Holistic Virtual Machine Scheduling in Cloud Datacenters towards Minimizing Total Energy

Xiang Li, Peter Garraghan, Xiaohong Jiang, Zhaohui Wu and Jie Xu, *Member, IEEE*

Abstract—Energy consumed by Cloud datacenters has dramatically increased, driven by rapid uptake of applications and services globally provisioned through virtualization. By applying energy-aware virtual machine scheduling, Cloud providers are able to achieve enhanced energy efficiency and reduced operation cost. Energy consumption of datacenters consists of computing energy and cooling energy. However, due to the complexity of energy and thermal modeling of realistic Cloud datacenter operation, traditional approaches are unable to provide a comprehensive in-depth solution for virtual machine scheduling which encompasses both computing and cooling energy. This paper addresses this challenge by presenting an elaborate thermal model that analyzes the temperature distribution of airflow and server CPU. We propose GRANITE – a holistic virtual machine scheduling algorithm capable of minimizing total datacenter energy consumption. The algorithm is evaluated against other existing workload scheduling algorithms MaxUtil, TASA, IQR and Random using real Cloud workload characteristics extracted from Google datacenter tracelog. Results demonstrate that GRANITE consumes 4.3% - 43.6% less total energy in comparison to the state-of-the-art, and reduces the probability of critical temperature violation by 99.2% with 0.17% SLA violation rate as the performance penalty.

Index Terms—Cloud computing; energy efficiency; datacenter modeling; workload scheduling; virtual machine

1 Introduction

The global uptake of Cloud computing has subsequently driven a dramatic increase in datacenter power consumption. Datacenters composed of thousands of interconnected servers built to provide various Cloud services globally, subsequently consuming enormous amount of energy. In the past ten years, Google servers' electricity demand has increased approximately 20 fold [1], accouting for nearly 1% of all electricity use for the world [2]. A study within 2011 [2] shows that the energy used by datacenters increased by 36% in US and 56% worldwide from 2005 - 2010, accounting for 2% and 1.3% of total electricity use, respectively. The energy consumption is expected to increase continuesly during the coming years, and it is supported by [3], predicting that a global annual datacenter construction size for 2020 will be $78 billion. In addition to high operational costs, a range of problems manifest due to high energy consumption and heat density including reduced system reliability, degraded service performance and environmental deterioration [3].

Datacenter energy usage can be categorized as stemming from computing and cooling, with the latter forming 43% of the total energy consumption as reported in [4]. On one hand, server computing energy consumption of underutilized resources accounts for a substantial amount of the actual energy use, particularly in Cloud environments [5]. For this reason, server consolidation is often used to achieve enhanced computing energy efficiency[6], [7], and functions by scheduling workloads to fewer servers and shutting down idle servers (or put them into sleep mode). On the other hand, due to the skewed temperature distribution (i.e. cooling must address the hottest server), workload balancing becomes an important consideration [8], [9] to reduce cooling energy draw. This is achieved by distributing workload evenly amongst servers to minimize the highest temperature in order to avoid hot spots. The analysis results in opposing objectives from the perspective of workload scheduling: load consolidation attempts to decrease the number of active servers to save computing energy, however consolidation onto fewer servers can form hot spots which results in higher cooling energy requirement. Load balancing attempts to avoid hot spots however more active servers operating at lower utilization can consume unnecessary computing power.

Furthermore, with the rapid development of virtualization technology and the emergence of Cloud computing paradigm, a large number of future-generation datacenters use virtualization technology allowing dynamic resource scaling and migration. As a result, it is imperative to address the scheduling of Virtual Machines (VMs) considering both computing and cooling energy. However, this is challenging as it requires in-depth interdisciplinary knowledge of computing, fluid mechanics and thermodynamics. Firstly, fine-grained models capturing the Computer Room Air Conditioner (CRAC), airflow and server have to be developed and evaluated to lay the foundation for algorithm design. Secondly, these models should be integrated into an energy-aware scheduling algorithm exploiting datacenter characteristics that minimizes the total energy consumption while adhering to performance overheads within an acceptable range dictated by the Service

- *Xiang Li, Xianghong Jiang and Zhaohui Wu are with the Department of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China.*
  E-mail: {lixiang2011, jiangxh, wzh}@zju.edu.cn
- *Peter Garraghan: is with School of Computing & Communications, Lancaster University, LA1 4WA, UK.*
  E-mail: p.garraghan@lancaster.ac.uk
- *Jie Xu is with the School of Computing, University of Leeds, Leeds LS2 9JT, UK.*
  E-mail: j.xu@leeds.ac.uk.

Level Agreement (SLA). Finally, in order to evaluate such an algorithm, it is also necessary to conduct experiments in order for evaluation against other algorithms in terms of energy efficiency, system availability and reliability. Our paper addresses these challenges and the main contributions can be summarized as follows.

- *Cooling models capturing thermal features of CRACs, air and servers within datacenters holistically*. We describe the methodology of defining the model parameters with Computational Fluid Dynamics (CFD) technique. We further implement the models in Cloud simulator to demonstrate its usage. To the best of our knowledge, this is the first paper addresses server CPU temperature considering cooling infrastructure, datacenter layout and server workload in a dynamic and holistic manner.
- *Virtual machine placement and migration algorithm to minimize total energy*. We present GRANITE - a VM scheduling algorithm for reducing holistic datacenter energy consumption. We consider both initial VM placement and dynamic live migration to achieve better energy efficiency. Our algorithm explicitly takes account of energy consumed by cooling devices and servers.
- *Algorithm performance evaluation in comparison with other algorithms.* Four representative energy-aware scheduling algorithms are selected for comparing GRANITE. Experiments demonstrate that it can achieve 43.6% less than the worst algorithm and 4.3% less than the second best algorithm in terms of total energy, and maintain the SLA violation in an acceptable range.

The remainder of this paper is organized as follows. Section 2 gives the background and the challenges of this work. Section 3 describes the problem statement and the holistic datacenter modeling. Section 4 presents the methodology of model parameter identification. Section 5 details our greedy based VM scheduling algorithm. Section 6 presents the performance results. Section 7 surveys the precious work on energy-aware management and Section 8 discusses the conclusions and further research directions.

2 Background

2.1 Datacenter Energy Characteristics

The location of hot recirculation regions within in the facility and the mixing pattern of hot rack exhaust air with the cold supply air are key issues in datacenter thermal management. There are several candidate configurations available for the air ducting designs for datacenters [10], referred as Supply and Return Schema. Without loss of generality, we consider the "raised floor & ceiling return" schema, which is demonstrated as the best option in datacenter design. In this scenario, cold air supplied by CRACs passes through the raised floor plenum to cool computing equipment within the datacenter. Assuming the datacenter comprises A CRACs, represented by

$$\mathrm{C} = \{AC_1, AC_2, \ldots, AC_{\mathrm{A}}\}, \tag{1}$$

in which AC_i is the i-th CRAC. Cold air supplied by CRACs enter each rack through the inlet and flows out from the rear, removing the heat generated by computing servers as hot air. The space between two inlet sides is known as the cold aisle, while the space between two outlet sides is called hot aisle [11], [12]. Hot air eventually is exhausted through return vents locating near the ceiling. A typical Cloud datacenter provides services using virtualization technology such as Xen, KVM or VMware. The workload from users during time interval $[t_1, t_2]$ is a sequence comprising ς VMs, and the workload is scheduled among M servers Π within the datacenter. We represent the workload ς and servers Π by:

$$\overset{t1\to t2}{\mathrm{V}} = \{VM_1, VM_2, \ldots, VM_{\mathrm{V}}\}, \tag{2}$$

$$\mathrm{P} = \{PM_1, PM_2, \ldots, PM_{\mathrm{M}}\}. \tag{3}$$

Generally, any specific VM scheduling solution Σ during time interval $[t_1, t_2]$ can be represented as Equation (4), including initial placement I and dynamic migration Γ [13].

$$\begin{aligned} &\overset{t1\to t2}{\mathrm{S}} = (\mathrm{I}, \mathrm{G}), \\ &\mathrm{I} = \{PM(VM_1), PM(VM_2), ..., PM(VM_{\mathrm{V}})\}, \\ &\mathrm{G} = \{PM(\mathrm{O}_1), PM(\mathrm{O}_2), ..., PM(\mathrm{O}_{\mathrm{W}})\}, \\ &\text{where} \\ &PM(VM_i) \in \mathrm{P}, PM(\mathrm{O}_i) \in \mathrm{P}, \\ &\forall i \le \mathrm{W} \le \mathrm{V}, \mathrm{O}_i \subseteq \overset{t1\to t2}{\mathrm{V}}. \end{aligned} \tag{4}$$

The scheduling system will initiate a newly arriving virtual machine VM_i within a server according to a specific algorithm. The initial placement stage I is represented as a map from the VMs to servers, and $PM(VM_i)$ is the placement for VM_i. In migration stage Γ, the status of all servers are checked at a regular interval. The scheduling system selects a virtual machine subset O comprising Ω VMs which are migrated from their original host server to another server, denoted by $PM(\mathrm{O}_i)$. For algorithms which do not consider VM migration [14], [15], Γ is configured as null. Total energy required to operate a datacenter is considered as the sum of computing energy $E_{computing}$ and cooling energy $E_{cooling}$ [16], [17], which are considered as energy

TABLE 1
SYMBOLS AND DEFINITIONS

Symbol	Definition
X, ς, Π, T	Sets of CRACs, VMs, servers and tasks
A, ς, M, K	Numbers of CRACs, VMs, servers and tasks
AC	Abbreviation of CRAC
t	Time [s]
Σ, I, Γ	Scheduling solution Σ includes: (1) I, initial placement stage and (2) Γ, dynamic migration stage
PM, *VM*	Abbreviation of servers and virtual machines
E	Energy consumption [kWh]
Є	Configuration / capacity of CPU, memory, etc.
u, *θ*	Task CPU utilization and length [instructions]
λ	Task submission rate
P	Power consumption [W]
T	Temperature [K]
Q_{AC}	Heat removed by CRACs [J]
w	Rotation speed of the fan [r/s]
R_k	Temperature raise of rack *k* by air recirculation [K]
C	(Specific) heat capacity [J/K, J/(kg*K)]
R	Thermal resistance [K/W]
comp/fan	CRAC compressor unit / fan unit
sup	CRAC supply air
inlet	Rack inlet for cooling air

consumption by servers and CRACs in this paper, respectively. That is,

$$E_{total}^{t1\to t2} = E_{computing}^{t1\to t2} + E_{cooling}^{t1\to t2}. \quad (5)$$

Each specific VM scheduling solution Σ corresponds to a workload distribution and server utilization profiles that determine the server energy consumption. Therefore the computing energy $E_{computing}$ is a function of Σ. On the other hand, the cooling efficiency is positively correlated with its Supply Air Temperature (SAT, T_{sup}) [8], [18]. However, T_{sup} should be set low enough to keep the server temperatures under their critical temperatures [16], [19]. Generally, higher server temperature requires lower T_{sup} to cool down. Meanwhile, server temperature is affected by its workload status which is a function of scheduling Σ. Therefore, scheduling solution Σ determines cooling energy in an indirect manner. Given the workload and datacenter thermal characteristics, we represent the energy consumption as

$$E_{total} = E_{computing}(\mathrm{S}) + E_{cooling}(\mathrm{S}). \quad (6)$$

2.2 Challenges in Total-Energy-Aware Scheduling

Equation (6) indicates a necessity to consider both computing and cooling operation together in order to reduce total datacenter energy. However, this is challenging due to complicated relationship between $E_{computing}$ and $E_{coooling}$ [20]. There traditionally exist two different perspectives for energy-aware scheduling in datacenters: computing [5], [15] and cooling [8], [21]. On one hand, minimizing computing energy typically involves consolidating workloads into fewer servers, however results in increased likelihood of high temperature hot spots [16] needing additional cooling energy for removal. On the other hand, the problems of minimizing cooling energy and the highest server temperature are shown to be equivalent [18]. Therefore, minimizing cooling energy entails workload balancing. However this results in more active servers yielding higher computing energy. More formally, assuming that Σ′ and Σ″ are scheduling solutions that produce the minimum computing energy and cooling energy, respectively.

$$\begin{aligned} E_{computing}(\mathrm{S}') &= \min\left[E_{computing}(\mathrm{S})\right], \\ E_{cooling}(\mathrm{S}'') &= \min\left[E_{cooling}(\mathrm{S})\right]. \end{aligned} \quad (7)$$

However, solutions that produce localized optimization value do not necessarily achieve the globally minimized value. In order to achieve the best energy efficiency in terms of total energy, we attempt to find the global optimization solution $\tilde{\mathrm{S}}$ that

$$\begin{aligned} &E_{computing}(\tilde{\mathrm{S}}) + E_{cooling}(\tilde{\mathrm{S}}) = \min\left[E_{total}\right], \\ &\text{however } \tilde{\mathrm{S}} \neq \mathrm{S}' \text{ and } \tilde{\mathrm{S}} \neq \mathrm{S}''. \end{aligned} \quad (8)$$

In other words, uncoordinated scheduling algorithms that consider minimizing computing and cooling energy usage as isolated optimization problems do not necessarily result in the minimal total datacenter energy [17], [20]. Therefore it is critical for energy-aware workload scheduling to consider computing and cooling energy as a single optimization problem to ascertain an optimal trade-off point and achieve the best energy efficiency.

Another challenge is developing fine-grained simulation environment with holistic models. Computing energy modeling has been studied in detail [5], [16]. Traditional CFD technique has been intensively studied [22], [23], but is not applicable for online scheduling due to its time-consuming characteristic. CloudSim [24], [25] is one of the most powerful simulation platforms for Cloud computing. However, an existing limitation of CloudSim is the inability to provide models of cooling infrastructures within datacenters. Prior to algorithm design, it is necessary to implement the cooling models presented in this paper. In order to make simulation results more convincing, we compare the model outputs with CFD modeling results. Furthermore, we try to use real measured parameters to define our models, such as the workload from real Google datacenter tracelog, real server power profiles, etc.

3 Cloud Datacenter Modeling

In this section, we introduce the models our solution is based on, followed by the problem statement targeted by our paper. Fig. 1 illustrates the overall architecture of our methodology. Users submit tasks deployed within VMs. The scheduling system is responsible for VM placement and migration. Furthermore, it dynamically adjusts CRAC capacity in order to reduce cooling costs. Decision making of the scheduling system is based on the models identified off-line, including workload models, server models and cooling models. The variables and parameters used in the modeling are presented within Table 1.

3.1 Workload Models

Assume that during any time interval $[t_1, t_2]$, the workload comprises ς VMs, presented in Equation (2). More specifically, the configuration of each VM is represented by

$$\epsilon_{VM\,i} = \left\{\epsilon_{core}, \epsilon_{mips}, \epsilon_{memory}\right\}, i \in [1, 2, ..., \mathrm{V}], \quad (9)$$

in which $\epsilon_{VM\,i}$ is the configuration of i-th VM. Each VM includes the number of VCPU (ϵ_{core}), core processing speed (ϵ_{mips}) and memory size (ϵ_{memory}). In each VM, we deploy the tasks submitted by users. In other words, we consider the scenario that tasks are deployed in VMs rather than directly in servers. Each server hosts one or more VMs at any given time. The corresponding VM is instantiated when deploying a task and destroyed after finishing the task. The task sequence submitted by users is represented by

$$\begin{aligned} \mathrm{T}^{t1\to t2} &= \left\{task_1, task_2, \ldots, task_{\mathrm{K}}\right\}, \\ task_i &= \left\{u, \theta\right\}, i \in [1, 2, ..., \mathrm{K}], \end{aligned} \quad (10)$$

where K is the number of all tasks, and each task is represented by CPU utilization u and length θ (instructions). Assuming that each VM runs a single task, we have ς = K. The tasks are submitted according to a submission rate λ, which is a function of time denoted by

$$\lambda = f_{submission}(t). \quad (11)$$

The total number of submitted tasks are the integral of submission rate if we consider the submission rate as a continuous function, we have

$$\mathrm{K} = \int_{t1}^{t2} f_{submission}(t)\, dt. \quad (12)$$

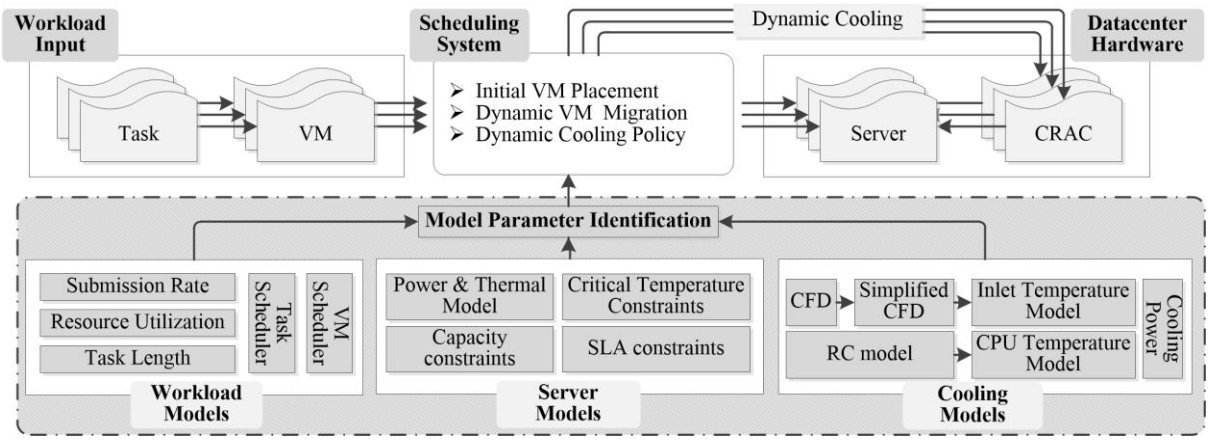


Fig. 1. Holistic Cloud datacenter models for total-energy-aware VM scheduling

3.2 Server Models

Assuming the datacenter comprises M heterogeneous servers shown in Equation (3) with different CPU capacity, memory size, etc. Given the workload during time interval $[t_1, t_2]$ and the server status at time t_1, a specific scheduling solution Σ will produce corresponding workload distribution and server status. The CPU utilization is as follows.

$$\mathrm{U}(t)=\left\{u_{PM1}(t),u_{PM2}(t),\ldots,u_{PMM}(t)\right\},\ t\in[t_1,t_2], \quad (13)$$

where $u_{PM1}(t)$ is the CPU utilization of i-th PM at time t. Server power consumption is primarily dependant on CPU utilization. In practical terms, we use the power models of HP ProLiant ML110 G4 and G5 [25], which define the power consumption at {0%, 10%, …, 100%}. Power at the utilization at any interval is modelled by a segmented linear function. Server status is defined as *active* and *inactive*. A server utilization greater than 0% indicates that a server is active. An inactive server (sleep mode, turned off) results in power utilization equals to 0, requiring ignorable energy. The computing power of all servers at time t is given by

$$P_{PM}(t)=\sum_{i=1}^{i=\mathrm{M}}P_{PM\,i}\left(u_{PM\,i}(t)\right)=\sum_{i=1}^{i=\mathrm{M}}P_{PM\,i}\left(\mathrm{U}_i(t)\right). \quad (14)$$

The energy of all servers within a datacenter during time interval $[t_1, t_2]$ is obtained by

$$E_{PM}=\int_{t_1}^{t_2}P_{PM}(t)\,dt. \quad (15)$$

Meanwhile, resources required by all VMs that run on server i cannot exceed its capacity $\mathcal{C}_{PM\,i}$, since we donnot use oversubscription technique [15], [25]. Therefore a feasible scheduling algorithm mush satisfy

$$\sum\mathcal{C}_{VMs\ \mathrm{in}\ PM\ i}\le\mathcal{C}_{PM\,i}, i\in[1,2,...,\mathrm{M}]. \quad (16)$$

Additionally, in order to ensure that the servers will not overheat and eventually fail, it is necessary to maintain server temperature under the critical threshold ($T_{critical}$) at all times [19], satisfying

$$\boldsymbol{T}_{\boldsymbol{PM}}(t)\le\boldsymbol{T}_{\boldsymbol{critical}}, \quad (17)$$

where $\boldsymbol{T}_{\boldsymbol{PM}}$ and $\boldsymbol{T}_{\boldsymbol{critical}}$ are bold, representing vectors of server temperatures and their critical temperatures. Numerous works [16], [17], [19] consider inlet air temperature $\boldsymbol{T}_{\boldsymbol{inlet}}$ as $\boldsymbol{T}_{\boldsymbol{PM}}$ while [12], [26] consider the CPU temperature $\boldsymbol{T}_{\boldsymbol{cpu}}$ as $\boldsymbol{T}_{\boldsymbol{PM}}$. We argue that inlet temperature is an insufficient performance indicator as thermal management target is indicated by CPU temperature. As a result, this paper focuses on CPU temperature as analysis objective.

An important aspect for Cloud providers is the set of Quality of Service (QoS) guarantees. This is commonly referred as a SLA [27]. In this paper, we consider the constraints of SLA as follows [25].

$$SLA_v=\left(CPU_R-CPU_A\right)/CPU_R\le\hat{S}, \quad (18)$$

in which SLA_v is a metric to evaluate the level of violation in SLA, and $\hat{S}$ is the maximum acceptable threshold. CPU_R and CPU_A are the CPU capacity required by users and actual allocation by the Cloud system, respectively.

3.3 Cooling Models

We consider the CRACs as the only cooling devices since it accounts for the most cooling energy [4]. The energy consumed by each CRAC during time interval $[t_1, t_2]$ comprises compressor (i.e. a component of the CARC, responsible for air compression) energy and fan energy [10], [11].

$$E_{AC}^{t1\to t2}=E_{comp}^{t1\to t2}+E_{fan}^{t1\to t2}. \quad (19)$$

The power consumption of a fan unit is proportional to the cubic of its rotation speed w, and the energy consumption is obtained by time integral of the power. The energy consumed by the compressor is given by

$$E_{comp}^{t1\to t2}=Q_{AC}^{t1\to t2}/CoP\left(T_{sup}\right), \quad (20)$$

where $Q_{AC}^{t1\to t2}$ is the heat removed by the CRAC, which can be modelled according to server energy [19] or the heat disparity between datacenter out flow and CRAC supply air [11]. CoP is the Coefficient of Performance [8], [11]. A higher CoP indicates a more efficient process, requiring less work to remove a constant amount of heat. Research shows a positive correlation between CoP and supply air temperature (T_{sup}) [8]:

$$CoP=0.0068\times T_{sup}^2+0.008\times T_{sup}+0.458. \quad (21)$$

Equation (21) implies that we can improve cooling efficiency by maximizing T_{sup} to reduce cooling cost. However, T_{sup} must be set low enough to satisfy the CPU temperature

constraints shown in Equation (17). In order to understand the relationship between CPU temperature and T_{sup}, we propose a two-step temperature model. The first step is ***rack inlet temperature modeling*** considering both CRAC status and datacenter layout. The second step is ***CPU temperature modeling*** that factors server thermal characteristics and its respective workload. The airflow of the inlet air is a mix of recirculated hot air exhausted from other servers and supplied cool air from CRACs. The time-discrete form of rack inlet temperature is as follows [9], [28]:

$$T_{inlet,k}(t+\Delta t) = T_{inlet,k}(t) + \sum_{j=1}^{A} g_{j,k} \times w_j(t) \times \left(T_{sup,j}(t) - T_{inlet,k}(t)\right) + r_k(t), \quad (22)$$

where $T_{inlet,k}(t+\Delta t)$ and $T_{inlet,k}(t)$ are the inlet temperature of rack k at time step t+1 and t, respectively. $g_{j,k}$ quantifies the influences of the cooling settings of CRAC j to rack k, including supply air temperature $T_{sup,j}$ and the fan speed w_j. In our paper, we consider fan speed as a constant. We focus on the optimization of the supply air temperature to conduct thermal management. Here $r_k(t)$ is a time-varying item, representing the temperature effect of recirculated hot air. However, previous work lacks detailed analysis to identify $r_k(t)$. Here we introduce our approach to model the rack inlet temperature and identify $r_k(t)$ based on Equation (22) as follows.

Within a scenario of stable airflow pattern in a datacenter, according to [16], [19] the inlet temperature of rack k ($T_{inlet,k}^{stable}$) is the weighted sum of supply air temperature (T_{sup}) of each CRAC and the recirculation influence. Theoretically, $T_{inlet,k}^{stable}$ is influenced by all the working CRAC units within the datacenter. However as observed from [9], rack inlet temperature is predominately affected by a selected small number of CRAC units. In this paper, we only consider the closest CRAC (numbered as 0). The supply air temperature and the fan speed are denoted as $T_{sup,0}$ and w_0, respectively. The temperature raise of rack k imposed by recirculated influence derives from exhausted air of all the other racks, denoted by R_k. We have

$$T_{inlet,k}^{stable} = T_{sup,0} + R_k. \quad (23)$$

Assume that $g_{0,k}$ in Equation (22) is a time-ste p-proportional parameter shown as follows, defining the temperature impact from CRAC status.

$$g_{0,k} \times w_0(t) = g' \times w_0 \times \Delta t = G' \times \Delta t, \quad (24)$$

where $g'$and G' are constants. A differential equation to capture the dynamics of inlet temperature based on Equation (22) and its solution is shown below,

$$T_{inlet,k}(t+dt) - T_{inlet,k}(t) = + G' \times dt \times \left(T_{inlet,k}^{stable} - T_{inlet,k}(t)\right). \quad (25)$$

$$\text{Solution: } T_{inlet,k}(t) = T_{inlet,k}^{stable} + \left(T_{inlet,k}(0) - T_{inlet,k}^{stable}\right)e^{-ct}, \quad (26)$$

where c is a constant, capturing the temperature influence by the closest CRAC unit, and $T_{inlet,k}^{stable}$ is obtained with Equation (23). $T_{inlet,k}(0)$ is the inlet temperature of rack k at the very beginning. It is now possible to present the complete inlet temperature models. The unknown parameters to be identified are R_k and c, which will be described in Section 4.2. Next, we can conduct CPU temperature modeling, which is the most important indicator of thermal management. The RC model shown below is the most established means to obtain CPU temperature [26], [29].

$$RC \times \frac{dT_{cpu}(t)}{dt} + T_{cpu}(t) - PR = T_{inlet}, \quad (27)$$

where R and C are thermal resistance and heat capacity of the server, respectively. T_{inlet} is obtained with Equation (26). By solving this differential equation, CPU temperature is modelled as follows.

$$T_{cpu}(t) = PR + T_{inlet} + \left(T_0 - PR - T_{inlet}\right) \times e^{-\frac{t}{RC}}, \quad (28)$$

where and $T_{cpu}(0)$ is the initial CPU temperature. Equation (28) implies that the stable CPU temperature is $PR + T_{inlet}$. The constraint of RC model is that it assumes the CPU power and inlet temperature is constant. However, CPU power depends on its utilization which is time-varying, and rack inlet temperature is dynamic as well described in Equation (26). Our solution is to compute CPU temperature at a regular interval with Equation (26) and (28).

3.4 Problem Statement

We formulate the problem as optimize workload scheduling to minimize the total energy consumption, which is presented in Equation (29). Given the status of servers, CRACs and workloads (represented by VMs and deployed tasks), our main target is to find the optimal scheduling solution and supply air temperature to minimize the total energy under the constraints of server capacity, SLA and CPU critical temperature.

$$\begin{aligned} &\text{Given : } \mathrm{P}_{t1}, \mathrm{C}_{t1}, \overset{t1\to t2}{\mathrm{V}}, \overset{t1\to t2}{\mathrm{T}}, \\ &\text{find : } \overset{t1\to t2}{\mathrm{S}} = \{\mathrm{I}, \mathrm{G}\} \text{ and } T_{sup}(t), \\ &\text{minimizing } \overset{t1\to t2}{E_{total}} : \overset{t1\to t2}{E_{computing}} + \overset{t1\to t2}{E_{cooling}}, \\ &\text{subjected to : } \sum \epsilon_{VMs \text{ in } PM\ i} \le \epsilon_{PM\ i}, i \in [1,2,...,\mathrm{M}], \\ &\qquad SLA_v \le \hat{S}, \\ &\qquad \boldsymbol{T_{PM}}(t) \le \boldsymbol{T_{critical}}. \end{aligned} \quad (29)$$

4 Model Parameter Identification

Prior to designing a holistic energy-aware scheduling algorithm, we implement the models proposed in Section 3 in CloudSim. Specifically, the newly added modules provide: (1) task models that capture the features of tasks submitted by users in Cloud scenarios, (2) inlet temperature model that describe the relationship between CRAC cooling capacity and rack inlet temperature, (3) RC model that gives the CPU temperature, and (4) CRAC power model used to obtain the CRAC power consumption. Above models follow the formulation presented in Section 3. However, how to define the corresponding parameters should be analyzed. Since the parameter selection of module (3) and (4) have been studied extensively [8], [11], [16], [26]. This section focus on the parameter identification for tasks and rack inlet temperature.

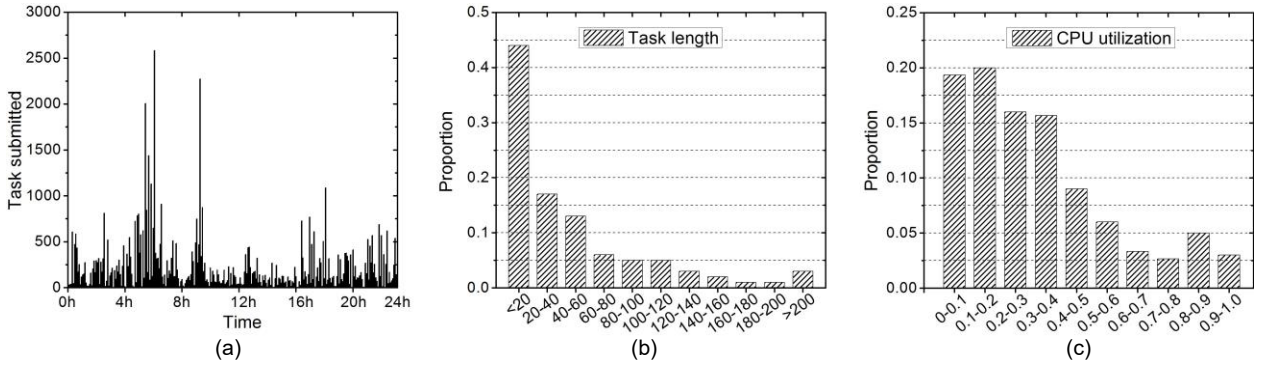


Figure 2. Task characteristics: (a) task submission rate, (b) task length distribution (billion instructions), (c) CPU utilization distribution

4.1 Task Parameters

In order to produce realistic models, it is critical to derive parameters from real-world tracelogs. We use the workload generated by our previous work in [30]. We presented a comprehensive analysis of the workload characteristics derived from Google datacenter that features approximately 25 million tasks. We model the submission characteristic of tasks through profiling submission rate hourly. Within each hour, we assume the submission rate follows a random distribution. All users are classified into six clusters as shown in [30]. We use the proportion of each cluster as the weight, and the overall submission rate follows the distribution of the weighted sum of these clusters. Then we model tasks with CPU utilization and length via fitted functions provided by our previous work [30]. Fig. 2(a) shows the modelled submission rate with 64,000 tasks. Fig. 2(b) and Fig. 2(c) show the distribution of task length and CPU utilization, respectively. According to our application scenarios, we further make the following assumptions.

Tasks are deployed in VMs rather than directly in servers. The deployment model of Cloud datacenters with IaaS (Infrastructure as a Service) typically provides Cloud services in terms of virtual machines. In this paper, users are required to initialize VMs and then deploy their tasks within. Resource utilization of each task keeps fluctuating even with fine-grained time interval. For instance, [25] assumes that task utilization varies every fixed interval (e.g. 5 minutes). Since the majority of tasks only utilize tiny proportion of resources [30], analysing utilization pattern in the level of task is reasonable. Therefore, in this paper, task utilization is assumed to be stable during its execution.

Availability constraint or communication constraint between VMs are not considered. In order to meet the requirement of users, VM scheduling is subjected to constraints such as availability constraint and communication constraint. The former is expressed as a combination of anti-collocation/collocation of VMs, implying that corresponding VMs must be placed on the same/different level (e.g. rack). Communication constraint is defined by the bandwidth and latency requirement between two VMs. In our paper, we only focus on VM scheduling under the constraints of server resource showed in Equation (16), which is termed demand constraints [13].

4.2 Rack Inlet Temperature

The advantage of time-discrete model in Equation (22) is that it simplifies the CFD modeling and captures the relationship between the CRAC cooling capacity and rack inlet temperature [9]. However, [9] does not present sufficient details on parameter identification. In order to identify the parameters R_k and c in Equation (23) and (26) for each rack, we model a datacenter using CFD technique.

The size of our modelled datacenter is $15.8 \times 6.5\text{m}^2$, comprising 4 CRACs, 2 vents and 24 rack. The model is 2-dimensional built in Gridgen, including 7,556 cells with the supply air temperature being 290K. The outlet temperature and the initial inlet temperature are configured at 310K. Fig. 3 presents the temperature contour within the datacenter at time 10s, 200s and 600s. We observe that racks near to CRAC units result in lower inlet temperature compared to racks far from CRAC units, showing that datacenter layout imposes great impact on the cooling efficiency of each rack. We also find that there is a significant difference between Fig. 3(a) and Fig. 3(b) in terms of temperature distribution. While the difference between Fig. 3(b) and Fig. 3(c) is not obvious even with wider time gap in comparison with that of Fig. 3(a) and Fig. 3(b), representing that the datacenter becomes stable after certain period of time. From our massive observations, temperature will be stabilized in 500s in majority cases. As a result we identify $T_{inlet,k}^{stable}$ in Equation (23) as the average temperature after 500s.

First, we identify the value of R_k in various of scenarios. R_k is related to the power consumption of servers since recirculation influence is produced by hot air from each rack. Without loss of generality, we assume that the outlet temperatures are identical amongst each rack and ranges in E = [305K, 315K], representing different workload intensities. For each of the workload intensities within range E, we monitor air temperature before entering each rack (inlet temperature) and compute corresponding $T_{inlet,k}^{stable}$, to determine R_k in Equation (23). R_k is then applied to Equation (26). Parameter c is repeatedly selected within a specific range with step of 0.0001 to produce a profile with Equation (26). Root Mean Square (RMS) errors are computed between the produced profile and the monitored temperature in CFD. This results in selecting c with the minimum RMS.

Using this method, we analyze R_k and c of Point 1-6 (illustrated in Fig. 3) under various of outlet temperatures within range E. These points are sufficient to capture the inlet temperatures of all racks since the datacenter is symmetrical. Fig. 4(a) presents the profile of R_k of each point under different outlet temperatures. We fit each profile

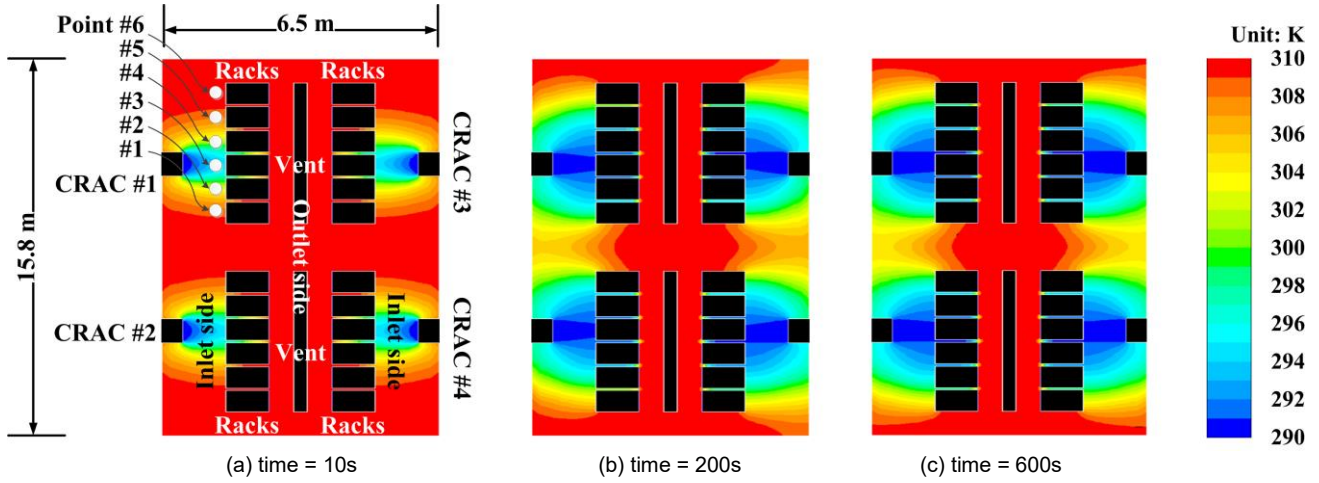


Figure 3. Temperature contour of datacenter

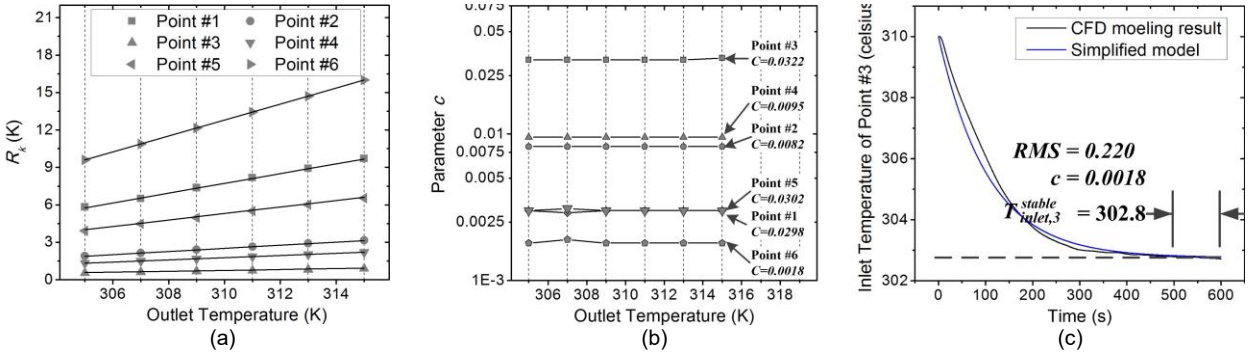


Figure 4. Profiles of (a) R_k, (b) parameter c, (c) a case study of inlet temperature modeling

with a linear function via regression. The R-square is greater than 99.6% and the standard error is within 0.02 in the regression, showing high linearity, which is consistent with the description in [31]. Since we represent different workload intensities in the datacenter via outlet temperature, in practical terms it is necessary to map the workload intensities proportionally into E to determine the corresponding R_k. Fig. 4(b) shows the identified parameter c for each level of workload intensities. As illustrated, it is stable under different scenarios. This is due to parameter c is dependant on datacenter layout and thermal characteristics.

From Fig. 4(a) and Fig. 4(b), we can observe that the points near a CRAC exhibit lower recirculation influence and higher parameter c, indicating higher cooling efficiency. Fig. 4(c) shows a case study to demonstrate the practicality of our model. In this case, we set the outlet temperature and the initial inlet temperature to 310K, and monitor Point 6 to analyze the model accuracy. First we identify R_k with the regression model shown in Fig. 4(a) and get 12.8K. With Equation (23), stable temperature is determined as 290K+12.7K=302.8K. Parameter c is 0.0018 identified from Fig. 4(b). In comparison with the CFD modeling results, the RMS is 0.220. Our experiments show that the RMS erro rs are under 0.24 in the majority of cases, demonstrating high modeling accuracy.

5 VM Scheduling Algorithm

The holistic modeling methodology has been detailed in Section 3, followed by the description of the approach to identify model parameters. Based on the presented models, we propose our solution to the problem described in Section 3.4 by introducing GRANITE, a GReedy *based scheduling Algorithm miNImizing Total Energy*. Accordingly, GRANITE contains two stages: (1) initial VM placement I and (2) dynamic live migration Γ. Meanwhile, the CRAC capacity is dynamically updated to achieve better cooling efficiency: the cooling capacity is adjusted to the lowest capacity (least cooling energy) while maintaining the CPU temperature lower than critical temperature. Our algorithm is based on the assumption that for Cloud providers the information of the requests submitted by users, such as the CPU capacity, memory size of each VM and the task utilization is given, or can be predicted. This assumption is widely adopted within the research area [5], [12], [19].

5.1 Initial Placement

In the initial placement stage, GRANITE selects the server with a greedy algorithm. We select the server resulting in the least increase in terms of total power consumption after VM placement. The increase is obtained as follows. First, the total power consumption before placement is given by

$$P_{total} = \sum_{i=1}^{i=M} P_{PM\ i} + \sum_{i=1}^{i=A} P_{AC\ i}. \quad (30)$$

For any submitted VM ςi, it is necessary to select a server for its placement. Assume that ς_i is allocated to PM_1, and the CPU utilization of PM_1 after allocation is $(u_{PM\ 1})'$. Accordingly, the power cost of PM_1 calculated with a segmented linear power model. The CPU temperature is then predicted after allocation during the next time interval (e.g.

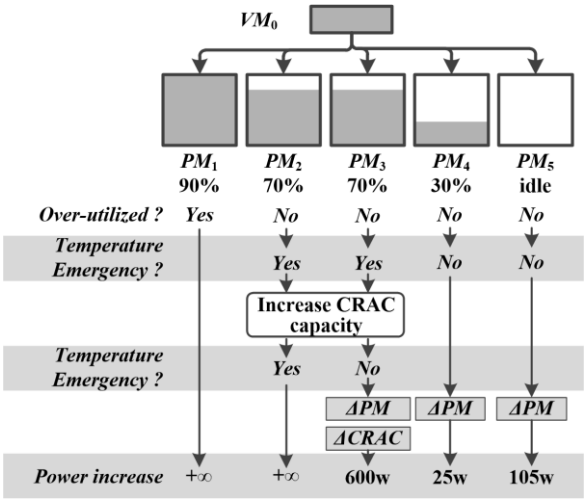


Figure 5. Initial VM placement in GRANITE

```
Function dynamicMigration ()
    For: every migration interval
        run ()
End Function

Function run ()
    Rank all servers according to predicted temperature
            with Equation (28) in descending order.
    threshold proportion ← 10%
    δ ← minimum temperature of top 10 percentile of servers
    For: each server PM in datacenter
        If: PM.T_cpu > δ
            workloadBalance ()
End Function

Function workloadBalance ()
    While: (PM.T_cpu > δ)
        V ← VM with the minimum utilization within PM
        PM_target ← select a server with the minimum S
        migrate V to PM_target
End Function
```

Figure 6. Pseudo-code of dynamic migration

5 minutes), denoted by $(T_{PM\,1})'$, with temperature models given in Section 3.3. If $(T_{PM\,1})'$ is less than its critical temperature, CRACs will not be adjusted. Otherwise, the cooling capacity of the nearest CRAC is gradually increased by decreasing the SAT, until the predicted $(T_{PM\,1})'$ is less than the critical temperature. Applying Equation (30) again with updated servers and CRACs, we obtain total power after placing ς_i to PM_1, denoted by $(P_{total})'$. Similarly, if we place ς_i to PM_j, $j \in [2, 3, \ldots, M]$, corresponding total power after placement is obtained. The power increase S is

$$S = (P_{total})' - P_{total}, \tag{31}$$

in which P_{total} and $(P_{total})'$ are the total power cost before and after the VM placement, respectively. Our algorithm selects the server with the minimum S as the placement target. Note that our algorithm will not select servers that result in fully utilized resource after placement, since resource contention potentially imposes great impact on QoS.

To better illustrate the greedy-based selection of initial placement, we present a case study shown in Fig. 5. VM_0 is submitted to Cloud datacenter, and will be scheduled among 5 servers numbered as 1-5. Allocating VM_0 to PM_1 will results in overutilized CPU and potential of SLA violation since PM_1 is close to full prior to allocation, the power increase S is regarded as infinite. For PM_2 and PM_3, the newly coming VM_0 will result in CPU temperature exceeding the critical threshold. When increasing the cooling capacity, CPU temperature of PM_3 can be maintained under critical threshold, while PM_2 will be definitely greater than threshold even with the highest capacity. Therefore, $\mathfrak{S}$ for PM_2 is infinite, and $\mathfrak{S}$ for PM_3 is the sum of the power increase of server (ΔPM) and the power increase of the CRAC ($\Delta CRAC$). For PM_4 and PM_5, $\mathfrak{S}$ is the power increase of the allocated server (ΔPM) only, since the allocation does not incur critical temperature violation. However, power increase of PM_5 (ΔPM) is significantly greater than PM_4 due to inactive status before allocation, which consumes negligible energy. Eventually, PM_4 will be the allocation target with a power increase of 25W.

5.2 Dynamic Migration

In this stage, live migrations are conducted to balance the workloads and reduce cooling capacity with its pseudo-code shown in Fig. 6. Dynamic migration is performed at a regular interval. Within each interval, GRANITE checks the status of all servers and defines a temperature threshold $\boldsymbol{\delta}$. If the server temperature is greater than this threshold, one or more VMs will be migrated out. The threshold can be configured to be static or dynamic [25]. The static threshold lacks flexibility in a real Cloud datacenter, which is featured in dynamics and scalability. For this reason, a dynamic threshold is adopted which changes according to specific circumstances. We define the temperature ranking at a certain proportion as the threshold. For instance, we can define the proportion as 10%. Any server with CPU temperature ranks within top 10% is regarded as a hotspot and workload balancing is applied. The VM V with the

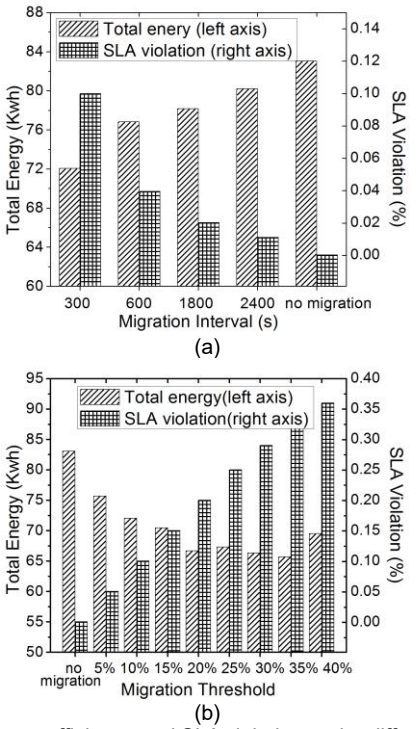


Figure 7. Energy efficiency and SLA violation under different (a) migration intervals, and (b) migration thresholds proportion

TABLE 2
EXPERIMENT PARAMETERS

Item		Value
Datacenter	Number of racks	24
	Number of servers each rack	15 (360 servers in total)
	Number of CRACs	4
	Parameters of inlet temperature model	According to Section 4.2
	Cooling update interval	300 seconds
Server	Heat capacity	340 [J/K]
	Thermal resistance	0.34 [K/W]
	Number of CPU cores	2-8
	MIPS of each core	1,860, 2,660
	CPU Power model	Xeon3040, Xeon3075
	CPU critical temperature	70°C
	VM scheduler (VMM)	Time shared
Workload	Parameters of tasks	According to Section 4.1
	Number of tasks (each VM runs one task)	32,000-64,000
	VCPU cores of each VM	1, 2 or 4
	MIPS of each VCPU	500, 1000, 2,000, 2,500
	Task scheduler	Time shared
Simulator	Simulation time	24 hours
	Simulator	Based on CloudSim V.3.0.3

minimum CPU utilization within the server *PM* will be selected to conduct live migration. The migration target PM_{target} is selected with greedy algorithm similar to the initial placement stage. Note that we do not use the widely adopted workload consolidation [6], [7] in GRANITE as the greedy-based algorithm always selects the server with the best energy-efficiency. In other words, energy-efficient servers will be allocated with more workloads which results in the similar effect of consolidation.

In order to evaluate the performance of dynamic migration. We conducted simulation experiments to demonstrate the impact of migration interval and threshold using the parameters in Table 2. Fig. 7(a) shows that as the migration frequency decreases, we observe an increasing trend in total energy consumption and decreasing trend in SLA violation rate (SLA_v, defined in Equation (18)). Since oversubscription is not allowed in our scenario, SLA violation incurs when operating VM live migration, which results 10% performance degradation [32]. From the figure we learn that a migration interval should be selected as a trade-off of energy efficiency and performance overhead. Fig. 7(b) shows the impact of different thresholds proportion within 0%-40%. Higher threshold proportion requires more VM migration leading to higher SLA violation rate. This results in more balance temperature distribution, requiring more computing energy and less cooling energy.

6 EXPERIMENT RESULTS

6.1 Baseline Algorithms

To better illustrate the effectiveness of our algorithm GRANITE, we evaluated it against three representative scheduling algorithms: (1) MaxUtil [5] that attempts to minimize only computing energy by allocating workloads to the server with the maximum average utilization, (2) TASA [12] that tries to minimize only cooling energy by allocating workloads to the current coolest server, (3) IQR [25] considersing both computing and cooling energy, and (4) Random algorithm. We implemented the above algorithms within CloudSim V.3.0.3.

(1) *MaxUtil* (*Maximize Utilization*), it aims to consolidate workloads through intensifying the utilization of a small number of resources to reduce computing energy draw. Specifically, it defines a cost function as follows.

$$f_{i,j} = \sum_{t=1}^{\tau_0} u_i(t) \Big/ \tau_0 , \tag{32}$$

where τ_0 is the processing time of task *j*. The function value $f_{i,j}$ of task *j* on server *i* captures the average utilization during the task execution. For a given task, MaxUtil checks every servers from the first rack to the last, and selects the server with highest function value as scheduling target.

(2) *TASA* (*Thermal Aware Scheduling Algorithm*), it schedules workload uniformly to minimize the maximum temperature, which is a common approach to reduce the cooling capacity requirement and achieve cooling efficiency. The main idea of TASA is to schedule the "hottest" task prior to "coolest" task to the "coolest" server. In our experiment, the real-time schedule system has to allocate the submitted task immediately to meet the QoS. Therefore, we simplify TASA by allocating each task to the server with lowest CPU temperature without considering the task-temperature profile such as "cool task" and "hot task".

(3) *IQR* (*InterQuartile Range based scheduling*), it also consists of initial placement phase and dynamic migration phase. For initial placement, it allocates each VM to the server that produces the least computing power increment. In the dynamic migration phase, it conducts live migration at a regular interval to balance workloads. To identify the servers to be balanced, it checks if the server utilization is greater than the upper threshold T_u:

$$T_u = 1 - s \times IQR, \tag{33}$$

where IQR is the midspread or middle fifty, representeding a measure of statistical dispersion, being equal to the difference between the third and first quartiles, shown as follows. The parameter $s \in P+$ defines how aggressively the system balance workloads. A greater value for *s* results in more servers becoming balanced.

(4) *Random*, randomly places VMs to the servers which can accommodate them (i.e. available capacity). Apart from initial placement, we further conduct VM migration to improve its energy efficiency described in Section 5.2.

6.2 Experiment Result

The parameters for constructing the experiment are shown in Table 2. Simulation parameters are configured identically for each algorithm to make the comparison, including datacenter hardware, workloads and simulation environment. Additionally, we set proportional threshold as 20%, and migration interval as 300 seconds in GRANITE.

First we compare the energy consumption of each algorithm. We conduct experiments under different workload intensities and the results are presented in Fig 8. We observe that GRANITE achieves the best energy efficiency in terms of total energy: 43.6% less than the worst (Random algorithm), 4.3% less than the second best (IQR), and 21.0% less than the average. To ascertain insight into the performance of each algorithm, we further analyze the details in

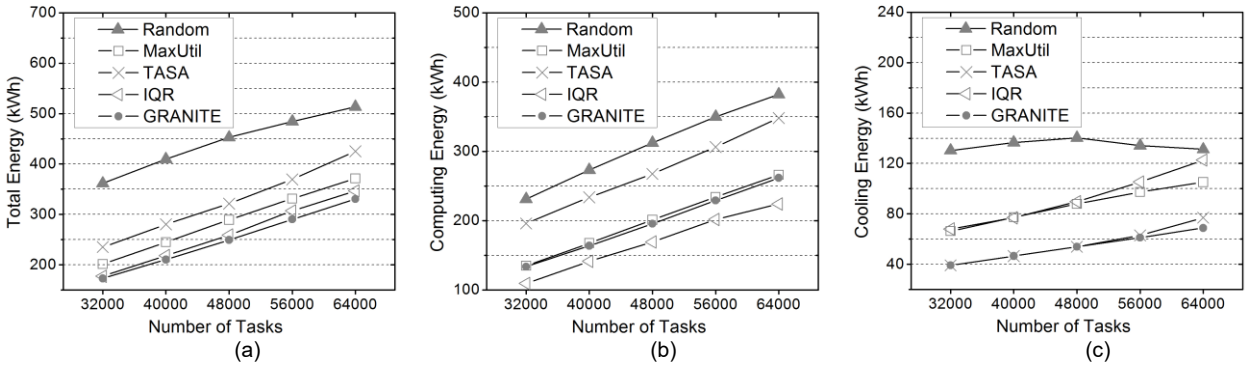


Figure 8. Energy consumption of each algorithms: (a) total energy, (b) computing energy, (c) cooling energy.

TABLE 3
AVAILABILITY OVERHEAD

Algorithm	Task Submitted	Task Finished	Average Execution Time	SLA Violation Rate (%)
Random	64,000	62,451	219.02 s	0.340
MaxUtil		62,825	218.59 s	0
TASA		62,825	218.59 s	0
IQR		62,792	218.82 s	0.013
GRANITE		62,700	218.94 s	0.170

TABLE 4
HIGHEST CPU TEMPERATURE (COLLECTED EVERY 300S)

Algorithm	Average Highest Temperature (°C)	Critical Temperature Violation (times)
Random	71.54	1390
MaxUtil	69.32	212
TASA	65.68	120
IQR	69.42	335
GRANITE	63.50	17

computing energy, cooling energy, system availability and reliabilities [33], respectively.

As illustrated in Fig 8(b), Random algorithm uses the most computing energy (48.1% more than others in average), followed by TASA. Random algorithm does not take computing energy into consideration, which means an idle or inactive (e.g. sleep mode) server has the same chance to be the allocation target. This is contradictory to the basic idea of workload consolidation and leads to computing energy waste. For TASA, due to its tendency of scheduling workloads to the server with the lowest CPU temperature, which may potentially be in sleep mod e or has been switched off. In other words, TASA balances tasks leading to much more active servers among the datacenter.

Fig. 9(a) shows the number of active servers during the experiment with 64,000 tasks. As illustrated, Random and TASA uses more active servers than the other algorithms. On the other hand, MaxUtil, IQR and GRANITE use less active servers, since (1) MaxUtil allocates workloads to the server with the maximum average utilization; (2) for IQR, it uses computing energy aware best-fit policy to initially pla ce virtual machines, allowing to intensify workloads in a smaller number of server; (3) for GRANITE, it selects a server with the minimum total power increase, and prefers to allocate the newly arrived workload to a non-idle server than activating a new server (described in Section 5.1). These three algorithms all consolidate workloads and lead to a satisfactory amount of computing energy.

Fig 8(c) shows that Random, IQR and MaxUtil consume the most cooling energy, while GRANITE and TASA consume much less. To better understand the behind reason of the cooling efficiency, we demonstrate the average workload distribution among the datacenter with 64,000 tasks by Fig 10. There are 24 racks in total, placed in 2 columns that are described in the CFD modeling part (Section 4.2). The utilization is collected every 300 seconds during the simulation and the averaged value is presented.

We can observe that both TASA (Fig 10(c)) and GRANITE (Fig 10(e)) allocate more tasks to racks nearer to the CRAC, which are more cooling efficient. This is because GRANITE - regardless whether it is within the initial placement stage or migration stage - attempts to select server without triggering temperature emergency to avoid cooling power increase. In other words, servers near a CRAC will more likely be the allocation candidate since they are comparatively cooler. Similar to GRANITE, TASA selects the coolest servers to allocate workloads and results in the similar workload distribution shown in Fig 10(c). For Random algorithm (Fig. 10(a)), utilization of the racks near CRACs is slightly higher than others. The algorithm randomly places VMs in the initial placement phase, and dynamically migrates VM from hotter servers to cooler ones described in Section 5.2. Meanwhile, MaxUtil (Fig 10.(b)) and IQR (Fig 10.(d)) schedule workloads based on CPU utilization and are not temperature-aware. The former consolidates most workloads in the first group of racks, which are far from the CRACs, while the latter consolidates workloads to more racks which are not cooling efficient. The SAT using Random, IQR and MaxUtil will be lower compared to TASA and GRANITE, indicating higher CRAC usage cost.

Fig. 9(b) shows the results that are consistent with our analysis. In our experiment, each CRAC is dynamically adjusted every 300 seconds according to datacenter temperature. Results shown in Fig. 9(b) are the average SAT of four CRACs. We can learn that temperature-aware scheduling algorithms GRANITE and TASA can significantly increase the SAT and dramatically reduce cooling capacity.

Cloud providers must guarantee that the agreed services can be satisfied in terms of availability. Since we use dynamic migration, inevitably introducing potential SLA violation (10% during the migrations [32]). As shown in Fig.

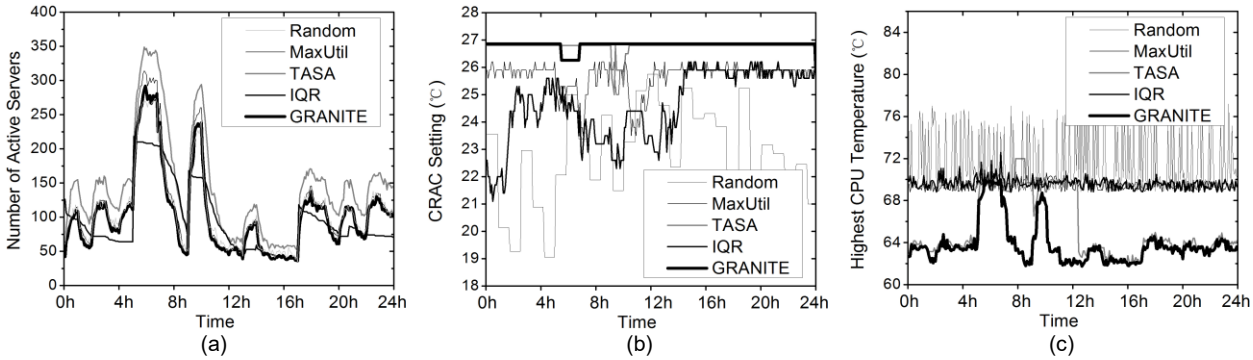


Figure 9. The profiles of (a) number of active servers, (b) supply air temperature, (c) highest CPU temperature within datacenter.

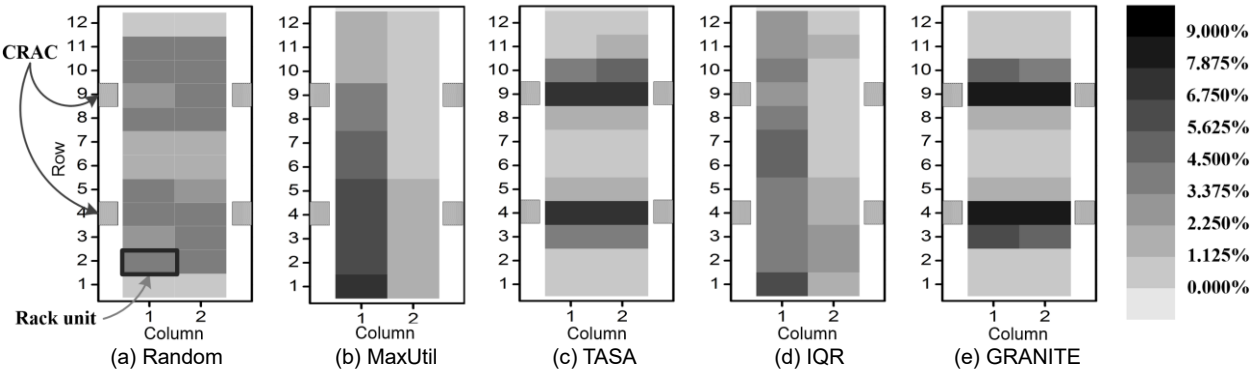


Figure 10. Average workload distribution during 24 hours (collected every 300s)

7, SLA violation rate is affected by the migration interval and migration threshold, which are defined as 300 seconds and 20%, respectively, as described above. The comparison of SLA violation rate between each algorithm is presented in Table 3. In our experiments, MaxUtil and TASA do not leverage live migration, so we assume that they can perfectly meet the SLA. As illustrated, GRANITE suffers from higher SLA violation rate (0.17%), and requires more time (218.94 s) to finish tasks in average. However, the overhead in system availability remains satisfactory and acceptable.

Highest CPU temperature within the datacenter is another important indicator of thermal management [12], [18] because (1) CRAC cooling efficiency are highly dependent on the temperature of the hottest server [34], [35], (2) system reliability [12], [33] will be significantly reduced if CPU violates its critical temperature frequently [26]. Fig. 9(c) shows the profiles of *highest CPU temperature* within the datacenter. Furthermore, Table 4 concludes the average highest temperature and the frequency of CPU critical temperature (70°C) violation. Since we observe that our algorithm rarely violates the critical temperature, we extend the experiment timespan to 10 days in order to get results with statistical significance. From the above results, we can observe that temperature-aware algorithms GRANITE and TASA achieve better result in most cases than non-temperature-aware algorithms MaxUtil, IQR and Random. Specifically, GRANITE reduce the probability of critical temperature violation by 99.2% in average, indicating a lower cooling requirement and higher hardware reliability.

7 Related Work

This section focuses on energy-aware management policies designed for heterogeneous distributed systems such as clusters and da tacenters. These policies comprises two categories: (1) software-level methods, such as task scheduling and VM scheduling; (2) hybrid methods, referred as mechanical design based methods [11], which further consider datacenter layout design, dynamic cooling, etc. Our method belongs to the latter category. This section introduces the representative work in each field, and a summarization is presented in Table 5 in terms of analysis scope, management methodology, evaluation, etc .

7.1 Software-Level Methods

Workload consolidation [5] is a common method to reduce computing energy and its main objectives is minimizing the number of active servers, assigning higher scheduling priority to servers with greater energy-efficiency, and avoidance of resource fragmentation [36]. Workload consolidation is commonly modelled as a bin packing problem which has been proved to be NP-complete [15]. Work such as [7], [37] attempt to discover the optimal solution with linear program. However their method is confined only to comparatively small-scale datacenters. To address the NP-complete problem of scheduling, researchers tend to achieve the algorithm efficiency at cost of less solution accuracy. An intuitive algorithm called First-Fit Decreasing (FFD) is introduced [38] to address this problem, and it is further improved by [6] and [7]. Similarly, Lee and Zomaya [5] proposed two algorithms: ECTC and MaxUtil to consolidate tasks, with the former algorithm trying to maximize time period of tasks running in parallel with other tasks, and the latter trying to maximize average CPU utilization during execution. These above works mainly focus on problem with 1-dimensional constraint (CPU availability). To characterize the multi-dimensional resource

TABLE 5
SUMMARIZATION OF REPRESENTATIVE MULTI-NODE ENERGY-AWARE MANAGEMENT POLICIES
Computing energy: ✶, cooling energy: ❄, SLA: ◴ (response time, capacity requirement, etc.)

Author (first) *Policy Name*	Analysis Scope	Scheduling Components	Management Methodology	Workload Model/ Source	Performance Indicator	Evaluation
Pakbaznia [19] *TA-DRP*	✶❄	Task	• Dynamic server retirement/employment • Dynamic cooling	Unknown	Total energy	TOMLAB/ MATLAB
Lee [5] *MaxUtil, ECTC*	✶	Task	• Greedy based task consolidation • Task migration	• Uniform distribution • Gaussian distribution	Computing energy	Simulation
Parolini [17]	✶❄◴	Task	Task scheduling and dynamic migration	Constant arrival rates	• Total energy • PUE	TOMSYM/ MATLAB
Deng [42]	✶◴	Task	Solve Knapsack problem via dynamic programming	World Cup Web site requests	Throughput	Real measurement
Wang [12] *TASA, TASA-B*	❄◴	Task	Temperature-aware task scheduling	Datacenter of State University of New York	• CPU temperature • Job performance • Carbon emission	Simulation
Tang [21] *Xint*	❄	Task	Task scheduling minimizing heat recirculation	Unknown	• Inlet temperature • Cooling energy	Flovent
Banerjee [19] *HTS*	✶❄◴	Task	Task scheduling and dynamic cooling	ASU datacenter	Total energy	Flovent
Khosravi [14] *ECE*	✶❄◴	VM	VM placement across multiple datacenter based on Best-Fit algorithm	Hyper-Gamma distribution	Carbon emission	CloudSim
Li [15] *EAGLE*	✶	VM	Multi-dimensional resource constraint aware VM scheduling	• Uniform distribution • Google cluster • Production system	Computing energy	Simulation
Mhedheb [26] *ThaS*	✶◴	VM	Temperature-aware VM scheduling	50 tasks	• Computing energy • SLA violation	CloudSim
Beloglazov [25] *THR, IQR, MAD*	✶◴	VM	Utilization-aware VM scheduling	PlanetLab	• Computing energy • SLA violation	CloudSim
Ferreto [7]	✶◴	VM	Consolidation via linear programming and heuristic based VM migration	• TU-berlin datacenter • Google datacenter	• Number of used PMs • Number of migrations	Simulation
Proposed approach *GRANITE*	✶❄◴	**VM**	**• Greedy based VM allocation • Dynamic VM migration, • Dynamic cooling**	**Google Cloud datacenter (25 million tasks)**	**• Total energy • Job performance • CPU temperature**	**• Fluent • Enhanced CloudSim**

usage states of servers, Li et al. [15] presented a multi-dimensional space partition model, based on which they propose a VM placement algorithm called EAGLE to balance the utilization of multi-dimensional resources and reduce the number of running servers. Furthermore, a concept termed "skewness"is introduced by [39] to measure the unevenness in the multi-dimensional resource utilization of a server. By minimizing skewness, they can combine different types of workloads to reduce computing energy.

Apart from initial task/VM placement, scheduling systems needs to dynamically adjust their execution for better energy efficiency. This becomes particularly important as the development of virtualization technology which allows for live migrations of VMs. Workload co nsolidation via VM migration can be modelled with linear integer programming formulation [36]. Beloglazov and Buyya [25] presented various of algorithms to deal with key problems in migrations such as selecting VMs to migrate out from overloaded server and selecting new placement for migrated VMs. [27] detailed their methodology in detecting overload and determing the best time for migration.

Mhedheb et al. [26] combined computing energy reduction with CPU temperature management. Unfortunately, due to the incompleteness of models, they cannot explicitly connect CPU temperature with cooling capacity/status and propose any cooling energy aware scheduling algorithms. Moore et al. [8] proposed a system-level solution to control the heat generation through temperature-aware workload placement and reduce cooling energy. A sensor-based model to predict temperature distribution was proposed in [31], based on which Tang et al. [21] proposed an algorithm termed XInt to achieve cooling efficiency via minimizing heat recirculation and peak inlet temperature. But the model proposed in [31] only considers a stable status of airflow and workload distribution. Such work is incapable of capturing sudden fluctuation of workload and CRAC setting. Furthermore, the model only considers rack inlet temperature and does not model CPU temperature, which is a key indicator for thermal management [8], [12].

7.2 Hybrid Methods

CFD is a commonly deployed technique to model an entire datacenter with a specific layout [22], [23], [40]. Research in this field seeks to save energy and optimize datacenter operation at the hardware-level. Durand et al. [22] introduced a Proportional Integral Derivative (PID) algorithm to control the fan speed, combined with modeling servers and air conditioners. They consider the overall energy consumption in datacenters, and select the appropriate cooling temperature set point to reach the best compromise between energy consumption of chillers and servers. Different airflow configurations within a datacenter was studied by [10], who compared different airflow configurations and concluded that the vertical cooling schema "raised floor & ceiling return" was the most effective. Further studies show that datacenter racks with vertically placed servers attain enhanced cooling efficiency when vertical cooling schema is adopted [41].

Dynamic cooling is an effective means to enhance datacenter energy efficiency towards minimizing cooling power draw. A proactive control approach is proposed in [11] that jointly optimizes the air conditioner compressor duty cycle and fan speed to reduce cooling cost and minimize the risk of equipment damage due to overheating.

Banerjee et al. [16] proposed a coordinated cooling-aware job placement and dynamic cooling management policy, which is a typical hybrid method. It places jobs to reduce cooling demands upon CRACs, and updates CRAC thermostat settings based on temperature distribution, also found in [19], [31]. They adopt a greedy based job placement method that selects the server with the best cooling efficiency. Further, job scheduling policy proposed by [19] uses Integer Linear Programming (ILP) to produce optimal solution, coupled with dynamic retirement and employment to minimize both computing and cooling energy.

8 Conclusions and Future Work

This paper presents an in-depth model capturing the operational and thermal characteristics of Cloud datacenters. We detail the methodology used to identify key parameters for cooling model construction and validate its accuracy in simulation using realistic datacenter operational conditions. This model forms the foundation to propose a greedy-based VM scheduling algorithm named GRANITE. It comprises two separated components: initial placement and dynamic live migration that targets reducing total energy cost of cooling and servers while minimizing the likelihood of SLA violation. The algorithm is evaluated against numerous algorithms within CloudSim - a well-known tool for simulating Cloud datacenter. From the observations and experiment results presented within this paper, we draw the following conclusions.

To our knowledge, this is the first paper that captures CRAC cooling, datacenter thermal characteristics, CPU temperature and workload in a fine-grained manner. Our model is capable of modeling server temperature by capturing thermal characteristics of datacenter holistically. This model can be used by researchers in order to evaluate datacenter thermal operation in numerous configurations. Furthermore, we identify that cooling inlet temperature dynamically can be effectively modeled by exponential function corroborating similar characteristics to server level inlet temperature found within the RC model.

In our paper, we demonstrate that workload scheduling combining server status and datacenter thermal characteristics in a fine-grained manner is an effective way to reduce total energy draw. Experiment results show that GRANITE is capable of achieving lower levels of energy consumption by 21.0% compared to the state-of-the-art techniques. Our work explores the field of datacentre thermal management and energy saving in a sophisticated manner, analysing the reasons behind the effectiveness of GRANITE, laying the foundation for further research towards more accurate models and efficient scheduling algorithms in datacenters.

Future work includes further augmentation of the datacenter modeling, and the performance of scheduling algorithm. The CFD model we currently use is 2-dimensional. We aim to enhance by 3-dimensional CFD model yielding more accurate findings. Additionally, we plan to further explore other algorithms for workload scheduling and compare their results with respect to energy efficiency and task performance. The future candidate algorithms include genetic algorithm and linear programming, etc.

Acknowledgment

This work is supported by National High Technology Research 863 Major Program of China (No.2011AA01A207), National Science Foundation of China (No.61272128) and the program of China Scholarship Council (CSC).

References

[1] R. Brown, E. Masanet, B. Nordman et al., "Report to Congress on Server and Data Center Energy Efficiency: Public Law, 109-431," *Berkeley: Lawrence Berkeley National Laboratory*, technical report, 2008.

[2] K. Jonathan, "Growth in data center electricity use 2005 to 2010," 2011; http://www.analyticspress.com/datacenters.html

[3] L. Christian, "Projecting Annual new Datacenter Construction Market Size," Technique report, Microsoft Corp., 2010.

[4] L. A. Barroso, J. Clidaras, U. Hölzle, "The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines," *Synthesis lectures on computer architecture*, 2013, 8(3): 1-154.

[5] Y. C. Lee and A. Y. Zomaya, "Energy Efficient Utilization of Resources in Cloud Computing System," *The Journal of Supercomputing*, 2012, vol. 60(2), pp. 268-280.

[6] A. Murtazaev, S. Oh, "Sercon: Server Consolidation Algorithm Using Live Migration of Virtual Machines for Green Computing," *IEEE Technical Review*, 2011, 28(3): 212-231.

[7] T. C. Ferreto, M. Netto, R. Calheiros et al., "Server Consolidation With Migration Control for Virtualized Data Centers," *Future Generation Computer Systems*, 2011, 27(8): 1027-1034.

[8] J. Moore, J. Chase, P. Ranganathan et al., "Making Scheduling 'cool': Temperature-Aware Workload Placement in Data Centers," *USENIX Annual Technical Conf.*, 2005.

[9] R. Zhou, Z. Wang, C. Bash et al., "Data Center Cooling Management and Analysis-A Model Based Approach". *Semiconductor Thermal Measurement and Management Symposium* (SEMI-THERM), 2012: 98-103.

[10] S. Shrivastava, B. Sammakia, R. Schmidt R et al., "Comparative Analysis of Different Data Center Airflow Management Configurations," *Proc. of IPACK*, 2005.

[11] E. Lee, I. Kulkarni, D. Pompili et al., "Proactive Thermal Management in Green Datacenters," *The Journal of Supercomputing*, 2012, vol. 60(2), pp. 165-195.

[12] L. Wang, S. Khan, J. Dayal, "Thermal Aware Workload Placement with Task-Temperature Profiles in a Data Center," *The Journal of Supercomputing*, 2012, vol. 61(3), pp. 780-803.

[13] D. Jayasinghe, C. Pu, T. Eilam et al., "Improving Performance and Availability of Services Hosted on IaaS Clouds with Structural Constraint-Aware Virtual Machine Placement. *IEEE Intl. Conf. on Services Computing*, 2011: 72-79.

[14] A. Khosrav, S. Garg, R. Buyya," Energy and Carbon-Efficient Placement of Virtual Machines in Distributed Cloud Data Centers," *Euro-Par 2013 Parallel Processing*, 2013: 317-328.

[15] X. Li, Z. Qian, S. Lu et al., "Energy Efficient Virtual Machine Placement Algorithm with Balanced and Improved Resource Utilization in a Data Center," *Mathematical and Computer Modelling*, 2013, 58(5): 1222-1235.

[16] A. Banerjee, T. Mukherjee, G. Varsamopoulos et al., "Cooling-Aware and Thermal-Aware Workload Placement for Green HPC Data Centers," *Proc. of Green Computing Conf.*, 2010, pp. 245-256.

[17] L. Parolini, B. Sinopoli, B. Krogh et al., "A Cyber–Physical Systems Approach to Data Center Modeling and Control for Energy Efficiency," Proc. of the IEEE, 2012, 100(1): 254-268.

[18] Q. Tang, S. Gupta, G. Varsamopoulos, "Thermal-Aware Task Scheduling for Data Centers through Minimizing Heat Recirculation," *IEEE Int'l Conf. on Cluster Computing*, 2007: 129-138.

[19] E. Pakbaznia, M. Ghasemazar, M. Pedram. "Temperature-Aware Dynamic Resource Provisioning in a Power-Optimized Datacenter," *Proc. of the Conf. on Design, Automation and Test in Europe*, 2010: 124-129.

[20] X. Li, X. Jiang, K Ye, "Virtual Machine Scheduling Considering both Computing and Cooling Energy. *IEEE Int'l Conf. on High Performance Computing and Communications*, 2014: 244-247.

[21] Q. Tang, S. Gupta, G. Varsamopoulos, "Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Data Centers: A Cyber-Physical Approach," *IEEE Transactions on Parallel and Distributed Systems*, 2008, vol. 19(11), pp. 1458-1472.

[22] Durand, C. Bot, J. Manco et al., "Data Center Optimization Using PID Regulation in CFD Simulations," *Energy and Buildings*, 2013, vol. 66, pp. 154-164.

[23] A. Radmehr, B. Noll, J. Fitzpatrick et al., "CFD Modeling of an Existing Raised-Floor Data Center," *Semiconductor Thermal Measurement and Management Symposium* (SEMI-THERM), 2013: 39-44.

[24] R. Calheiros, R. Ranjan, A. Beloglazov et al., "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms," *Software: Practice and Experience*, 2011, 41(1): 23-50.

[25] A. Beloglazov, R. Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers," *Concurrency and Computation: Practice and Experience* (CCPE), 2012, vol. 24(13), pp. 1397-1420.

[26] Y. Mhedheb, F. Jrad, J. Tao et al., "Load and Thermal-Aware VM Scheduling on the Cloud," *Algorithms and Architectures for Parallel Processing*, 2013: 101-114.

[27] A. Beloglazov, Buyya R, "Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers under Quality of Service Constraints," *IEEE Transactions on Parallel and Distributed Systems*, 2013, 24(7): 1366-1379.

[28] R. Zhou, C. Bash, Wang et al., "Data Center Cooling Efficiency Improvement Through Localized and Optimized Cooling Resources Delivery. *Mechanical Engineering Congress and Exposition*, 2012, pp. 1789-1796.

[29] S. Zhang, K. Chatha, "Approximation Algorithm for the Temperature-Aware Scheduling Problem," *Proc. of the IEEE/ACM Int'l Conf. on Computer-Aided Design*, 2007, pp. 281-288.

[30] I. Moreno, P. Garraghan, P. Townend et al., "Analysis, Modeling and Simulation of Workload Patterns in a Large-Scale Utility Cloud," *IEEE Transactions on Cloud Computing*, 2014, 2(2): 208-221.

[31] Q. Tang, T. Mukherjee, S. Gupta et al., "Sensor-based Fast Thermal Evaluation Model for Energy Efficient High-Performance Datacenters," *Int'l. Conf. on Intelligent Sensing and Information*, 2006, pp. 203-208.

[32] W. Voorsluys, J. Broberg, S. Venugopal et al., "Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation," *Int'l Conf. on Cloud Computing*, 2009: 254-265.

[33] A. Avižienis, J. Laprie, B. Randell et al., "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Transaction on Dependable and Secure Computing*, 2004, 1(1): 11-33.

[34] D. Vanderster, Baniasadi, Dimopoulos, "Exploiting Task Temperature Profiling in Temperature-Aware Task Scheduling for Computational Clusters," *Advances in Computer Systems Architecture,* 2007, pp. 175-185.

[35] G. Liu, M. Fan, G. Quan et al., "On-Line Predictive Thermal Management under Peak Temperature Constraints for Practical Multi-Core Platforms," *Journal of Low Power Electronics*, 2012, vol. 8(5), pp. 565-578.

[36] K. Ye, Z. Wu, X. Jiang et al., "Power Management of Virtualized Cloud Computing Platform," *Chinese Journal of Computers*, 2012, 35(6): 1262-1285.

[37] C. Ghribi, M. Hadji, D. Zeghlache, "Energy Efficient VM Scheduling for Cloud Data Centers: Exact Allocation and Migration Algorithms," *IEEE/ACM Int'l Symp. on Cluster, Cloud and Grid Computing* (CCGrid), 2013: 671-678.

[38] E. Coffman, J. Csiri, Woeginger, "Approximate Solutions to Bin Packing Problems," *WOE-29, Institut FR Mathematik B, TU Graz, Steyrergasse 30,* A-8010. 2002.

[39] Z. Xiao, W. Song, Q. Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment," *IEEE Transactions on Parallel and Distributed Systems*, 2013, 24(6): 1107-1117.

[40] J. Summers, N. Kapur, H. Thompson, "Design of Data Centre Rack Arrangements Using Open Source Software," *Semiconductor Thermal Measurement and Management Symposium*, 2013: 45-51.

[41] Z. Song, X. Zhang, C. Eriksson, "Data Center Energy and Cost Saving Evaluation. *Proc. The 7th Int'l Conf. on Applied Energy*, 2015, pp. 1255-1260.

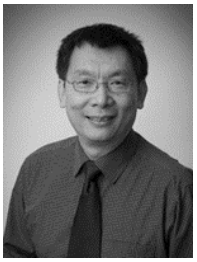
[42] N. Deng, C. Stewart, D. Gmach et al. "Policy and Mechanism for Carbon-Aware Cloud Applications," *IEEE Network Operations and Management Symposium*, 2012: 590-594.

Xiang Li is a PhD student in the Department of Computer Science and Technology at Zhejiang University since 2011. He received his BSc degree from Liaoning University in 2011. He was a visit scholar in the School of Computing, University of Leeds. His research interests focus on Cloud computing, energy-aware workload scheduling and datacenter modeling. He is a student member of IEEE.

Peter Garraghan is a Lecturer in the School of Computing & Communications, Lancaster University. He has industrial experience building large-scale systems and his research interests include distributed systems, Cloud datacenters, dependability, data analytics and energy-efficient computing.

Xiaohong Jiang received her B.Sc. and M.Sc. degree in computer science from Nanjing University and the Ph.D. degree in Zhejiang University, China, in 2003. She is an associate professor at the Department of Computer Science, Zhejiang University. Her research focuses on distributed systems, virtual environment, cloud computing, and data service.

Zhaohui Wu is the President of Zhejiang University, He is a chief scientist in the 973 Project, an information expert in the 863 Project and the team leader of the national panel of the modern service industry. He is the winner of the first-tier talent in the National Talents Program. His major research is focused on service computing, Cloud computing and ubiquitous computing, etc.



Jie Xu is a chair of computing and head of the ICSS at the University of Leeds. He is the director of the UK EPSRC WRG e-Science Centre. He is also a guest professor of Beihang University, China. He has published more than 300 academic papers in areas related to dependable distributed systems. He has led or coled many research projects to the value of more than $30M. He is a member of the IEEE.