IEICE TRANS. INF. & SYST., VOL.E101–D, NO.9 SEPTEMBER 2018

PAPER

Trading-Off Computing and Cooling Energies by VM Migration in Data Centers

Ying SONG[†a)], ***Member***, **Xia ZHAO**[††b)], **Bo WANG**[†††c)], ***and*** **Yuzhong SUN**[††††d)], ***Nonmembers***

SUMMARY High energy cost is a big challenge faced by the current data centers, wherein computing energy and cooling energy are main contributors to such cost. Consolidating workload onto fewer servers decreases the computing energy. However, it may result in thermal hotspots which typically consume greater cooling energy. Thus the tradeoff between computing energy decreasing and cooling energy decreasing is necessary for energy saving. In this paper, we propose a minimized-total-energy virtual machine (VM for short) migration model called C^{2}vmMap based on efficient tradeoff between computing and cooling energies, with respect to two relationships: one for between the resource utilization and computing power and the other for among the resource utilization, the inlet and outlet temperatures of servers, and the cooling power. Regarding online resolution of the above model for better scalability, we propose a VM migration algorithm called C^{2}vmMap_heur to decrease the total energy of a data center at run-time. We evaluate C^{2}vmMap_heur under various workload scenarios. The real server experimental results show that C^{2}vmMap_heur reduces up to 40.43% energy compared with the non-migration load balance algorithm. This algorithm saves up to 3x energy compared with the existing VM migration algorithm.

key words: *energy saving, trading off, computing energy, cooling energy, VM migration*

1. Introduction

Currently, tremendous amount of energy is consumed by data centers. Greenpeace estimates the data center electricity demand is at around 31GW globally, which is equivalent to nearly 180,000 homes' supply [1]. Gartner reports energy costs account for 12% of the total costs of data centers, and energy costs associated with cooling data centers account for one-third to over half of data center energy consumption [2]. In other words, most energy is used to compute and cool in data centers. Thus, a large number of researchers focus on decreasing the computing or cooling energy consumption in data centers, wherein workload, in the form of VM or not, scheduling is an important method.

However, the objectives of most of the above work are to reduce either the computing energy or the cooling energy. Consolidating workload onto fewer servers decreases the computing energy. However, it may result in thermal hotspots which typically consume greater cooling energy [3]. Moreover, in some cases, such cooling energy increasing outweighs the computing energy decreasing. Some work such as Mooa [4] and VMAP [5] simultaneously consider the computing and cooling energies. However, they do not trade off the two metrics well. Mooa [4] needs the system administrators to specify the weights of different conditions.

In this paper, we propose a VM migration model called C^{2}vmMap that can minimize the total energy consumption and a VM migration algorithm called C^{2}vmMap_heur based on efficient tradeoff between computing and cooling energies. C^{2}vmMap is a nonlinear integer programming problem which aims to find a new mapping of VMs to physical servers ('VM-PM mapping' for short) to minimize the total energy consumption as well as guarantee the resource requirements of the hosted applications. Guaranteeing the resource requirements of the hosted applications will not violate SLAs of these applications at runtime. This problem is NP-hard [6], thus its resolution faces the conflict between online computing and scalability. We present a heuristics method called C^{2}vmMap_heur to solve C^{2}vmMap. Difference between the new mapping solved from the model and the initial mapping guide the VM migration algorithm. The real server experimental results show that C^{2}vmMap_heur reduces up to 53.2% computing energy, 30.3% cooling energy as well as 40.43% total energy, compared with the non-migration load balance algorithm. This algorithm saves up to 3x energy compared with the existing VM migration algorithm. The simulation experimental results illustrate that C^{2}vmMap_heur reduces up to 54% energy in the scale of 500 PMs compared with the non-migration load balance algorithm.

This paper has the following contributions. a) We propose C^{2}vmMap model using optimization theory based on the relationships among the resource utilization, the inlet and outlet temperature of servers, the computing power and the cooling power to guide the design of the VM migration algorithms. b) To decrease the computational complexity of C^{2}vmMap, the heuristics method was developed to re-

Manuscript received October 5, 2017.
Manuscript revised March 13, 2018.
Manuscript publicized June 1, 2018.

[†]The author is with the Computer School, Beijing Information Science & Technology University, Beijing Key Laboratory of Internet Culture and Digital Dissemination, Beijing, China.
[††]The author is with the Beijing Key Laboratory of Big Data Technology for Food Safety, School of Computer and Information Engineering, Beijing Technology and Business University, Beijing, China.
[†††]The author is with the Software Engineering College, Zhengzhou University of Light Industry, Zhengzhou, China.
[††††]The author is with the State Key Laboratory of Computer Architecture, Institute of Computing Technology, CAS, Beijing, China.
a) E-mail: songying@bistu.edu.cn
b) E-mail: zhaox@btbu.edu.cn (Corresponding author)
c) E-mail: wangb@zzuli.edu.cn
d) E-mail: sunyuzhong@ict.ac.cn
DOI: 10.1587/transinf.2017EDP7329

duce the C^{2}vmMap solution into one sorting of PMs and one sorting of VMs following some linear computations and comparisons. c) We implement C^{2}vmMap_heur in the above way and a simulator with the objective of evaluating the scalability of our algorithm.

2. Related Work

Currently, a large body of research is about energy saving by workload scheduling. We classify such research into three sub-fields. The works in the former two sub-fields differ from our work, because they focused on energy saving by reducing either the computing energy or the cooling energy separately. Our work trades off the computing and cooling energies concurrently, which falls into the third sub-field.

2.1 Computing Energy Saving Scheduling Methods

Some research approaches are energy-oriented workload dispatch [7]–[9], [34] and server consolidation [10]–[17], [33] which dynamically adjust the active server set in order to turn off a portion of servers and save computing energy without compromising the quality of service.

2.2 Cooling Energy Saving Scheduling Methods

UOP, MCE, UT [18] and ZBD [19] are energy or thermal-aware scheduling, ignoring the cooling efficiency of the cooling device, which may create thermal hotspots. Reference [20] sorts all servers according to the cooling efficiency of their locations, and then preferentially allocates jobs to servers with higher cooling efficiency. HTS [21] reduces the total energy of a data center by reducing the cooling energy. MinHR [19] and XInt [22] minimize the peak temperature by minimizing the total recirculation of hot air. TASA [23] aims to reduce cooling power consumption by allocating "hot" jobs to "cold" servers with the cost of prolonging response time of jobs. CoolProvision [24] selects the cheapest provisioning within performance constraints. In order to tackle the cooling inefficiencies of datacenters, ATAC [25] senses the ambient temperature of each server and triggers a performance capping mechanism.

Google Datacenter tries to reduce cooling energy efficiency by a neural network technique [36]. They trained three deep neural networks to predict the average future PUE (Power Usage Effectiveness), the future temperature and pressure of the data centre over the next hour. The purpose of these predictions is to simulate the recommended actions from the PUE model, to ensure that they do not go beyond any operating constraints. In 2015, we cooperated with HUAWEI Co., Ltd. to propose and implement the technology of reducing the cooling energy consumption of data center through virtual machine migration [35]. This technology we called E_Saving_Huawei in the following has become an important energy saving feature of HUAWEI ManageOne. Such work is the basis of our work.

2.3 Total Energy Saving Scheduling Methods

A.M. Al-Qawasmeh et al. [26] reduce energy by prolonging the execution time of jobs. Such work constrains the power under the red-line but may increase total energy in some cases. Differing from this work, our work reduces the total energy by migrating workloads to some servers with higher computing and cooling energy efficiency.

References [27] and [28] focus on the scheduling at the level of tasks to cores. Mooa [4] simultaneously optimizes multiple objectives, it dynamically places VMs only using the weighted sum of temperature, utility, and power efficiency. However, the weights of different conditions are specified by the system administrators and different conditions may conflict with each other. Differing from Mooa, our work automatically and concurrently trades off the computing and cooling energies. VMAP [5] is a VM consolidation technique to maximize computing resource utilization, to minimize datacenter energy consumption for computing, and to improve the efficiency of heat extraction. VMAP selects the source and target servers of migrations according to the relationship of the heat generation and extraction of servers, which differs from our work. However, such relationship only denotes the increasing or decreasing of the temperature of the server but not the value of the future temperature of the server. Such decision may result in new thermal hotspots.

TACOMA [29] and [30] leverage the correlation between the power consumption of servers and CRACs. However, both works only focus on the workload distribution when the request arrives under the scenario of the web traffic, ignoring the run-time workload migration among servers to find the most energy-saving solution. Similarly, DartScheduler [31] presents the VM deployment method exploring the tradeoff between load consolidation and balance. Reference [30] considers the temperature at the level of rack which is much coarser than our work.

3. Minimized-Total-Energy VM Migration Model

In order to address the above problem of energy saving in data centers, we analyze the relationship between the resource utilization and the computing power and the relationship among the resource utilization, the inlet and outlet temperatures of servers, and the cooling power. Based on such analysis, we design a nonlinear integer programming model to find a new VM-PM mapping to minimize the total energy of a data center. First, we introduce the notations and assumptions. Then, the nonlinear integer programming model is presented.

3.1 Notations and Assumptions

Table 1 summarizes all the notations used in this paper. We explain the assumptions and relationships as follows.

Table 1 Notations

Notations	Description
M	Number of PMs
N	Number of VMs
$y_i(t)$	Server i is power on ($y_i(t) = 1$) or off ($y_i(t) = 0$) at time t
R	Types of resources, $R \in \{c, m, d, n\}$, where c, m, d and n denote CPU, memory, disk and NIC respectively.
$P\_R\_PM_i$	The total amount of resource R of server i
$a_{ij}(t)$	The mapping of VM_i and PM_j at time t. $a_{ij}(t) = 0$ denotes PM_j does not host VM_i at time t; $a_{ij}(t) = 1$ denotes PM_j hosts VM_i at time t
$Req\_R\_VM_i$	The amount of resource R required by VM_i, which is specified by the creator of the VM when it is created
$P_{Total}(t)$	The total power used by the server set and CRACs in a data center at time t
$P_{comp}(t)$	The computing power used by server set in a data center at time t
$P_{cool}(t)$	The cooling power used by the CRACs in a data center at time t
$T_{sup}(t)$	the supplying temperature of the CRACs at time t
$T^j_{out}(t)$	the outlet temperature of server j at time t
$T^j_{in}(t)$	the inlet temperature of server j at time t
$u^R_j(t)$	The utilization of resource R of server j at time t
$u^R_{j,k}(t)$	The utilization of resource R of virtual machine k hosted on server j at time t
$s_j(t)$	the speed of fans of server j at time t
P^i_{idle}	The power consumed by server i when it is idle
P^i_{busy}	The power consumed by server i when it is busy (namely the CPU utilization is 100%)

3.1.1 Assumptions

(1) Assumption 1

The total power ($P_{Total}(t)$) consumed by the server set and CRACs in a data center at time t is the sum of the consumed computing power ($P_{comp}(t)$) and cooling power ($P_{cool}(t)$) at time t, namely,

$$P_{Total}(t) = P_{cool}(t) + P_{comp}(t) \tag{1}$$

The VM migration processes and temperature changes of source and target nodes of migrations take some time, we set x to be the sum of the time of VM migrations and the time of temperature changes of source and target nodes.

(2) Assumption 2

The temperature of the supplied cooled air at time t, denoted as $T_{sup}(t)$, should be low enough so that the outlet temperatures of the computing nodes do not go beyond the red line temperatures ($T^j_{out\_red}$) which are specified by the manufacturers and remain unchanged. In general cases, the red line temperatures of various servers are close. In order to simplify the expression, we assume the red line temperatures of various servers are the same one, namely T_{out_red}.

$$T^j_{out\_red} \le T_{out\_red}, \forall j \tag{2}$$

In the ideal case, $T^j_{out\_red} = T_{out\_red}$ and all the hot air should directly go back to the CRAC. Then $T^j_{in}(t)$, the inlet temperature of server j at time t can be written as:

$$T^j_{in}(t) = T_{sup}(t) + \Delta T^j(t) \tag{3}$$

Where $\Delta T^j(t)$ is close to zero. Thus, all $T^j_{in}(t)$ are close.

(3) Assumption 3

Cooling energy of the CRACs can be modeled by the coefficient of performance (CoP), which is the ratio of the heat removed (i.e., computing energy) over the work required to remove that heat (i.e., cooling energy) [29]. CoP is usually increasing function of the supplied temperature, e.g., for an HP data center CoP reported as $CoP(T_{sup}) = 0.0068T^2_{sup} + 0.0008T_{sup} + 0.458$ [19]. The cooling power at time t, denoted by $P_{cool}(t)$, can be written as a function of the CoP of the supplied temperature:

$$P_{cool}(t) = P_{comp}(t)/CoP(T_{sup}(t)) \tag{4}$$

(4) Assumption 4

The outlet temperature of server j is infected by the inlet temperatures of server j, the heat generated by all resources of such server which is determined by the utilizations of these resources, and the heat extraction which is determined by the speed of fans of such server.

$$T^j_{out}(t) = T^j_{in}(t) + f^j_a(u^c_j(t), u^m_j(t), u^d_j(t), u^n_j(t)) - f^j_b(s_j(t)) \tag{5}$$

f^j_a and f^j_b represent the functions of the heat generation and the heat extraction, respectively, to be convenient for describing the relationship of maximum outlet temperature of servers and the supplying temperature of CRACs. The form of these functions has no effect on the application of our proposed method, and their definition or construction, which is out of our paper's scope, has been studied by several researchers. We get the following relationship based on expressions (2), (3) and (5).

$$T_{sup}(t) + \Delta T^j(t) + T^j_{in}(t) + f^j_a(u^c_j(t), u^m_j(t), u^d_j(t), u^n_j(t)) - f^j_b(s_j(t)) \le T_{out\_red}, \forall j \tag{6}$$

We get expression (7) from expression (6).

$$\max_j \{T_{sup}(t) + \Delta T^j(t) + T^j_{in}(t) - f^j_b(s_j(t)) + f^j_a(u^c_j(t), u^m_j(t), u^d_j(t), u^n_j(t))\} \le T_{out\_red} \tag{7}$$

From expression (7) we can draw a conclusion: the more heat accumulated by each server is (namely the heat generation minus heat extraction), the lower supplying temperature of CRACs is needed under the idea case in which $\Delta T^j(t)$ is

close to zero and all $T^j_{in}(t)$ are close. Expression (5) denotes that the outlet temperature of a server is primarily determined by the accumulated heat of such server. Thus, the lower the maximum outlet temperature of all servers is, the higher supplying temperature of CRACs is needed. We assume that the increased value of the supplying temperature of CRACs equals to the decreased value of the maximum outlet temperature of all servers.

(5) Assumption 5

All CRACs setting to the same temperature are not powered off, even if all servers are turned off.

(6) Assumption 6

The outlet temperature of server j at time $t + x$ can be predicted by the inlet and outlet temperatures of server j at time t, the utilizations of resource R of server j at time t and at time $t + x$ and the speed of fans of server j at time t, namely, $T^j_{out}(t + x) = f^j_c(T^j_{in}(t), T^j_{out}(t), u^c_j(t), u^m_j(t), u^d_j(t), u^n_j(t), u^c_j(t + x), u^m_j(t+x), u^d_j(t+x), u^n_j(t+x), s_j(t))$. In the implementation of our algorithms,

$$\begin{aligned} f^j_c(T^j_{in}(t), T^j_{out}(t), u^c_j(t), u^m_j(t), u^d_j(t), u^n_j(t), u^c_j(t+x), \\ u^m_j(t+x), u^d_j(t+x), u^n_j(t+x), s_j(t)) = c_0 + c_1 \times T^j_{in}(t) \\ + c_2 \times T^j_{out}(t) + c_3 \times u^c_j(t) + c_4 \times u^m_j(t) + c_5 \times u^d_j(t) \\ + c_6 \times u^n_j(t) + c_7 \times u^c_j(t+x) + c_8 \times u^m_j(t+x) \\ + c_9 \times u^d_j(t+x) + c_{10} \times u^n_j(t+x) + c_{11} \times s_j(t) \end{aligned} \quad (8)$$

where $c_0 \sim c_{11}$ are obtained using off-line linear regression method. As to the multicore CPU server, the CPU utilization is the average utilization of multi cores. Such assumption is verified by our previous experiments. The maximum prediction errors are 0.15°C and 1.54°C, and the average prediction errors are 0.03°C and 0.24°C for $x = 5s$ and $x = 60s$ respectively in our test. Of course, we can use any other prediction methods with better accuracy in our model.

(7) Assumption 7

The computing power at time t is determined by the CPU utilizations of the active servers at time t, and the power consumed by each active server when it is idle and busy [32], namely,

$$\begin{aligned} P_{comp}(t) = \sum\nolimits_{j=1}^{M} \{ & y_j(t) \cdot (P^j_{idle}(t) \cdot (1 - u^c_j(t))) \\ & + P^j_{busy}(t) \cdot u^c_j(t)\} \end{aligned} \quad (9)$$

As to server j, if $\sum_{i=1}^{N} a_{ij} = 0$, then $y_j(t) = 0$, else $y_j(t) = 1$.

(8) Assumption 8

During migration, there are some extra resource utilizations at both the source and destination servers, which is provisional and unremarkable. Such utilizations bring to extra power cost which is ignorable compared to the power changing after migration in our experiments. Thus, we do not care about the extra resource utilizations during migration. The amounts of resources used before and after migration (namely, at time t and time t+x) by each VM are the same.

3.1.2 Relationships

(1) Relationship 1

$a_{im}(t) = 1$ and $a_{ij}(t + x) = 1$ denote VM_i migrates from server m to server j from time t to $t + x$.

(2) Relationship 2

In a heterogeneous data center, when a VM migrates from one server to the other, its resource utilization may change because of the different maximum capacities of the source and target servers of migration. In order to avoid heavy overhead from a large number of VM migrations, as to each server, it can only be the source node or the target node at one round migration in our migration model. Thus, as to the difference in resource utilization after VM migration under Assumption 8, there are the following relationships.

As to the source servers of migration, e.g., server j,

$$u^R_j(t + x) = u^R_j(t) - \sum_{k \in A1} u^R_{j,k}(t), \quad (10)$$

where $R \in \{c, m, d, n\}$ and $A1 = \{i | a_{ij}(t + 1) = 0, a_{ij}(t) = 1, i = 1, \ldots, N\}$.

As to the target servers of migration, e.g., server j,

$$\begin{aligned} u^R_j(t + x) &= u^R_j(t) + \sum_{k \in A2} u^R_{j,k}(t) \\ &= u^R_j(t) + \sum_{k \in A2} (u^R_{m,k}(t) \cdot P\_R\_PM_m / P\_R\_PM_j), \end{aligned} \quad (11)$$

where $R \in \{c, m, d, n\}$ and $A2 = \{i | a_{ij}(t + 1) = 1, a_{ij}(t) = 1, i = 1, \ldots, N, m = 1, \ldots, j - 1, j + 1, \ldots, M\}$.

3.2 Minimized-Total-Energy VM Migration Model

The objective of the optimization problem is,

Minimize: $P_{total}(t + x)$,
Subject to: **C1, C2, C3, C4**.

The first constraint (**C1**) ensures that any VM is allocated to one and only one server at the same time, i.e.,

C1: $a_{ij}(t), y_j(t) \in \{0, 1\}, \forall i, \forall j, \forall t$;
$\sum_{j=1}^{M} a_{ij}(t) = 1, \forall i, \forall t$;
$y_j(t) \le \sum_{i=1}^{N} a_{ij}(t)$;
$y_j(t) \ge a_{ij}(t), \forall i, \forall j$.

The second constraint (**C2**) ensures that the resource requirements of all VMs hosted by one server do not exceed the maximum capacity of this server, namely, as to resource set $R = \{c, m, d, n\}$, and $P\_R\_PM_j \ge 0, Req\_R\_VM_i \ge 0, \forall i, \forall j$. Such constraint which is also used by other VM

migration work [5], [11] guarantees the resource requirements of the hosted applications, i.e.,

$$\textbf{C2:}\ \sum_{i=1}^{N} a_{ij}(t) \cdot Req\_R\_VM_i \le P\_R\_PM_j, \forall j, \forall t;\quad P\_R\_PM_j \ge 0, Req\_R\_VM_i \ge 0, \forall i, \forall j.$$

The third constraint (**C3**) is to avoid heavy overhead from a large number of VM migrations. As to each server, it can only be the source node or the target node at one round migration, namely, if exist $a_{ij}(t+1) - a_{ij}(t) \ge 0$, then as to any i, $a_{ij}(t+1) - a_{ij}(t) \ge 0$; if exist $a_{ij}(t+1) - a_{ij}(t) \le 0$, then as to any i, $a_{ij}(t+1) - a_{ij}(t) \le 0$. Such constraint can be converted to:

$$\textbf{C3:}\ -1 \le (a_{ij}(t+1) - a_{ij}(t)) - (a_{kj}(t+1) - a_{kj}(t)) \le 1, \forall j, \forall i, \forall k.$$

The fourth constraint (**C4**) is to ensure that the temperature of each server does not climb above a preset limit, i.e.,

$$\textbf{C4:}\ \ T^j_{out}(t+x) \le T_{out\_red}, \forall j.$$

We can calculate $P_{total}(t+x)$ using the expressions (1)–(11). The given inputs are $a_{ij}(t), u^R_j(t), u^R_{j,k}(t), Req\_R\_VM_i$, $P\_R\_PM_j, T^j_{in}(t), T^j_{out}(t), T_{out\_red}, s_j(t), P^j_{idle}, P^j_{busy}$. Taking these inputs into the above programming formulation, we get $a_{ij}(t+1)$ which satisfy the minimization objective.

4. Minimized-Total-Energy VM Migration Algorithms

The above nonlinear integer programming problem C^{2}vmMap is NP hard [6], thus its solution faces the conflict between the online computing and the scalability. Regarding the online computing of VM migration, we present a heuristic resolution method for the C^{2}vmMap model. Based on this resolution method, we design and implement a VM migration algorithm, called C^{2}vmMap_heur aimed to reduce the total energy consumption in a data center. This algorithm makes VM migration decisions according to the difference between the initial VM-PM mapping ($a_{ij}(t)$) and the solved mapping ($a_{ij}(t+1)$).

The objective of the above model is to minimize the total power of the new VM-PM mapping. We sort PMs by largest metric $(P^j_{idle} + (P^j_{busy} - P^j_{idle}) \cdot u^c_j(t) + f^j_a(\Delta T^j(t)))/(\sum_R(\beta_R \cdot P\_R\_PM_j))$ first. Such metric denotes the change of the maximum computing and cooling powers incurred by unit capacity of server j. Such metric can evaluate a server in the power efficiency combining computing power with cooling power. The smaller the metric is the higher power efficiency the server will be. $\Delta T^j(t) = T^j_{out}(t) - \min(T^k_{out}(t))$, where $k = 1 \ldots M$. The capacity of server j in such metric is denoted by $\sum_R(\beta_R * P\_R\_PM_j)$, where β_R denotes the degree of resource R contributing to the computing and cooling power within a server. We set $\beta_{cpu} = 1, \beta_{mem} = 0.5, \beta_{disk} = 0.5, \beta_{net} = 0.1$ in the implement of C^{2}vmMap_heur according to our experimental experience. The main ideas of C^{2}vmMap_heur are preferential migrating VMs from servers with lower power efficiency to servers with higher power efficiency if the predicted total power using the expressions (1)–(11) decreases, and turning off servers if no VM is hosted by them. As to the computational time complexity of C^{2}vmMap_heur, it sorts all PMs and then sorts all hosted VMs for each PM and makes decision for each VM migrated to each PM. Thus, the computational time complexity of C^{2}vmMap_heur is $O(M^2 n log_2 n)$, wherein M is the number of PMs, and n is the largest number of VMs hosted by each PM.

Although Hermenier et al. [11] validate that a VM migration has little impact on the overall performance. C^{2}vmMap_heur may result in a large number of VM migrations. And turning servers on/off have non-negligible overheads. Thus, it should not be executed frequently. We separate the migration commands from the algorithm. The separated algorithm makes decisions periodically, e.g., 120 seconds. Only when the new VM-PM mapping is predicted to bring remarkable energy saving, e.g. larger than 10%, the migration commands will be executed.

Algorithm 1 C^{2}vmMap_heur

Input: $a_{ij}(t), u^R_j(t), u^R_{j,k}(t), Req\_R\_VM_i, P\_R\_PM_j, T^j_{in}(t), T^j_{out}(t), T_{out\_red}, s_j(t), P^j_{idle}, P^j_{busy}$

```
while true do
    collect the input informantion;
    sort PMs by largest metric (P^j_idle + (P^j_busy − P^j_idle) · u^c_j(t) + f^j_a(ΔT^j(t)))/(∑_R(β_R · P_R_PM_j)) and put the sorted PMs into a sorted_PM_queue first;
    for each PM_i from the head of sorted_PM_queue and PM_i not in the target_PM_queue do
        sort VMs hosted by the selected source PM_i by largest CPU utilization of the VM first;
        for each VM in sorted VM queue do
            add the selected VM to the wairing-to-migrate VM list;
            select PM_j with the minimum metric from sorted_PM_queue and satisifying the resource requirement of the added VM to be the target PM of migration;
            compute P_Total(t + x) and T^j_out(t + x) using the expressions (1)–(11) according to the current mapping and the original one;
            if T^j_out(t + x) ≤ T_out_red then
                record the minimum P_Total(t+x) and the corresponding mapping;
            end if
        end for
        record the minimum P_Total(t + x) and the corresponding mapping for the selected PM;
        if all VMs hosted by the PM need to migate out then
            this PM is deleted from the sorted PM queue;
        end if
        put the selected target PMs into the target_PM_queue.
    end for
    get the new VM-PM mapping a_ij(t + 1);
    send commands of VM migration according to the difference between a_ij(t) and a_ij(t + 1);
end while
```

5. Performance Evaluation

5.1 Testbed and Experiment Design

5.1.1 Real Server Testbed

We use 25 servers in our experiments, wherein 22 servers forming a heterogeneous server pool. In the server pool, one server is used to store the image files of VMs, one server is used to be a management node running the VM migration algorithm, and 20 servers are used to be the managed nodes which host VMs. The rest three servers are used to emulate clients of services. All servers are connected with a Gigabit Ethernet.

We use a plain 2.6.18 Linux kernel that comes with the CentOS5.5 standard distribution for the management and storage servers. We use CentOS5.5 and Xen-3.0.4 for the managed servers. We collect P_{comp} before and after migrations by an electric parameter tester, which measures the power consumed by all servers switching in it. We calculate P_{cool} by using Assumptions 2, 3 and 4 and the CoP of HP data center [19] in Sect. 3.1 with the input of the monitored outlet temperatures of servers before and after migrations. All the inlet and outlet temperatures of servers are measured by the temperature monitoring system which collects the temperature information using sensors, developed by us.

5.1.2 Simulation Testbed

We developed a simulator to evaluate the scalability of our algorithm. We use one physical server to run the simulator. The simulator generate the specified number and workload of managed node information including the resource utilization, the inlet and outlet temperatures, the idle and busy powers and the hosted VM resource requirement and utilization information based on the real information collected from our 20 physical servers under various workload scenarios to be the input. It outputs the computing powers, the maximum outlet temperatures, the cooling powers, and the total powers before and after VM migrations, as well as the decision time of the VM migration algorithm.

In order to evaluate the accuracy of our simulator, we compare the simulated results and the real results of 20 PMs under various workload scenarios with the same input and the same VM migration algorithm C^{2}vmMap_heur. From Table 2 we can see that the simulating results are acceptable with the less than 9% and 0.5°C maximum errors of the decreased total power and the predicted hotspot temperature respectively. Our simulator computes the computing power using expression (9) introduced in Sect. 3.1.1 which only takes the CPU utilization into account. Such computation may contribute to the main errors of decreased computing power. At the same time, our simulator ignores the extra computing power resulting from the processing of VM migration. Such extra computing power also contributes to the errors.

Table 2 Maximum and average errors of the simulator

Types of Errors	Maximum and Average Errors
Errors of the decreased computing power	12.18%, 7.27%
Errors of the decreased cooling power	2.17%, 1.49%
Errors of the decreased total power	8.71%, 4.37%
Errors of the predicted hotspot temperature	0.47°C, 0.28°C

5.1.3 Experiment Design

The experiments are designed with the goal of evaluating C^{2}vmMap and C^{2}vmMap_heur. In our real server testbed, we create different number of VMs to reflect different average workloads of servers. We allocate 1G memory and 2 vCPUs to each VM. There are the following three applications in our experiments. Each copy of them runs in a VM.

1) Web service: Apache is used for the Web server. LVS dispatches requests among the Web VMs using round robin algorithm. SPECWeb2005 is used to generate e-commerce workloads. 2) HPC application: We run a matrix multiplication to generate sequential HPC workloads. 3) Office application: We emulate the real-world office applications based on the traces collected from the real desktop operations of 18 members in our research team in three days. The percentage of web services, HPC applications and office applications is 3:6:1, which is close to the actual percentage in our data center.

We evaluate C^{2}vmMap_heur from two points of view. Evaluation 1 compares this algorithm with VMAP [5] in the three workload scenarios illustrated in Table 3 and compares this algorithm with E_Saving_Huawei [35] in the middle workload scenario. Such three workload scenarios mean the load sizing of a data center. For example, the quantity of requests arriving at Web service per second for SPECWeb. Such workload sizing depends on the number of VMs. The average CPU and memory utilizations (denoted by cpu_utilization and mem_utilization) of different workload scenarios are measured when the applications are running. In these three scenarios, we map VMs to PMs according to a traditional workload balancing policy with the goal of balancing the rate of the required resources by the hosted VMs and the maximum capacity of each server in the initial status. Evaluation 2 analyzes the scalability of this algorithm by adjusting the scale of PMs using the developed simulator.

Table 3 Workload scenarios of experiments

	VM_num	cpu_utilization	Mem_utilization
Light workload	100	18%	66%
Middle workload	150	22%	76%
Heavy workload	200	36%	93%

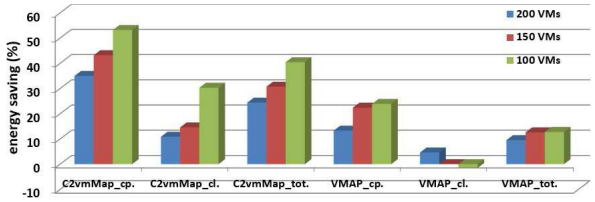


Fig. 1 Energy saved by C^{2}vmMap_heur and VMAP on 20 PMs. C^{2}vmMap_cp, C^{2}vmMap_cl and C^{2}vmMap_tot denote the computing, cooling and total energies saved by C^{2}vmMap_heur respectively. VMAP_cp, VMAP_cl and VMAP_tot denote the computing, cooling and total energies saved by VMAP respectively.

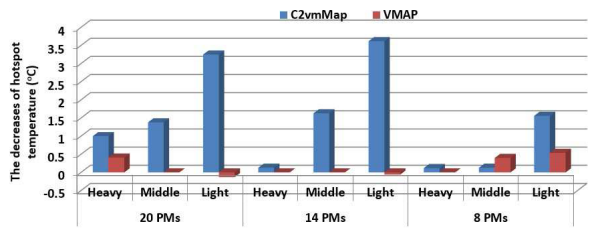


Fig. 2 The decreases of thermal hotspot temperature after VM migrations controlled by C^{2}vmMap_heur and VMAP, wherein 20 PMs, 14 PMs and 8 PMs denote the number of active PMs before migrations.

5.2 The Experimental Results and Analysis

5.2.1 Evaluation 1: Performance Comparison

Comparison-I: C^{2}vmMap_heur vs. VMAP

In the three workload scenarios illustrated in Table 3, we run C^{2}vmMap_heur and VMAP respectively to control VM migrations based on the same initial status. We evaluate them by the metrics of the computing energy saved, the cooling energy saved and the total energy saved compared with the initial status. We analyze the rate of migrated VMs, the numbers of active servers, the changes of thermal hotspot temperatures and the performance of the hosted applications of these three scenarios before and after VM migrations controlled by these two algorithms. We also analyze the cost of migration, i.e. migration time and energy consumption, and the energy cost of the management server running our algorithms.

From Fig. 1 we can see the lighter the workload is, the more energy is saved by these algorithms. C^{2}vmMap_heur reduces up to 53.2% computing energy, 30.3% cooling energy as well as 40.43% total energy, compared with the non-migration load balance algorithm. The computing energy and cooling energy saved by C^{2}vmMap_heur are most remarkable in the light workload scenario. It saves up to 3x total energy compared with VMAP.

As to the decreases of cooling energy, the decreasing of thermal hotspot temperature is the decisive factor. Figure 2 illustrates the decreases of thermal hotspot temperatures after one round VM migrations controlled by these two algorithms. The decreasing of thermal hotspot temperature provided by C^{2}vmMap_heur is most remarkable in the

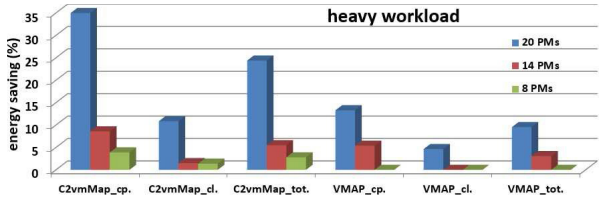


Fig. 3 Energy saving provided by C^{2}vmMap_heur and VMAP in heavy workload scenario.

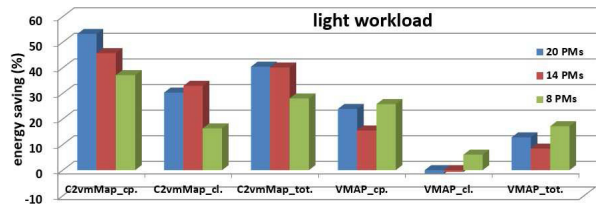


Fig. 4 Energy saving provided by C^{2}vmMap_heur and VMAP in light workload scenario.

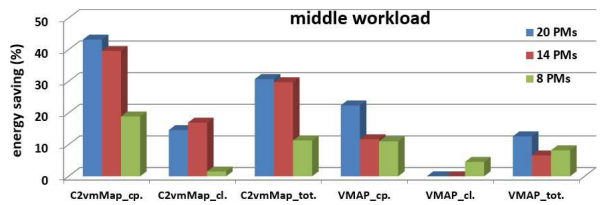


Fig. 5 Energy saving provided by C^{2}vmMap_heur and VMAP in middle workload scenario.

light workload scenario, which is consistent with the most remarkable cooling energy saved by it in such scenario. As to VMAP, the computing energy saving is much more than the cooling energy saving, especially in the light workload scenario. The server whose outlet temperature is the highest is determined by VMAP to migrate in some VMs if its heat generation is less than its heat extraction, which results in the increasing of cooling energy in the light workload scenario. VMAP prefers the initial scenario in which all outlet temperatures are close to each other. In such scenario, the outlet temperature of the server whose heat generation is more than its heat extraction will increase, and VMAP avoids from the new thermal hotspot by migrating VMs out from this server. However, in our 20 PMs experimental environment the outlet temperatures of some servers are much higher than others. The server whose heat generation is less than its heat extraction may be the thermal hotspot, which may result in cooling energy lost by VMAP. In order to validate such analysis, we select 8 PMs with the close initial outlet temperatures to evaluate VMAP. The experimental results (illustrated in Fig. 3, Fig. 4 and Fig. 5) show that the cooling energy is saved by VMAP in all the three workload scenarios. As to the computing energy saving, C^{2}vmMap_heur outperforms VMAP. The degree of computing energy saving with steady workload depends on two factors, one is the number of servers which is powered down, and the other is the energy efficiency of active servers. Fig-

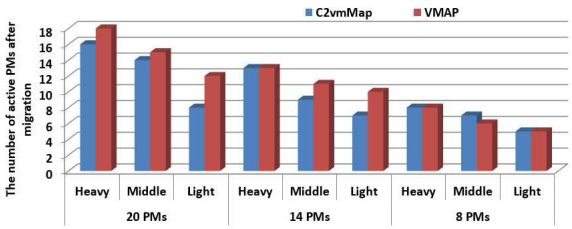


Fig. 6 The number of active PMs before and after VM migrations controlled by C^{2}vmMap_heur and VMAP, wherein 20 PMs, 14 PMs and 8 PMs denote the number of active PMs before migrations.

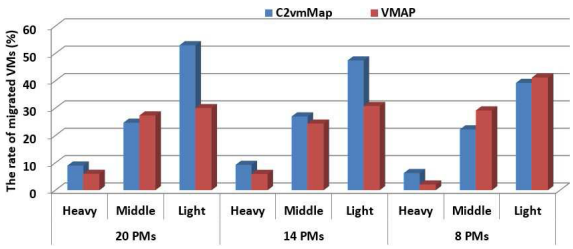


Fig. 7 The rate of migrated VMs determined by C^{2}vmMap_heur and VMAP in various workload scenarios, wherein 20 PMs, 14 PMs and 8 PMs denote the number of active PMs before migrations.

ure 6 illustrates the number of active servers before and after one round VM migrations controlled by the two algorithms. In most cases, VMAP results in more active servers than C^{2}vmMap_heur, because VMAP select servers to migrate out VMs according to the relationship of the heat generation and extraction. The servers whose heat generation is more than extraction when there's only one VM hosted on it will be powered down after the last one VM migrating out. In some cases, VMAP results in no more even less active servers than C^{2}vmMap_heur. However, the computing energy it saved is less than C^{2}vmMap_heur, because it powered down servers according to the relationship of the heat generation and extraction but not the energy efficiency of servers. VMAP powers down some servers with higher energy efficiency and leaves some servers with lower energy efficiency to be active, which results in higher energy cost than powering down servers with lower energy efficiency even when its number of active servers is smaller than the latter. C^{2}vmMap_heur powers down servers concurrently according to the computing and cooling energies efficiency of servers. Thus, C^{2}vmMap_heur saves more computing energy than VMAP.

Figure 3, Fig. 4 and Fig. 5 illustrate the energy saving with the increasing of the number of PMs in various workload scenarios respectively. The energy saving increases with the increasing of the number of PMs, and the energy saving in light workload scenario is better than that in middle and heavy workload scenarios. Figure 7 illustrates the rate of migrated VMs by the two algorithms in various workload scenarios. This figure shows that the rate of migrated VMs increases with the workload decreases, and the rates of migrated VMs in light workload scenarios are larger than those in middle and heavy workload scenarios. From the

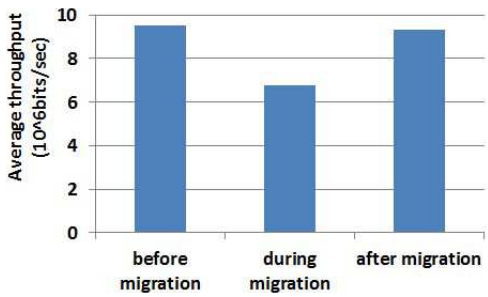


Fig. 8 The performance of the hosted applications before, during and after VM migration.

above analysis, we can conclude the degree of energy saving is related to the number of VMs which can be migrated. For the same algorithm, the degree of energy saving decreases with the increasing workload as well as with the decreasing numbers of PMs.

Comparison-II: C^{2}vmMap_heur vs. E_Saving_Huawei

We compare C^{2}vmMap_heur with E_Saving_Huawei [35] in the middle workload scenario using the similar evaluations of Comparison-I. The experimental results show that E_Saving_Huawei reduces up to 14% total energy which is only contributed by the 32.1% cooling energy saving. Compared with E_Saving_Huawei, C^{2}vmMap_heur saves up to 2.3x total energy. The goal of E_Saving_Huawei is to reduce the hotspot temperature so as to save the cooling energy. Thus, it reduces the hotspot temperature 3.5^oC which is remarkable than that of C^{2}vmMap_heur with 1.38^oC hotspot decreases. However, E_Saving_Huawei do not care the computing energy saving. It did not reduce the number of active PMs, and it did not contribute to the computing energy saving. At last, we compare the decision-making time (millisecond) of E_Saving_Huawei with C^{2}vmMap_heur. The results show that E_Saving_Huawei using 34ms to make VM migration decision, which is much shorter than that of C^{2}vmMap_heur with 126ms.

In order to evaluate the influence of VM migration on the performance of the hosted applications, we give the throughput of web service before, during and after VM migration using our algorithms in Fig. 8. The throughputs of web service before and after VM migration are close, while it decreases 29% during the process of VM migration. These results confirm that our algorithms guarantee the performance of the hosted applications after migration. The performance loss only occurs during the process of VM migration which comes from the migration scheme of Xen.

We also analyze the cost of migration and the energy cost of the management server running our algorithm. The cost of migration includes the migration time and the energy cost of a VM migration. In our experiment, the average migration time for a VM is 86 seconds, and the average migration energy cost for a VM is 0.00003 J. We obtain the average migration energy cost by the increased average power for the migration source and target servers during the migration compared with the average power for these servers before and after migration and the average migration time.

Table 4 Decision-making time (millisecond) of C^{2}vmMap_heur and VMAP for various numbers of PMs and various workload scenarios.

PM_number	workload	C^{2}vmMap_heur	VMAP
20	Heavy	102ms	22ms
	Middle	126ms	44ms
	Light	136ms	14ms
14	Heavy	49ms	12ms
	Middle	84ms	13ms
	Light	81ms	9ms
8	Heavy	18ms	11ms
	Middle	31ms	10ms
	Light	30ms	7ms

The total extra power consumed by all VM migrations determined by C^{2}vmMap_heur is up to 0.5% compared with the total saved power by this algorithm. As to the management server which runs the migration algorithm, the maximum CPU utilization is 12.5% which is lower than the minimum average CPU utilization of the 20 managed servers when it runs our migration algorithm. The power consumed by the management server is up to 1.66% compared with the total saved power by this algorithm. With the increasing of the number of managed servers, the total saved power by this algorithm will increase remarkably, while the power consumed by the management server will increase slightly. Thus, with the increasing of the number of managed servers, the rate of power consumed by the management server and the total saved power by this algorithm will decrease.

5.2.2 Evaluation 2: Scalability

We analyze the scalability of C^{2}vmMap_heur. First, we compare the decision-making time of C^{2}vmMap_heur and VMAP by adjusting the scale of real server pool using 8, 14, and 20 servers respectively. Then we evaluate the power saving and decision-making time of C^{2}vmMap_heur in the scale of 100, 200, 300, 400 and 500 PMs using the developed simulator.

Table 4 illustrates the decision-making time of the two algorithms for 20 PMs, 14 PMs, and 8 PMs in heavy, middle, and light workload scenarios. The decision-making time of C^{2}vmMap_heur is longer than that of VMAP (from 1.6x to 9x). With the increasing of the number of PMs, the decision-making time of the two algorithms increases. However, with the increase of the number of VMs in each scale of PMs, the decision-making time of the two algorithms does not show evident regulations. The reason is that it is the VM number hosted by the source PMs of migrations not the total VM number which influences the decision-making time of C^{2}vmMap_heur and VMAP. The decision-making time of VMAP outperforms C^{2}vmMap_heur in all scenarios. It selects the PM with the minimum difference between its heat generation and its heat extraction to be the target PM for a VM, while does not need to makes decisions for each VM migrated to each PM.

Figure 9 and Fig. 10 illustrate the power saving and

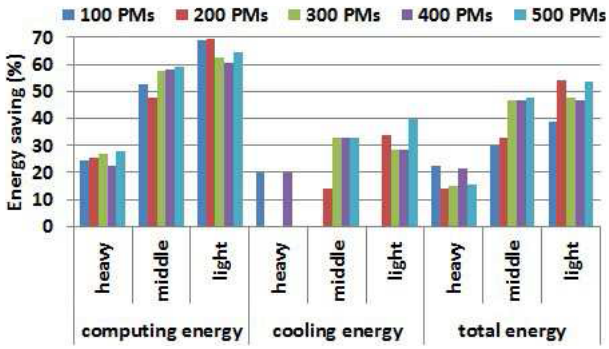


Fig. 9 Energy saving provided by C^{2}vmMap_heur in various PM scales and under various workload scenarios using simulator.

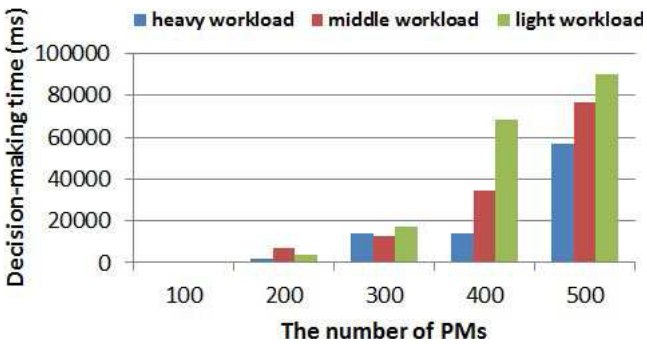


Fig. 10 Decision-making time of C^{2}vmMap_heur in various PM scales and under various workload scenarios using simulator.

decision-making time provided by C^{2}vmMap_heur in the scale of 100, 200, 300, 400 and 500 PMs under various workload scenarios using simulator. From Fig. 9, we can see that the energy savings are close in various scales under the same workload scenario, and the energy saved by C^{2}vmMap_heur is most remarkable in the light workload scenario which is consistent with the results of the real server experiments. C^{2}vmMap_heur reduces 70%, 22% and 54% computing energy, cooling energy and total energy at most than the non-migration load balance algorithm respectively. Figure 10 illustrates with the increasing of the scale of PMs, the decision-making time increases. The decision-making uses 90 seconds when the number of PMs scales up to 500. In lighter workload scenario, C^{2}vmMap_heur takes longer time to make VM migration decisions in the same PM scale, because more target selections resulted from more idle resources of each servers for the migration of each VM. The decision-making time of 90 seconds is acceptable in current data centers, because it is close to the average time of one VM migration (86 seconds in our test) and the execution period of such decision-making is much larger than 90 seconds.

5.3 Discussion about the Experiments

The VM migrations determined by the migration algorithm will not influence the energy of the management server which running our algorithm and the storage server which

stores the VM image files. Thus, in the above evaluation, we only take the energy of the 20 managed servers which host VMs in the server pool into account, ignoring the energy of the management server and the storage server. In our test environment, all the VM image files are stored in the storage server. The disk reading and writing by each application in a VM do not within the server which hosts the VM, and they are executed in the storage server. Thus, the disk resource utilization of each VM is set to be zero when computing the temperature contribution of each VM in our experiments. Of course, C^2vmMap model and C^2vmMap_heur algorithm are also applicable to other storage modes. If the VM image file is stored in the server which hosts the VM, and it migrates with the migration of the VM, the disk resource utilization of each VM is obtained by the monitor software when computing the temperature contribution of each VM. If the VM image file does not migrate with the VM, and it is stored in any server of the server pool based on distributed file systems, e.g. HDFS (Hadoop Distributed File System), the disk resource utilization of each VM is set to be zero when computing the temperature contribution of each VM, because VM migration will not influence the disk temperatures of the source and target servers.

6. Conclusion

This work proposes C^2vmMap model and C^2vmMap_heur algorithm to minimize the total energy consumption. The experimental results show that C^2vmMap_heur reduces more than 40% energy. C^2vmMap_heur migrates VM according to the workload of each VM and temperature of each PM, the relevancy of workloads of various VMs does not affect the decision of the algorithm. Thus, as to the applications that communicates over multiple VMs (MPI, MapReduce, etc.), C^2vmMap_heur can handle them. If the workloads of the hosted applications fluctuate drastically within short time period, our algorithm may give rise to frequent VM migrations to adjust the workload distribution. Thus, as to the applications with stable workload, our algorithm may get better results. The accuracy of the prediction functions for the outlet temperatures of servers in Assumption 6 influences the results of energy saving by our algorithm. The prediction method will be refined in the future to improve the prediction accuracy. For example, we will take the air velocity not only the speed of fans into account, and we will design an automatic learning method to on-line update the coefficients of the prediction functions.

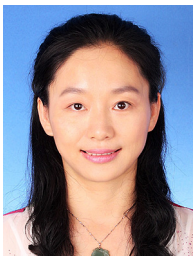
Acknowledgments

This work was supported in part by the fund of the Beijing Key Laboratory of Big Data Technology for Food Safety, Beijing Technology & Business University (BTBU)(BKBD-2016KF12), the State Key Laboratory of Computer Architecture (CARCH201605), Beijing Municipal Science and Technology Project (Z171100004717002), Beijing Natural Science Foundation (Z160002), and the Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (ICDDXN004).

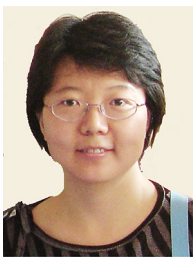
References

[1] G. Cook, How clean is your cloud, Catalysing an energy revolution, 2012.
[2] Gartner Research, Top 10 Techniques to Improve Data Center Cooling Efficiency, June 2012.
[3] A. Faraz and T.N. Vijaykumar, "Joint optimization of idle and cooling power in data centers while maintaining response time," ACM SIGARCH Computer Architecture News, vol.38, no.1, pp.243–256, 2010.
[4] J. Xu and J. Fortes, "A Multi-objective Approach to Virtual Machine Management in Datacenters," ICAC 2011.
[5] E.K. Lee, H. Viswanathan, and D. Pompili, "VMAP: Proactive Thermal-aware Virtual Machine Allocation in HPC Cloud Datacenters," HiPC 2012.
[6] K.G. Murty and S.N. Kabadi, "Some NP-complete problem in quadratic and nonlinear programming," mathematical programming, vol.39, no.2, pp.117–129, 1987.
[7] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz, "Energy efficiency for large-scale MapReduce workloads with significant interactive analysis," EuroSys 2012, pp.43–56, 2012.
[8] X. Yang, Z. Zhou, S. Wallace, Z. Lan, W. Tang, S. Coghlan, and M.E. Papka, "Integrating dynamic pricing of electricity into energy aware scheduling for HPC systems," SC 2013.
[9] S. Ren, Y. He, and F. Xu, "Provably-Efficient Job Scheduling for Energy and Fairness in Geographically Distributed Data Centers," ICDCS 2012, pp.22–31, 2012.
[10] J. Choi, S. Govindan, B. Urgaonkar, and A. Sivasubramaniam, "Profiling, prediction, and capping of power consumption in consolidated environments," In Proceedings of IEEE Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2008.
[11] F. Hermenier, X. Lorca, J.-M. Menaud, G. Muller, and J. Lawall, "Entropy: a consolidation manager for clusters," VEE 2009. pp.41–50, 2009.
[12] X.-F. Liu, Z.-H. Zhan, K.-J. Du, and W.-N. Chen, "Energy Aware Virtual Machine Placement Scheduling in Cloud Computing Based on Ant Colony Optimization Approach," GECCO 2014, pp.41–47, 2014.
[13] A. Verma, P. Ahuja, and A. Neogi, "pMapper: power and migration cost aware application placement in virtualized systems," In Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware, vol.5346, pp.243–264, 2008.
[14] B. Speitkamp and M. Bichler, "A Mathematical Programming Approach for Server Consolidation Problems in Virtualized Data Centers," IEEE Trans. on Services Computing, vol.3, no.4, pp.266–278, 2010.
[15] J. Kim, M. Ruggiero, D. Atienza, and M. Lederberger, "Correlation-aware virtual machine allocation for energy-efficient datacenters," DATE 2013, pp.1345–1350, 2013.
[16] G. Jung, M.A. Hiltunen, K.R. Joshi, R.D. Schlichting, and C. Pu, "Mistral: Dynamically Managing Power, Performance, and Adaptation Cost in Cloud Infrastructures," ICDCS 2010, pp.62–73, 2010.
[17] D. Panagiotou, E. Oikonomou, A. Rouskas, "Energy-efficient Virtual Machine Provisioning Mechanism in Cloud Computing Environments," PCI 2015, pp.197–202, 2015.
[18] Q. Tang, S.-S. Gupta, D. Stanzione, and P. Cayton, "Thermal-Aware Task Scheduling to Minimize Energy Usage of Blade Server Based Datacenters," Proc. Second IEEE Int',l Symp. Dependable, Autonomic and Secure Computing, pp.195–202, 2006.
[19] J. Moore, J. Chase, P. Ranganathan, and R. Sharma, "Making scheduling cool: Temperature-aware resource assignment in data centers," Proc. Usenix Ann. Technical Conf., April 2005.

[20] M. Mase, J. Okitsu, E. Suzuki, T. Nojiri, K. Sano, and H. Shimizu, "Cooling efficiency aware workload placement using historical sensor data on IT-facility collaborative control," 2012 International Green Computing Conference, pp.1–6, 2012.
[21] A. Banerjee, T. Mukherjee, G. Varsamopoulos, and S.K.S. Gupta, "Cooling-aware and thermal-aware workload placement for green HPC data centers," Proc. of the First International Green Computing Conference (IGCC'10), pp.245–256, 2010.
[22] Q. Tang, S.K.S. Gupta, and G. Varsamopoulos, "Thermal-Aware Task Scheduling for Data Centers through Minimizing Heat Recirculation," Proc. IEEE Cluster, pp.129–138, Sept. 2007.
[23] L. Wang, S.U. Khan, and J. Dayal, "Thermal aware workload placement with task-temperature profiles in a data center," J Supercomput., vol.61, no.3, pp.780–803, 2012. doi: 10.1007/s11227-011-0635-z
[24] I. Manousakis, Í. Goiri, S. Sankar, T.D. Nguyen, and R. Bianchini, "CoolProvision: Underprovisioning Datacenter Cooling," SoCC'15, pp.356–367, Aug. 27–29, 2015.
[25] S. Yeo, M.M. Hossain, J.-C. Huang, and H.-H.S. Lee, "ATAC: Ambient Temperature-Aware Capping for Power Efficient Datacenters," SoCC'14, 3–5 Nov. 2014.
[26] A.M. Al-Qawasmeh, S. Pasricha, A.M. Maciejewski, and H.J. Siegel, "Thermal-Aware Performance Optimization in Power Constrained Heterogeneous Data Centers," IEEE IPDPSW, pp.27–40, 2012.
[27] H.F. Sheikh, I. Ahmad, D. Fan, "An Evolutionary Technique for Performance-Energy-Temperature Optimized Scheduling of Parallel Tasks on Multi-Core Processors," IEEE Trans. Parallel Distrib. Syst., vol.27, no.3, pp.668–681, March 2016.
[28] A.M. Al-Qawasmeh, S. Pasricha, A.A. Maciejewski, and H.J. Siegel, "Power and Thermal-Aware Workload Allocation in Heterogeneous Data Centers," IEEE Trans. Comput., vol.64, no.2, pp.477–491, Feb. 2015.
[29] Z. Abbasi, G. Varsamopoulos, and S.K.S. Gupta, "TACOMA: Server and workload management in Internet data centers considering cooling-computing power trade-off and energy proportionality," ACM Transactions on Architecture and Code Optimization, vol.9, no.2, pp.1–37, 2012.
[30] J. Yao, H. Guan, J. Luo, L. Rao, and X. Liu, "Adaptive Power Management through Thermal Aware Workload Balancing in Internet Data Centers," IEEE Trans. Parallel Distrib. Syst., vol.26, no.9, pp.2400–2409, Sept. 2015.
[31] X. Li, X. Jiang, and Y. He, "Virtual Machine Scheduling Considering Both Computing and Cooling Energy," 2014 IEEE Intl Conf on Embedded Software and Syst, pp.244–247, 2014.
[32] R. Buyya, et al., "Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges," PDPTA 2010, 2010.
[33] A. Alssaiari, R.A.J. Gining, and N. Thomas, "Modelling Energy Efficient Server Management Policies in PEPA," ICPE'17 Companion, Italy, pp.43–48, April 22–26, 2017.
[34] T.-C. Tang and Y.-S. Chen, "Thermal-aware MapReduce Real-Time Scheduling in Heterogeneous Server Systems," RACS'16, Denmark, pp.207–212, Oct. 11–14, 2016.
[35] Y. Song, Y. Sun, X. Zhao, and K. Kang, Method, Apparatus, and System for Migrating Virtual Machine, Application Number: 14/725057, public date:2015/12/03, Huawei Technologies Co., Ltd.
[36] DeepMind, https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/.



Ying Song received the Ph.D. degree in computer engineering from the Institute of Computing Technology (ICT), Chinese Academy of Sciences. She is an associate professor at the Computer School, Beijing Information Science & Technology University. Her work thus far has covered topics such as performance modeling, resource management, cloud computing, and big data computing platform. Her main research interests include computer architecture, parallel and distributed computing, and virtualization technology. She has authored or coauthored more than 30 publications in these areas since 2007, and she served in various academic conferences. She is a member of the IEICE.



Xia Zhao received Ph.D. degree in computer science and technology from the Peking University. She is an associate professor and the Chair of the Department of Computer Science and Technology at Beijing Technology and Business University. Her research interests focus on system software and big data processing. She has authored and co-authored more than 20 publications, and served in various academic conferences and journals. She is a member of the IEEE, ACM and CCF.

Bo Wang received the Ph.D. degree in computer science at Xi'an Jiaotong University (XJTU). He is an associate professor at Software Engineering College, Zhengzhou University of Light Industry. His research interests include distributed systems, cloud computing, resource management and big data computing platform.

Yuzhong Sun received the Ph.D. degree in computer engineering from the Institute of Computing Technology (ICT), Chinese Academy of Sciences. He is a professor in the State Key Laboratory of Computer Architecture at ICT in Beijing, China. His research interests focus on distributed system software and computing/programming models. He has authored and coauthored more than 50 publications, and he has served in various academic conferences and journals. He is a member of the IEEE and the IEEE Computer Society.