning VMs to 3 CPUs	**+18.57** %	**+16.23** %	**+37.83** %	**2.5 °C**
Pinning VMs to 2 CPUs	+55.69 %	+37.03 %	+113.35 %	6 °C
Pinning VMs to 1 CPU	+165.74 %	+108.89 %	+455.11 %	10°C

"*Destination empty*" means that the destination server is running but idle, "*destination full*" means that the destination server has 4 VMs running on the 4 CPUs, and "*destination off*" means that de destination server is switched off

old for applying the pinning technique is 2 CPUs for the HPL workload, which results in a temperature decrease of 6 °C. This temperature decrease may be sufficient to react to many moderate hotspots.

Overall, the obtained results show that, depending on the temperature reduction required to mitigate the effects of a hotspot and the optimization goals (i.e., performance or energy efficiency), VM migration and pinning are the most effective techniques. The results also show that when there are available servers to migrate VMs and the main objective is optimizing performance (i.e., minimizing the makespan), it may be better migrating VMs rather than using other techniques. However, when the focus is energy efficiency, pinning may be a preferable technique in favor of VM migration. Furthermore, when VM migration is not feasible, pinning is the most effective mechanism to reduce the server's temperature while balancing performance and energy efficiency.

4 Proactive VM Allocation Model and Implementation

The purpose of our proactive VM allocation approach is to increase the resource utilization and, hence, the energy efficiency. It aims at aims at maximizing the system throughput by allocating the maximum possible number of VMs per node, without penalizing the applications' performance. In other words, the objective is to find out the trade-off between the applications' performance and the overall datacenter energy consumption when different number and combinations of a variety of VMs are allocated to a physical server. The performance of an application is measured in terms of its *average execution time*, which is defined as the ratio of the maximum application execution time when a number of VMs are running simultaneously to the number of VMs. This metric gives an insight into the gains obtained by multiplexing VMs (i.e., running them in parallel) over running them sequentially one after the other. It is important to note that by considering the average execution time of VMs we do not focus on minimizing the execution time of each application individually but we strive to improve the QoS by minimizing the number of SLA violations.

As the best number of VMs per node may be different for different application types (based on their usage of different subsystems), we consider the applications' profiles. Specifically, we focus on finding the best partition and allocation of VMs when the different VMs run applications of different types. In our approach, we assume that the applications' profiles are known in advance (e.g., specified by the user in the job definition). To find the allocation for a set of VMs that best matches energy efficiency/performance goals while ensuring QoS guarantees, we rely on a model based on empirical data from experiments. We have developed a methodology composed of the following steps:

1. Profile a comprehensive set of applications (standard HPC benchmark workloads);
2. Run benchmarks exhaustively (all possible allocations based on number of VMs and application type) and collect data;
3. Create a model (database) with all the data collected during the benchmarking process, including execution time and energy consumed;
4. Implement an algorithm that, given the i) model, ii) an optimization goal (either minimize energy consumption or minimize execution time), iii) a set of servers with their current allocations, and iv) a set of VMs along with their characteristics, returns a set of partitions and allocations of the VMs in the servers.

4.1 Application Profiling

The methodology involves profiling an application's (and, hence, a VM's) behavior as *I/O-intensive, memory-intensive,* and/or *CPU-intensive* based on its usage of different subsystems. Most of the standard profiling utilities are designed for comparing computation efficiency of the applications on systems on which they are running and, therefore, their outputs are not very useful from the subsystem usage point of view. We profiled standard HPC benchmarks with respect to their behaviors and subsystem usage on individual servers. To collect run-time OS-level

metrics for CPU utilization, hard disk I/O, and network I/O we used different mechanisms such as "mpstat", "iostat", "netstat" or "PowerTOP" from Intel. We also patched the Linux kernel 2.6.18 with "perfctr" so that we can read hardware performance counters on-line with relatively small overhead. We instrumented the applications with PAPI and, as the server architecture does not support total memory LD/ST counter, we counted the number of L2 cache misses, which indicates (approximately) the activity of memory.

We chose a comprehensive set of HPC benchmark workloads. Each workload stresses one or more of the following subsystems—CPU, memory, disk (storage), and network interface. They can be classified as:

- **CPU intensive,** e.g., `HPL Linpack`, which solves a (random) dense linear system in double precision arithmetic, and `FFTW`, which computes the discrete Fourier transform.
- **Memory intensive,** e.g., `sysbench`, which is a multi-threaded benchmark developed originally to evaluate systems running a database under intensive load.
- **I/O intensive,** e.g., `b_eff_io`, which is a MPI-I/O application, and `bonnie++`, which focuses on hard-drive and file-system performance.

An application usually demands the services of a given subsystem in discrete time windows. However, if the average demand for a subsystem X is significant, we consider the application to be X-intensive. Figure 9 (left) shows different subsystem utilizations of a CPU-intensive workload. Note that an application can also be deemed to be intensive along multiple dimensions if the demand for resources from multiple subsystems are significant. Figure 9 (right) shows different subsystem utilizations of a network- and CPU-intensive workload. The utilization of a particular subsystem by two different VMs running applications may either overlap (resulting in contention) or not overlap (be contention free).

4.2 Benchmarking

The benchmarking was conducted in the experimental setup described in Section 3.3. In addition to using a single server type, we made some additional assumptions, such as a single process per VM, to reduce the complexity. To run multiple processes (e.g., MPI applications) multiple VMs are required.

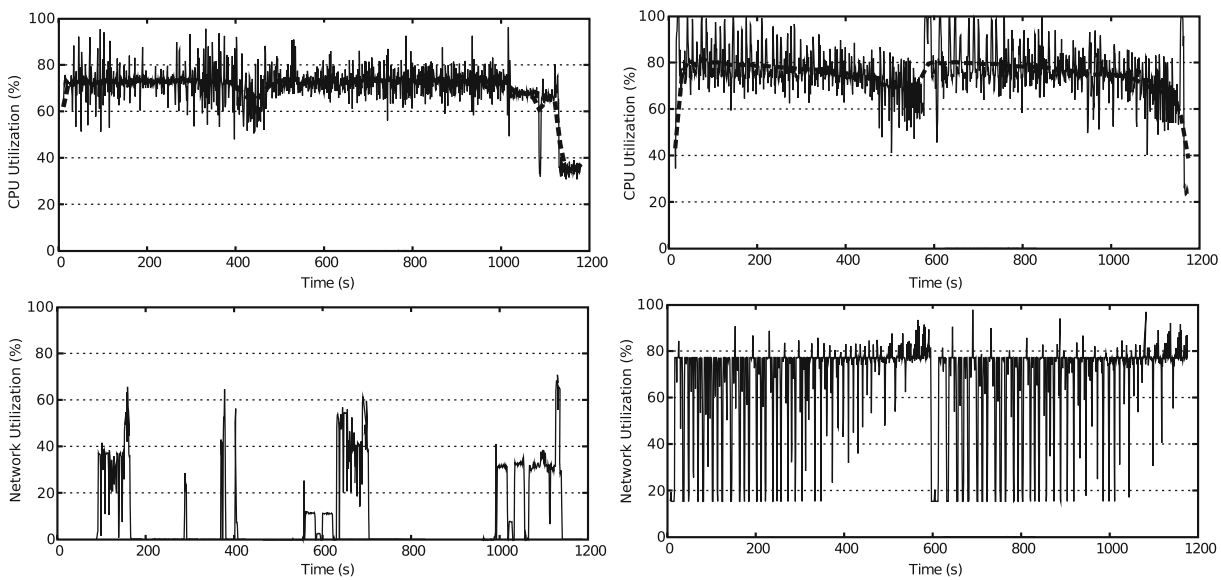


Fig. 9 Sub-system utilization over time for a CPU-intensive workload (*left*) and a CPU-plus-network-intensive workload (*right*)

In order to acquire sufficient data to create a VM allocation model, firstly, we conducted a set of base tests that consolidate different VM instances running applications of the same type in a single server. This allowed us to find out, on the one hand, the optimal scenarios to either maximize performance or minimize energy consumption and, on the other hand, the maximum number of VMs that can be consolidated in a single server without adversely impacting energy consumption and the applications' performance. We ran the base experiments with different number of VMs (up to 16) running the same application type for each of the application's profiles.

From the base tests, we obtained a set of optimal scenarios, i.e., optimal number of VMs for the shortest average execution times and for minimum energy consumption. The second part of the benchmarking consists of running all the possible combinations of workload types with different number of VMs. The combinations excluded those that do not require any VM of each workload type and the base tests. The experiments took several days to be completed and they were conducted using a platform that we developed to automatically run the benchmarks and process the data.

4.3 Database

In order to make our model available for proactive VM consolidation, the information collected from the benchmarking (base and combined tests) was stored in a database. As the amount of information was manageable using text files, we used a plain-text file with comma-separated values (CSV) instead of an actual database management system. Table 2 summarizes the information contained in the database.

In addition to the information listed in Table 2, we store other relevant information from the base experiments such as the number of VMs of optimal scenarios and reference execution times, in an auxiliary file. As the registers of the database are accessed using binary search, the searching cost is $\mathcal{O}(log(num\_tests))$. Therefore, we sorted (in the ascending order) the registers of the database by a searching key, which is composed of the parameters that indicate the number of VMs of each workload type (`Ncpu`, `Nmem`, `Nio`).

Table 2 Summary of the information stored in the database

Field	Description
`Ncpu`	#VMs running a CPU-intensive benchmark
`Nmem`	#VMs running a memory-intensive benchmark
`Nio`	#VMs running an I/O-intensive benchmark
`relTimeVM` (for each VM)	Relative execution time for each VM (relative time = total exec time/exec time on 1 VM)
`Time`	Total execution time of the outcome (s)
`Energy`	Energy consumed to run the outcome (J)
`MaxPower`	Maximum power dissipation measured (W)
`EDP`	Energy delay product (J × s)

4.4 VM Allocation Algorithm

Our VM allocation algorithm takes advantage of the model (in the form of a database) that is described above. It has two main objectives, (i) minimize energy consumption and (ii) maximize the performance (i.e., minimize workload execution time). As these are two conflicting objectives, we use a parameter α to adjust the possible trade-off between energy efficiency and performance; α is defined as follows: $\alpha \in \mathcal{R} \cap \alpha \in (0, \ldots, 1)$ and emphasizes the energy efficiency goal while $1 - \alpha$ emphasizes performance. For example, if $\alpha = 0.7$ the algorithm will try to minimize the energy consumption first (70 % of preference) and then the performance but with less intensity (30 % of preference). The allocation algorithm focuses on the objectives discussed above and does not consider specific policies such as those based on priorities. The input parameters of the algorithm are: (i) the database with the allocation model, (ii) values from the base experiments (can be extracted from the auxiliary file), (iii) a set of VMs and the application's profile and maximum execution time (QoS guarantees) for each of them, and (iv) the optimization goal (α).

First, as mentioned earlier, the algorithm sets up the knowledge database by loading it in memory and sorting it using a three-column key (e.g. `Ncpu`, `Nmem` and `Nio`) in order to accelerate the search. Then, the estimated runtime and energy for each combination of the VMs are calculated and the minimum of them is taken. Note that, if there are N VMs in the input set, it can have B_N partitions, where B_N is the Bell number. An exhaustive search over these B_N distinct partitions is guaranteed to provide an optimal solution. Each partition of the set contains several disjoint subsets, and each of these subsets contains a certain number of CPU-, memory- and/or I/O-intensive VMs. The number of CPU-, memory- and I/O-intensive VMs in the input are used to search the database entries associated to them. The estimated execution time of a subset is computed by multiplying the original input runtime (which is an input parameter) by `relTimeVM`. Similarly, the estimated energy consumption of a subset is computed by multiplying the estimated runtime by average power (also stored in the database). The computational complexity of the whole algorithm

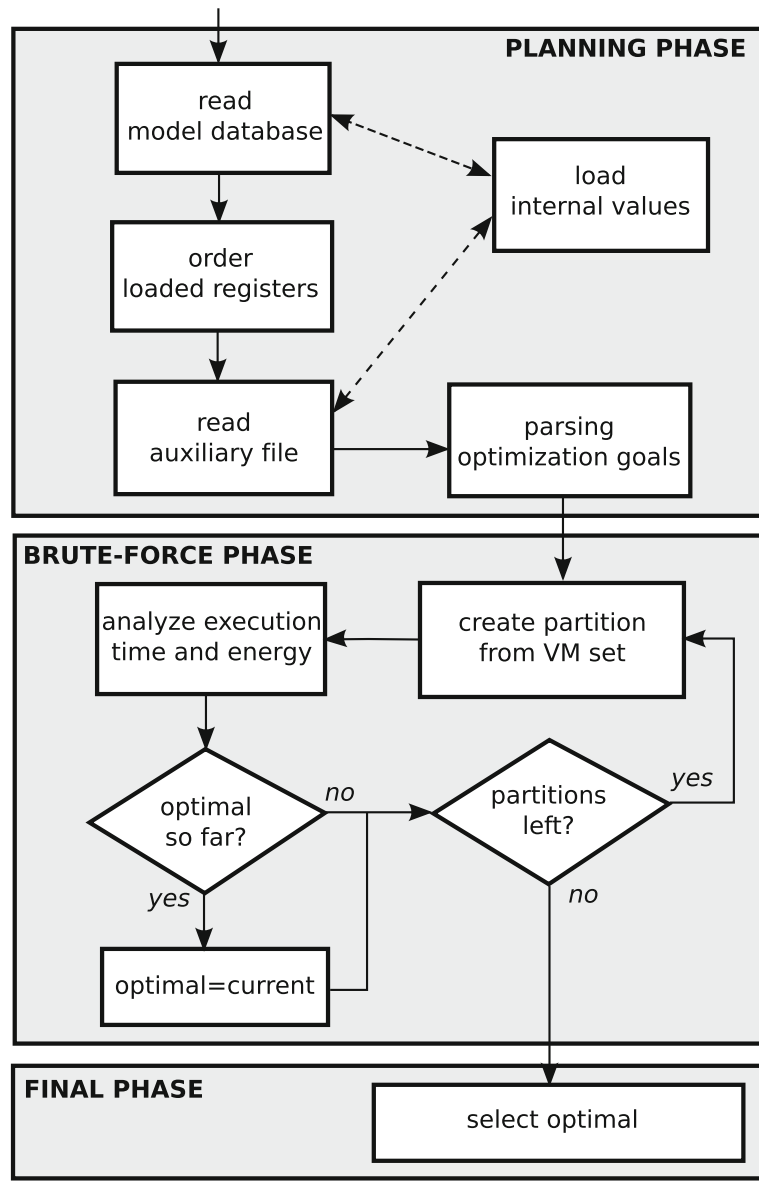


Fig. 10 VM allocation algorithm

is $\mathcal{O}(B_N \cdot log(N_{DB}))$, where N is the number of VMs to allocate and N_{DB} is the number of elements in the knowledge database. The algorithm returns the allocation of VMs that best matches the input optimization goal while satisfying the QoS constraints. The algorithm can be relaxed by disregarding the QoS guarantees but it might be not acceptable for production system.

To find the best partitions of the input set of VMs for allocation in individual servers, we used a brute-force search algorithm over the servers with their current VM allocations. In cases where the number of partitions of the input set was large, we used the search algorithm discussed in [36], which is efficient in terms of complexity. If two partitions have the same rank in different servers, we select the first server of the list. Figure 10 shows the main components and control flow of our VM allocation algorithm.

5 Evaluation

In this section, we discuss the performance of our proposed proactive VM allocation and reactive thermal management strategies. Firstly, we evaluated the possible energy savings and performance tradeoffs that can be achieved at the datacenter level using our proactive VM allocation algorithm described in the previous section. We conducted the simulations using parallel workload traces from real HPC production systems. Then, we conducted an experimental evaluation of our proposed reactive thermal management solution and compared it with other existing techniques.

5.1 Simulation

To evaluate the performance and energy efficiency of our VM allocation algorithm we used workload traces from real HPC production systems like the Grid Observatory [13], which collects, publishes, and analyzes logs of the behavior of the EGEE Grid [10]. As the readily available traces are in different formats and include data that are not useful for our purpose, they were pre-processed before use in the simulations. Firstly, we converted the input traces to the Standard Workload Format (SWF) [11] and as they are usually composed of multiple files we combined them into a single file. Then, we cleaned the traces, now in SWF format, in order to eliminate failed jobs, cancelled jobs, and anomalies. As the traces found from different systems did not provide all the information needed for our analysis, we needed to complete them using a model based on the benchmarking of HPC applications (see Section 4). We randomly assigned one of the four possible benchmark profiles to each request in the input trace, following a uniform distribution by bursts. The bursts of job requests were sized (randomly) from 1 to 5 job requests. These traces are intended to illustrate the submission of scientific HPC workflows, which are composed of sets of jobs with the same resource requirements.

As the EGEE Grid is a large-scale heterogeneous distributed system composed of a large number of nodes, we scaled and adapted the job requests to the characteristics of our system model and evaluation methodology. Specifically, we assigned 1 to 4 VMs per job request rather than the original CPU demand and we defined the QoS requirements (maximum response time) per application type and not for each specific request.

Data obtained from real experiments on representative server hardware were used to populate the database and to create the empirical model described in the previous section. Therefore, in our simulations we used a system model composed of several servers with the same characteristics of our real server testbed. To compute the estimated execution times and energy consumption we used the information from our allocation model. Given a specific partition with a subset of VMs running their associated applications types, we lookup our model database and use the matching values proportionally. As VM allocations may vary over time, we compute the estimated execution time and energy consumption with the weighted average of the values associated to each interval of time. We also assume a fixed power dissipation of 125 W when a server is idle.

Figure 11 illustrates a possible VM allocation outcome over time for a server. The application type associated with each VM is also shown. Different time intervals (`A`, `B`, `C`) have different VM allocations and, therefore, the estimated ex-

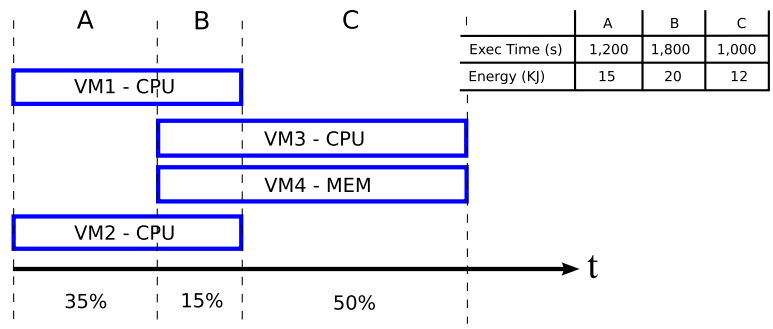


Fig. 11 Possible VM allocation outcome over time

ecution time of the applications and energy consumption for each interval will be different. For example, the execution time of `VM1` will be computed considering the relative weight of each allocation (70 % of allocation `A` and 30 % of allocation `B`) as follows: $ExecTime_{VM1} = 0.7 \cdot 1200\ \text{s} + 0.3 \cdot 1800\ \text{s} = 1380\ \text{s}$ and the energy consumption for the whole outcome will be: $Energy = 0.35 \cdot 15\ \text{KJ} + 0.15 \cdot 20\ \text{KJ} + 0.5 \cdot 12\ \text{KJ} = 14.25\ \text{KJ}$. As we focus on studying the impact of VM allocation on the performance/energy efficiency, we do not consider the overhead for scheduling and resource provisioning.

We evaluate the impact of our approach in terms of the following metrics: *makespan* (in seconds), which is the difference between the earliest time of submission of any of the workload tasks and the latest time of completion of any of its tasks, *energy consumption* (in Joules), and percentage of SLA violations. The number of SLA violations were calculated by summing the number of missed deadlines of all applications. The deadline here refers to the maximum response time as specified by the QoS requirements.

We have conducted our simulations using different allocation strategies that have different goals. Specifically, we have evaluated the following allocation strategies:

- FIRST-FIT (FF), in which job requests are allocated following the first-fit policy based on CPU slots. It means that an incoming job request is allocated to the first available server until the number of allocated VMs is equal to the number of CPUs (VM multiplexing on CPUs is not allowed). FIRST-FIT-2 (FF-2) and FIRST-FIT-3 (FF-3) are two variants of FIRST-FIT that allow multiplexing up to 2 and 3 VMs on each CPU, respectively.
- Proactive (PA), in which job requests are allocated to servers following the algorithm described in Section 4. We consider the following variations:
  - $\alpha = 1$ (PA-1): the goal is minimizing the energy consumed;
  - $\alpha = 0$ (PA-0): the goal is minimizing the execution time;
  - $\alpha = 0.5$ (PA-0.5): the goal is finding the best tradeoff between execution time and energy consumption. Note that, although existing literature have addressed the optimization of both metrics using dynamic concurrency throttling over parallel regions [8], in this paper we assume that energy efficiency and performance are, in general, conflicting goals as we have shown in our virtualized scenario.

Figures 12, 13 and 14 show the results obtained using different VM allocation strategies for handling the workloads traces described previously. Furthermore, in order to control the pressure of the system load, we modeled two different Clouds of different sizes rather than using different input traces with different arrival rates. The SMALLER Cloud system is the reference one and the LARGER Cloud system is overdimensioned (15 % approximately), which means that the former one is expected to be more loaded than the latter. The input trace used in the simulations requests a total of 10,000 VMs.

As we can observe in Fig. 12, the PA strategy can provide up to 18 % shorter execution times. This is due to the fact that application awareness

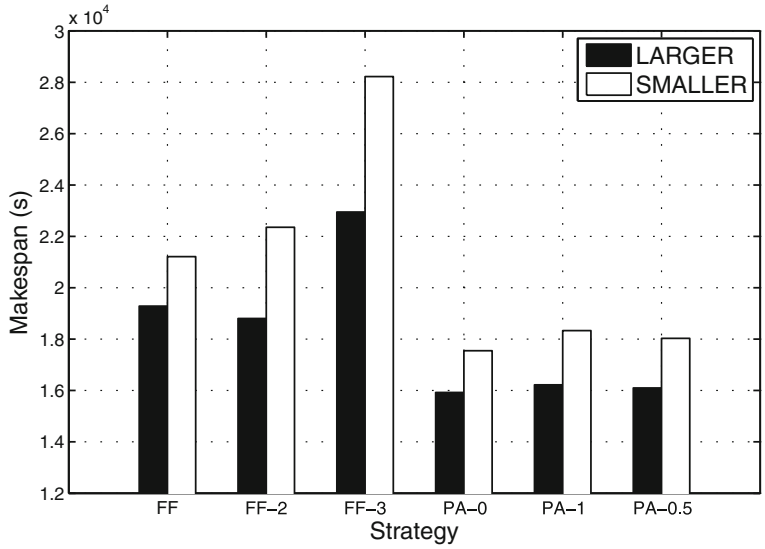


Fig. 12 Makespan (s)

results in fewer contentions for resources as only the most compatible VMs are consolidated. With the FIRST-FIT strategy the execution times are longer due to high resource contention, especially when multiplexing 3 VMs on the same CPU. Furthermore, Fig. 12 shows that the PA strategy with the performance optimization goal reduces the execution times by more than 3 % in comparison to the same strategy with the energy optimization goal. We can also appreciate that the execution times in the SMALLER system are higher than the execution times in the LARGER system due to higher load pressure. This is especially evident in the case of FF-3, FIRST-FIT strategy with multiplexing (up to 3 VMs per CPU), due to possible additional resource contention.

Figure 13 shows that the PA strategy reduces energy consumption by around 12 % on average with respect to FIRST-FIT (with and without VM multiplexing). In fact, makespan and energy consumption follow a similar pattern in the LARGER system. Furthermore, Fig. 13 shows that the PA strategy with the energy optimization goal saves almost 3 % more energy than the

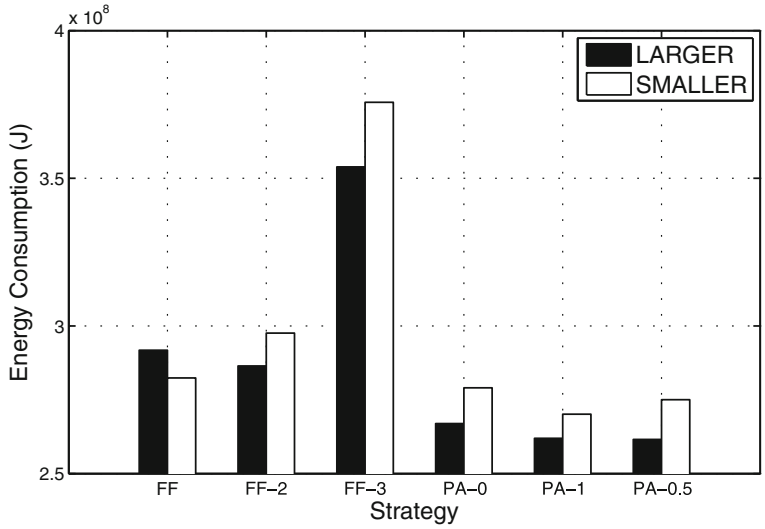


Fig. 13 Energy consumption (J)

same strategy with the performance optimization goal. However, with the goal of finding the best tradeoff it provides intermediate results (but the variations are not very significant, i.e., <2 %). Although the makespan in the SMALLER system is higher than the makespan in the LARGER system, the energy consumption in the SMALLER system is lower than the energy consumption in the LARGER system as in the SMALLER system there are fewer servers consuming energy and there are more opportunities for consolidation. However, with the FIRST-FIT strategy the resource contention penalizes the energy efficiency significantly when multiplexing of 2 or 3 VMs is allowed on the same CPU.

Figure 14 shows that the percentage of SLA violations with the PA strategies is also less compared to the traditional schemes. It means that the PA strategy can maintain or even provide better QoS guarantees than the traditional approaches. Furthermore, we can observe in Fig. 14 a correlation between execution time and SLA violations, the higher the makespan higher the percentage of SLA violations. We also can appreciate that the strategies evaluated present similar behaviors under higher load conditions. We do not show in this article the results obtained with other possible configurations of the PA strategy (e.g., $\alpha = 0.75$) since the variation in the results was not significant enough.

5.2 Validation

We performed experiments on real hardware in order to validate our proposed approach for energy-efficient reactive thermal management and to compare them with other existing VM management strategies. We considered 50 °C as thermal emergency temperature. Although existing literature considers emergency temperature of the datacenter at 105–135 °F (40–57 °C) [12] and per-component red line temperature to 72 °C for the CPU and 67 °C for the disk [38], we determined experimentally that there were hardware failures when the servers operating temperature raised above this threshold. Specifically, we evaluated the following thermal management strategies:

- FIRST-FIT (FF), in which job requests are allocated following the first-fit policy based on CPU slots as the one used in our simulations. This strategy provides an insight into the worst performance (lower bound) when a naive VM allocation strategy is followed without any regard to the thermal behavior of the underlying hardware.
- RANDOM (RND), in which job requests are allocated to servers randomly. Servers with least load (smallest number of VMs running on them) are chosen with a high probability to host the VMs corresponding to the current set

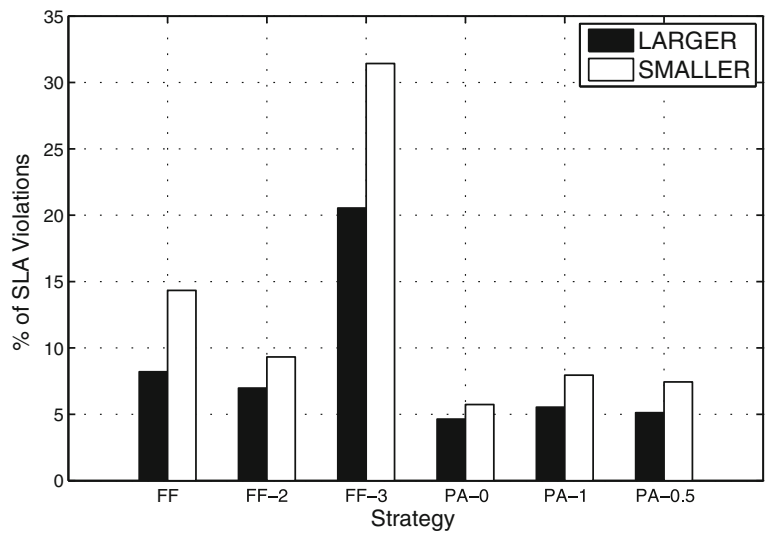


Fig. 14 Percentage of SLA violations

of job requests. This strategy spreads the job requests uniformly across all servers but may not guarantee a uniform thermal behavior.
- TEMPERATURE-AWARE (TA), in which the "coolest" among all servers whose operating temperatures are within a pre-specified "safety" threshold (i.e., under the threshold of 50 °C) is chosen to host the VMs corresponding to the current set of job requests. This strategy spreads the thermal load uniformly across all servers but does not ensure a uniform load distribution (in terms of number of job requests allocated per server).
- VM-MIGRATION, in which the VMs are initially allocated based on the FF approach. When a server reaches unsafe operating temperatures or thermal hotspots (regions that are more than 50 °C in temperature in the testbed used) are detected, VMs are migrated and again the FF approach is used to determine where to reallocate the VMs.
- ENERGY-AWARE THERMAL MANAGEMENT (EATM), in which VMs are initially allocated based on the proactive model described in Section 4, the cross-layer reactive thermal management described in Section 3 is applied when a server reaches a unsafe temperature or thermal hotspots are detected the proactive VM allocation model is again used to determine where to migrate the VMs (when migration is chosen as the most appropriate reaction to the thermal anomaly).

In all the strategies we limit the multiplexing factor (i.e., the number of VMs that can run simultaneously on a physical CPU) to 2 VMs/CPU. The experiments were conducted on a virtualized cluster composed of 8 of the nodes described in Section 3.1, stacked in one rack with Xen hypervisor. A *"Watts Up? .NET"* power meter was used for instantaneous power consumption measurements as described in Section 3.3 and internal sensors were used for detecting thermal anomalies. The cluster was operated in a poorly cooled environment and no cooling system optimization was done simultaneously. We used a workload based on the one described in Section 5.1 but scaled to the size of the testbed (providing an average load of around 75 %, which is a typical target for HPC virtualized clusters).

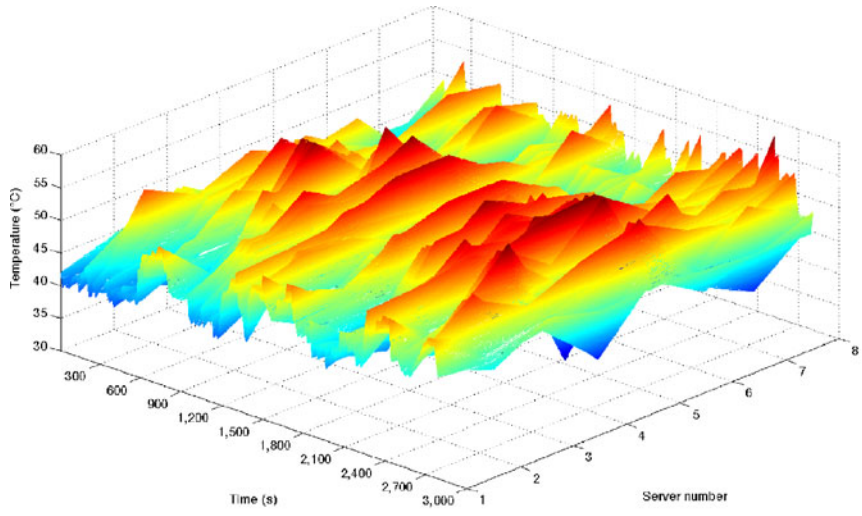
Table 3 shows the experimental results for the different strategies described above."%Unsafe" refers to the fraction of time that servers' operating temperature was higher than 50 °C (aggregate of all the servers's "unsafe" fractions of time). Figures 15 and 16 show the temperature in °C of each server over time and power dissipation of the cluster over time, respectively, for three representative strategies (i.e., RND, TA and EATM).

The overall results show that EATM outperforms the other strategies. EATM provides shorter makespan (2.25 % and 12 % shorter than FF's and TA's, respectively), lower energy consumption (around 9 % lower than both FF's and TA's) and a lower EDP. We can appreciate that TA achieves lower operating temperatures than FF does but results in longer a makespan resulting in higher energy consumption and, hence higher EDP. This is because when all the available servers' operating temperatures are higher than the safety threshold jobs cannot be scheduled and are blocked until the temperatures reduce.

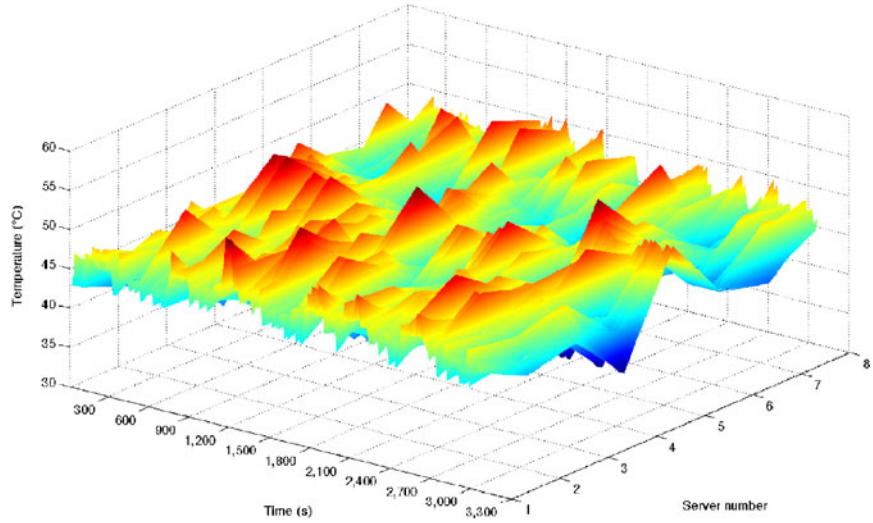
VM-MIGRATION technique outperforms FF, RND, and TA strategies. However, it results in a longer makespan and higher energy footprint than EATM. This is due to the computation overhead of VM migrations and the fact that the opportunities for VM migrations are low when the sys-

Table 3 Experimental results using the different thermal management strategies

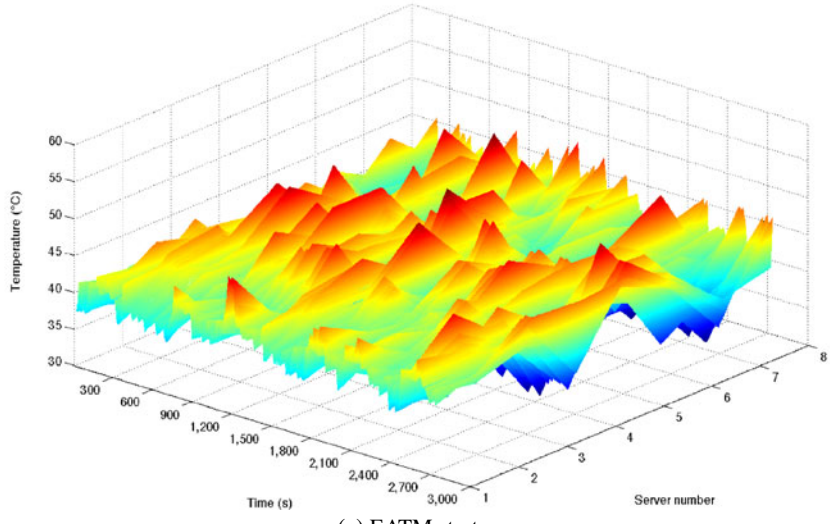
Strategy	Makespan (s)	Energy (KJ)	EDP $\times 10^6$	Power (watts)			Temperature (°C)			% Unsafe
				min	avg	max	min	avg	max	
FIRST-FIT	2,994	2,487	7,341	663	835	1,093	37.67	48.27	59.14	16.07
RANDOM	2,960	2,459	7,232	667	833	1,058	35.27	47.31	57.50	18.62
THERMAL-AWARE	3,282	2,502	8,211	679	817	1,050	32.45	45.13	54.12	7.25
VM-MIGRATION	3,180	2,407	7,654	482	768	1,067	29.50	44.61	57.74	9.12
EATM	2,928	2,281	6,719	662	781	942	32.17	43.17	52.64	6.87



(a) RANDOM strategy



(b) THERMAL-AWARE strategy



(c) EATM strategy

Fig. 15 Temperature (in °C) of each server over time

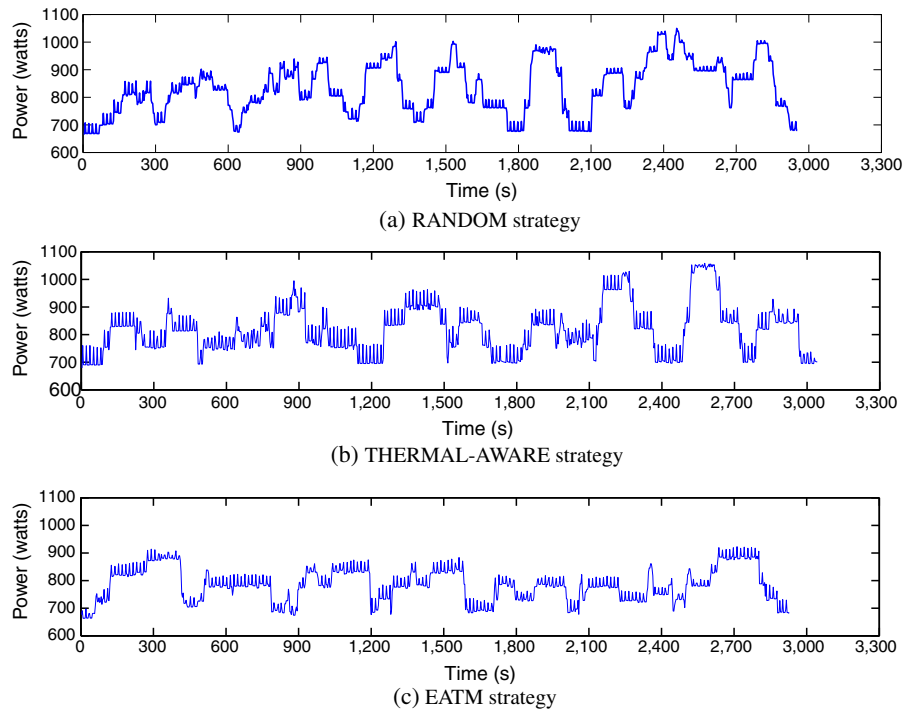


Fig. 16 Cluster power dissipation over time

tem is already heavily loaded. The average power consumption and minimum average temperature is lower for VM migration with respect to EATM due to the periods in which unused servers (which are only a few and for a very short duration) are switched off. VM-MIGRATION may work better for lightly loaded scenarios where the probability of having idle servers is high. However, we believe that EATM would be more effective from an holistic perspective. Note that, in this paper, we focus on HPC workloads and, therefore, VM migration overheads are specific to such scenarios and cannot be generalized for all types of workloads.

The average power dissipation while using EATM is 7 % and 5 % lower than those incurred by FF/RND and TA, respectively, but larger than the one incurred by VM-MIGRATION. However, EATM provides lower maximum power dissipation with respect to VM-MIGRATION due to the overhead of migrating VMs when the temperature is high. The maximum power dissipation is also significantly lower for EATM with respect to the other strategies (e.g., 16 % and 11 % lower than those incurred by FF and TA, respectively).

EATM provides a better thermal behavior (i.e., 10 % and 2 % lower average temperature compared to FF and TA, respectively and significantly lower maximum temperature) as it does react to thermal anomalies and also consolidates better the workload using an application-centric approach. In contrast to EATM, the TA policy reduces the maximum temperature but does not control the average temperature because it does nothing to mitigate thermal hotspots (only avoid new job requests in the server), which results in higher average temperature with respect to EATM. It can be clearly seen in Fig. 16c that the hottest servers are those in the middle of the rack due to heat propagation and probably some affinity of workloads to these servers in the policy implementation. We can also observe some clear thermal hotspots with temperatures up to 57 °C. As far as the thermal behavior of servers when using the TA strategy is concerned, higher temperature zones are uniformly spread and thermal hotspots are of lower intensity. Similarly, while using EATM, higher temperature zones are uniformly spread but these temperatures are lower compared to those achieved with TA and are well under

the safety threshold (due to reactive thermal management).

The fraction of time that servers run in unsafe conditions while using EATM is much shorter than the ones achieved while using FF (less than 50 %) and TA (less than 5 %), which indicates that EATM achieves the objective of reducing the probability of hardware failure while providing a positive tradeoff between performance and energy.

6 Conclusions and Future Work

In this article, we presented and evaluated i) a reactive cross-layer thermal management solution, which alleviates undesired thermal anomalies (i.e., hotspots) in virtualized HPC cloud infrastructure and ii) a proactive application-centric strategy for VM allocation, which aims at maximizing the resource utilization and energy efficiency while satisfying quality of service guarantees for HPC applications. The reactive thermal management solution considers the tradeoffs among performance, energy efficiency, and thermal efficiency while using different mechanisms such as VM migration, DVFS, and CPU pinning to alleviate thermal anomalies. The VM allocation algorithm leverages an empirical model for the average energy consumption and execution time derived from measurements on real hardware running HPC workloads.

The results obtained from simulations using real production HPC workload traces show that our proactive VM allocation solution significantly contributes to energy efficiency (12 % reduction in energy consumption) and/or optimization of the application performance (18 % reduction in execution time) depending on optimization goals. Our reactive Energy-Aware Thermal Management (EATM) solution in conjunction with the aforementioned allocation approach outperforms other traditional thermal management approaches like load redistribution (VM migrations) and Temperature-Aware (TA) VM placement in terms of energy consumption (up to 9 % less than that of TA's), makespan (up to 12 % less than that of TA's), maximum server operating temperatures (up to 10 % less than that achieved by the load redistribution technique), and percentage of time for which servers operate in unsafe temperatures (up to 25 % less than that achieved by the load redistribution technique). As an extension of the work presented in this article, we are currently working on a proactive VM allocation model that uses the concept of heat imbalance (which is the difference between heat generated by servers and the heat extracted by the cooling system) to project future temperatures and to place workloads on servers. Also, this paper deals with physical resources (testbed of servers) that belong to the same datacenter. As part of our study and validation of proactive thermal management solutions (such as the aforementioned VM allocation model), we are conducting experiments on a heterogeneous testbed (in terms of computing as well as cooling resources) spanning over two sites of NSF CAC at Rutgers University and the University of Florida.

Acknowledgements The research presented in this work is supported in part by National Science Foundation (NSF) via grants numbers IIP 0758566, CCF-0833039, DMS-0835436, CNS 0426354, IIS 0430826, CNS 0723594 and CSR-1117263, by the Department of Energy ExaCT Combustion Co-Design Center via subcontract number 4000110839 from UT Battelle and via the grant numbers DE-SC0007455 and DE-FG02-06ER54857, and by an IBM Faculty Award, and was conducted as part of the NSF Cloud and Autonomic Computing Center at Rutgers University.

References

1. Report to congress on server and data center energy efficiency. Tech. rep., U.S. Environmental Protection Agency (2007)
2. Ajiro, Y., Tanaka, A.: Improving packing algorithms for server consolidation. In: Proc. of Computer Measurement Group Conf. (CMG), San Diego, CA, pp. 399–406 (2007)
3. Apparao, P., Iyer, R., Zhang, X., Newell, D., Adelmeyer, T.: Characterization & analysis of a server consolidation benchmark. In: Proc. of ACM SIGPLAN/SIGOPS Conf. on Virtual Execution Environments (VEE), Seattle, WA, pp. 21–30 (2008)
4. Bash, C., Forman, G.: Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In: Proc. of USENIX Annual Technical Conf. (ATEC), Santa Clara, CA, pp. 363–368 (2007)
5. Beitelmal, A., Patel, C.: Thermo-fluids provisioning of a high performance high density data center. Distrib. Parallel Dat. **21**(2–3), 227–238 (2007)

6. Bobroff, N., Kochut, A., Beaty, K.: Dynamic placement of virtual machines for managing SLA violations. In: Proc. of IFIP/IEEE Symp. on Integrated Network Management (IM), Munich, Germany, pp. 119–128 (2007)
7. Chen, Y., Das, A., Qin, W., Sivasubramaniam, A., Wang, Q., Gautam, N.: Managing server energy and operational costs in hosting centers. In: Proc. of ACM Intl. Conf. on Measurement and Modeling of Computer Systems (SIGMETRICS), Banff, Canada, pp. 303–314 (2005)
8. Curtis-Maury, M., Shah, A., Blagojevic, F., Nikolopoulos, D.S., de Supinski, B.R., Schulz, M.: Prediction models for multi-dimensional power-performance optimization on many cores. In: Proc. of the 17th Intl. Conf. on Parallel Architectures and Compilation Techniques, pp. 250–259 (2008)
9. Das, R., Kephart, J.O., Lefurgy, C., Tesauro, G., Levine, D.W., Chan, H.: Autonomic multi-agent management of power and performance in data centers. In: Proc. of Intl. joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS), Estoril, Portugal, pp. 107–114 (2008)
10. Enabling Grid for E-sciencE (2010). http://www.eu-egee.org/. Accessed 1 Oct 2011
11. Feitelson, D.: Parallel workload archive (2010). http://www.cs.huji.ac.il/labs/parallel/workload/. Accessed 1 Oct 2011
12. Garday, D., Housley, J.: Thermal storage system provides emergency data center cooling. Tech. rep., Intel Corporation (2007)
13. Grid Observatory (2010). http://www.grid-observatory.org/. Accessed 1 Oct 2011
14. Gonzalez, R., Horowitz, M.: Energy dissipation in general purpose microprocessors. IEEE J. Solid-State Circuits **31**(9), 1277–1284 (1996)
15. Govindan, S., Nath, A.R., Das, A., Urgaonkar, B., Sivasubramaniam, A.: Xen and co.: communication-aware CPU scheduling for consolidated Xen-based hosting platforms. In: Proc. of the Intl. Conf. on Virtual Execution Environments (VEE), San Diego, CA, pp. 126–136 (2007)
16. Greenberg, S., Mills, E., Tschudi, B.: Best practices for data centers: lessons learned from benchmarking 22 data centers. In: Proc. of American Council for an Energy-Efficient Economy (ACEEE), Pacific Grove, CA (2006)
17. Gupta, D., Lee, S., Vrable, M., Savage, S., Snoeren, A.C., Varghese, G., Voelker, G.M., Vahdat, A.: Difference engine: harnessing memory redundancy in virtual machines. Commun. ACM **53**(10), 85–93 (2010)
18. Heath, T., Centeno, A.P., George, P., Ramos, L., Jaluria, Y., Bianchini, R.: Mercury and freon: temperature emulation and management for server systems. In: Proc. of Intl. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS), San Jose, CA, pp. 106–116 (2006)
19. Hermenier, F., Lorca, X., Menaud, J.M., Muller, G., Lawall, J.: Entropy: a consolidation manager for clusters. In: Proc. of the ACM SIGPLAN/SIGOPS Conf. on Virtual Execution Environments (VEE), Washington, DC, pp. 41–50 (2009)
20. Kang, S., Schmidt, R.R., Kelkar, K., Patankar, S.: A methodology for the design of perforated tiles in raised floor data centers using computational flow analysis. IEEE Trans. Compon. Packag. Technol. **24**(2), 177–183 (2001)
21. Kaxiras, S., Martonosi, M.: Computer Architecture Techniques for Power-Efficiency. Morgan and Claypool (2008)
22. Kephart, J.O., Chan, H., Das, R., Levine, D.W., Tesauro, G., Rawson, F., Lefurgy, C.: Coordinating multiple autonomic managers to achieve specified power-performance tradeoffs. In: Proc. of Intl. Conf. on Autonomic Computing (ICAC), Jacksonville, FL, pp. 24–34 (2007)
23. Kochut, A., Beaty, K.: On strategies for dynamic resource management in virtualized server environments. In: Proc. of the Intl. Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Istanbul, Turkey, pp. 193–200 (2007)
24. Kumar, S., Talwar, V., Kumar, V., Ranganathan, P., Schwan, K.: vManage: loosely coupled platform and virtualization management in data centers. In: Proc. of Intl. Conf. on Autonomic Computing (ICAC), Barcelona, Spain, pp. 127–136 (2009)
25. Laszewski, G., Wang, L., Younge, A.J., He, X.: Power-aware scheduling of virtual machines in DVFS-enabled clusters. In: Proc. of IEEE Intl. Conf. on Cluster Computing (CLUSTER), New Orleans, LA, pp. 1–10 (2009)
26. Lee, E.K., Kulkarni, I., Pompili, D., Parashar, M.: Proactive thermal management in green datacenter. J. Supercomput. **51**(1), 1–31 (2010)
27. Liu, J., Priyantha, B., Zhao, F., Liang, C., Wang, Q., James, S.: Towards discovering data center genome using sensor networks. In: Proc. of the Workshop on Embedded Networked Sensors (HotEmNets), Charlottesville, VA (2008)
28. Mannas, E., Jones, S.: Add thermal monitoring to reduce data center energy consumption (2009). http://pdfserv.maxim-ic.com/en/an/AN4334.pdf. Accessed 1 Oct 2011
29. Menasce, D.A., Bennani, M.N.: Autonomic virtualized environments. In: Proc. of Intl. Conf. on Autonomic and Autonomous Systems (ICAS), Santa Clara, CA, pp. 1–28 (2006)
30. Meng, X., Isci, C., Kephart, J., Zhang, L., Bouillet, E., Pendarakis, D.: Efficient resource provisioning in compute clouds via VM multiplexing. In: Proc. of Intl. Conf. on Autonomic Computing (ICAC), Washington, DC, pp. 11–20 (2010)
31. Moore, J., Chase, J., Ranganathan, P., Sharma, R.: Making scheduling "cool": temperature-aware workload placement in data centers. In: Proc. of USENIX Annual Technical Conf. (ATEC), pp. 61–75 (2005)
32. Moore, J.D., Chase, J.S., Ranganathan, P.: Weatherman: automated, online and predictive thermal mapping and management for data centers. In:

Proc. of Intl. Conf. on Autonomic Computing (ICAC), Dublin, Ireland, pp. 155–164 (2006)
33. Mukherjee, T., Banerjee, A., Varsamopoulos, G., Gupta, S., Rungta, S.: Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers. Comput. Networks **53**(17), 2888–2904 (2009)
34. Nathuji, R., Isci, C., Gorbatov, E.: Exploiting platform heterogeneity for power efficient data centers. In: Proc. of Intl. Conf. on Autonomic Computing (ICAC), Jacksonville, FL, pp. 1–5 (2007)
35. Nathuji, R., Schwan, K.: VirtualPower: coordinated power management in virtualized enterprise systems. In: Proc. of ACM SIGOPS Symp. on Operating Systems Principles (SOSP), pp. 265–278 (2007)
36. Orlov, M.: Efficient generation of set partitions. Tech. rep., Engineering and Computer Sciences, University of Ulm (2002)
37. Patel, C., Bash, C., Belady, L., Stahl, L., Sullivan, D.: Computational fluid dynamics modeling of high compute density data centers to assure system inlet air specifications. In: Proc. of Pacific Rim/ASME International Electronic Packaging Technical Conferenceof (IPACK), Kauai, HI (2001)
38. Rambo, J., Joshi, Y.: Modeling of data center airflow and heat transfer: state of the art and future trends. Distrib. Parallel Dat. **21**(2–3), 193–225 (2007)
39. Ramos, L., Bianchini, R.: C-oracle: predictive thermal management for data centers. In: Proc. of Intl. Symp. on High-Performance Computer Architecture (HPCA), pp. 111–122 (2008)
40. Ranganathan, P., Leech, P., Irwin, D., Chase, J.: Ensemble-level power management for dense blade servers. SIGARCH Comput. Archit. News **34**(2), 66–77 (2006)
41. Rodero, I., Chandra, S., Parashar, M., Muralidhar, R., Seshadri, H., Poole, S.: Investigating the potential of application-centric aggressive power management for HPC workloads. In: Proc. of the IEEE Intl. Conf. on High Performance Computing (HiPC), Goa, India, pp. 1–10 (2010)
42. Rodero, I., Lee, E.K., Pompili, D., Parashar, M., Gamell, M., Figueiredo, R.J.: Exploiting VM technologies for reactive thermal management in instrumented datacenters. In: Workshop on Energy Efficient Grids, Clouds, and Clusters in conjunction with IEEE Grid, Brussels, Belguim, pp. 321–328 (2010)
43. Rusu, C., Ferreira, A., Scordino, C., Watson, A.: Energy-efficient real-time heterogeneous server clusters. In: Proc. of IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS), St. Louis, MO, pp. 418–428 (2006)
44. Schmidt, R.R., Cruz, E.: Raised floor computer data center: effect on rack inlet temperatures of exiting both the hot and cold aisle. In: Proc. of Itherm Conference (ITHERM), San Diego, CA (2002)
45. Schmidt, R.R., Cruz, E.E., Iyengar, M.K.: Challenges of data center thermal management. IBM J. Res. Develop. **49**(4/5), 709–723 (2005)
46. Schmidt, R.R., Karki, K., Kelkar, K., Radmehr, A., Patankar, S.: Measurements and predictions of the flow distribution through perforated tiles in raised floor data centers. In: Proc. of Pacific Rim/ASME International Electronic Packaging Technical Conferenceof (IPACK), Kauai, HI (2001)
47. Sharma, R., Bash, C., Patel, R.: Dimensionless parameters for evaluation of thermal design and performance of large-scale data centers. In: Proc. of ASME/AIAA Joint Thermophysics and Heat Transfer Conference. St. Louis, MO (2002)
48. Sharma, R.K., Bash, C.E., Patel, C.D., Friedrich, R.J., Chase, J.S.: Balance of power: dynamic thermal management for internet data centers. IEEE Internet Comput. **9**(1), 42–49 (2005)
49. Song, Y., Sun, Y., Wang, H., Song, X.: An adaptive resource flowing scheme amongst VMs in a VM-based utility computing. In: Proc. of IEEE Intl. Conf. on Computer and Information Technology (ICCIT), Fukushima, Japan, pp. 1053–1058 (2007)
50. Steinder, M., Whalley, I., Carrera, D., Gaweda, I., Chess, D.: Server virtualization in autonomic management of heterogeneous workloads. In: Proc. of IEEE Symp. on Integrated Network Management, pp. 139–148 (2007)
51. Stoess, J., Lang, C., Bellosa, F.: Energy management for hypervisor-based virtual machines. In: Proc. of USENIX Annual Technical Conf. (ATEC), Santa Clara, CA, pp. 1–14 (2007)
52. Tang, Q., Gupta, S.K.S., Varsamopoulos, G.: 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)
53. Verma, A., Ahuja, P., Neogi, A.: pMapper: power and migration cost aware application placement in virtualized systems. In: Proc. of ACM/IFIP/USENIX Intl. Conf. on Middleware (MIDDLEWARE), Leuven, Belgium, pp. 243–264 (2008)
54. Verma, A., Ahuja, P., Neogi, A.: Power-aware dynamic placement of HPC applications. In: Proc. of Intl. Conf. on Supercomputing (ICS), Island of Kos, Greece, pp. 175–184 (2008)
55. Voorsluys, W., Broberg, J., Venugopal, S., Buyya, R.: Cost of virtual machine live migration in clouds: a performance evaluation. In: Proc. of Intl. Conf. on Cloud Computing (CloudCom), Beijing, China, pp. 254–265 (2009)
56. Wood, T., Tarasuk-Levin, G., Shenoy, P., Desnoyers, P., Cecchet, E., Corner, M.D.: Memory buddies: exploiting page sharing for smart colocation in virtualized data centers. In: Proc. of the ACM SIGPLAN/SIGOPS Conf. on Virtual Execution Environments (VEE), Washington, DC, pp. 31–40 (2009)
57. Zhu, Q., Zhu, J., Agrawal, G.: Power-aware consolidation of scientific workflows in virtualized environments. In: Proc. of Intl. Conf. on High Performance Computing, Networking, Storage and Analysis (SC), New Orleans, LA, pp. 1–12 (2010)