An Adaptive Energy-Aware Stochastic Task Execution Algorithm in Virtualized Networked Datacenters

Zhou Zhou, Kenli Li, *Senior Member, IEEE*, Jemal Abawajy, *Senior Member, IEEE*, Mohammad Shojafar, *Senior Member, IEEE*, Morshed Chowdhury, Fangmin Li, and Keqin Li, *Fellow, IEEE*

Abstract—Virtualized networked datacenters (VNDCs) are gaining considerable attention for stochastic task execution under real-time constraints. However, the problem of efficiently minimizing the high energy consumption while ensuring high quality of service (QoS) in VNDCs has not been fully addressed. Although many solutions have been proposed to address this challenge, they are not efficient and only consider one or two of the energy consuming resources of VNDCs. To this end, an adaptive energy-aware algorithm, *MCEC*, that efficiently reduces the energy consumption of VNDCs while ensuring high QoS is proposed. Different from the existing approaches, the MCEC algorithm considers energy consumed by computing resources, virtual machine (VM) reconfiguration, communication resources and storage media resources while meeting user QoS requirements defined in the service level agreement (SLA). To validate the effectiveness of our algorithm, we carried out extensive experiments and compared the performance of our algorithm with existing baseline algorithms. The results of the experiments show that our algorithm substantially outperforms the baseline algorithms with respect to reducing energy consumption while respecting the service level agreement.

Index Terms—Dynamic resource provisioning, energy-aware algorithm, optimization, resource management, virtualized networked datacenter (VNDC)

1 Introduction

THE increasing demand for computational power driven by modern large-scale computation-intensive applications and big data processing have led to the establishment of large-scale virtualized networked datacenters (VNDCs). VNDCs mean network virtualization in the data center. Network virtualization can bring advantages such as better utilization of network resources, isolation of specific protocols to a dedicated network segment for the sake of security, and port aggregation. Although VNDCs have recently gained wide attention among researchers and industry for processing a wide variety of demanding applications [1], [2], the enormous amounts of energy consumed by VNDCs have increasingly become a serious concern [3], [4]. It has been estimated that datacenters consumed in excess of 4.35 gigawatts of electricity in 2013, with a reported increase of up to 15% each year [5], [6], [7].

The main focus of this paper is on energy consumption and performance management strategies that can be applied to VNDCs. Energy consumption in particular is a major concern for VNDC providers[8], [9]. Therefore, we address the problem of how to decrease energy consumption in VNDCs with minimal impact on the delivery of quality of service (QoS) requirements to VNDC users. QoS is generally defined in terms of SLA [10], [11], [12], [13], [14], [15]. High energy consumption in datacenters can have serious consequences, such as system instability, increased operating costs of enterprises [16], [17], lower returns on investment [18], and increased carbon dioxide emissions [18], [19]. Therefore, it is imperative to develop approaches that can reduce energy consumption rates to make datacenters environmentally sustainable and economically profitable.

Energy-aware algorithms are some of the effective methods that deal with the problem of high energy consumption

- *Zhou Zhou is with the College of Information Science and Engineering, Hunan University, Changsha, Hunan 410082, China, and also with the Department of Computer Engineering and Applied Mathematics, Changsha University, Changsha, Hunan 410004, China. E-mail: zhouzhou03201@126.com.*
- *Kenli Li is with the College of Information Science and Engineering, Hunan University, Changsha, Hunan 410082, China. E-mail: lkl@hnu.edu.cn.*
- *Jemal Abawajy and Morshed Chowdhury are with the School of Information Technology, Deakin University, Waurn Ponds, Geelong, VIC 3216, Australia. E-mail: {jemal.abawajy, morshed.chowdhury}@deakin.edu.au.*
- *Mohammad Shojafar is with the GIC & 6GIC, Institute for Communication Systems (ICS), University of Surrey, Guildford GU2 7XH, U.K. E-mail: m.shojafar@surrey.ac.uk.*
- *Fangmin Li is with the Department of Mathematics and Computer Science, Changsha University, Changsha, Hunan 410004, China. E-mail: lifangmin@whut.edu.cn.*
- *Keqin Li is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan 410082, China, and also with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA. E-mail: lik@newpaltz.edu.*

Manuscript received 4 Oct. 2020; revised 2 Aug. 2021; accepted 20 Sept. 2021. Date of publication 24 Sept. 2021; date of current version 8 June 2022.
This work was supported in part by the National Key Research and development Program (High performance computing) under Grant Z202003420758, in part by the National Natural Science Foundation of China under Grant 61772088, and in part by the Hunan Province Key Laboratory of Industrial Internet Technology and Security under Grant 2019TP1011.
(Corresponding author: Kenli Li.)
Recommended for acceptance by A. Zomaya.
Digital Object Identifier no. 10.1109/TSUSC.2021.3115388

in VNDCs [20], [21]. Even though physical infrastructure advances have partially reduced the energy consumption issues of datacenters, the problem of high energy consumption in VNDCs is not just due to the power inefficiency of the hardware or to the usage of VNDC resources but is mainly due to the inefficient usage of VNDC resources [22], [23], [24], [25]. Therefore, effective resource management is vital for further reducing the energy consumption of VNDCs.

However, efficient VNDC resource management is a nontrivial task. Applications targeting VNDCs often experience highly variable workloads, which leads to dynamic resource usage patterns. This necessitates the development of an adaptive energy-aware algorithm that can maximize the energy reduction while meeting the QoS requirements for users. However, the responsibility of providing high QoS to VNDC users leads to obligations in addressing the energy-performance trade-off issues. However, since VNDCs are dynamic environments, guaranteeing QoS while simultaneously minimizing energy consumption is a challenging problem. This is why almost all existing approaches focus on either computational resources [26], [27] or communication resources [22] and rarely on both computational and communication resources [28], [29]. Moreover, existing approaches are not efficient and only consider one or two energy consuming components of VNDCs. Specifically, we respond to the following questions: *i)* Is it possible to design an adaptive energy-aware algorithm that offer a fair SLA for VNDCs environment? (see Section 4.2) *ii)* How can such algorithm preserve QoS constraints when processing tasks? (see Section 4), and *iii)* How can we solve the resource allocation optimization problem linearly? (see Section 4.3).

1.1 The Goal of the Paper and Contributions

The goal of our paper is to mathematically model the resource allocation in VNDC and propose a heuristic algorithm to solve it. To accomplish this, we propose an adaptive energy-aware approach that efficiently decreases energy consumption of VNDCs while respecting the SLA. The proposed approach integrates both virtualization and a DVFS (dynamic voltage and frequency scaling) methods to manage the consumption of energy and reduce SLA violation rates. DVFS adjusts the CPU frequency and voltages based on outside requirements from tasks to minimize energy consumption. Our main contributions can be summarised as:

1) Introduction of stochastic service system theory (queuing theory) to model the workload in VNDCs under real-time constraints.
2) Proposal of the MCEC algorithm that considers not only the computing-plus-communication energy consumption, but also VM reconfiguration and disk energy consumption.
3) Use of a mathematical method to change the non-convexity of communication energy consumption to convexity, proving that the communication energy consumption is a convex problem.
4) Evaluation and verification of the effectiveness of the proposed algorithm using Matlab software.

The remaining sections are organized as follows. The related works are described in Section 2. The background can be found in Section 3. Section 4 puts forward the optimization problem and the solution we proposed in this paper. The experimental evaluation, the conclusion and future works are discussed in Sections 5 and 6 respectively.

2 Related Work

With VNDCs consuming high energy, the challenge of how to minimise VNDCs energy consumption has become a hot topic that has been receiving an increasing attention among researchers. To address this challenge, many researchers have proposed methods such as energy-efficient modelling (C_1) [30], [31], [32], DVFS (C_2) [33], [34], [35], virtualization technology (C_3) [36], [37], [38], [39], [40], and a combination of both DVFS and virtualization technology (C_4) [26], [27], [28], [29], [41], [42] to reduce energy consumption and ensure the high QoS requirement from the cloud system.

Energy-efficient modelling [30], [31], [32], [43] is indispensable and important for any energy-aware algorithm, as it lays a foundation for energy-saving algorithms. At present, there are two types of methods for energy-efficient modelling: (1) The first class [30], [31] models the energy consumption based on resource utilization. For example, Garg *et al.* [30], put forward a linear model that leverages the server CPU utilization to develop a model. In contrast with the linear model, Zhang *et al.* [31] , based on the assumption that there is a cubic relationship between the consumption of energy and CPU utilization of a server, proposed a cubic model. This type of method is direct and is easily used to construct a power model. However, it is not accurate to build a power model considering only CPU or memory factors. (2) The second class models the energy consumption based on performance counters [32]. This kind of method collects all kinds of events related to energy consumption to model energy consumption. Although this method is promising, it is too complicated to extend to other datacenters.

Server energy consumption has a close relationship to its processor frequency. Therefore, DVFS [33], [34], [35] is widely used to to decrease the consumption of energy by adjusting the processor frequency. For instance, when the number of tasks processed on the server increases, DVFS raises frequency of the processor to guarantee the QoS demand. Conversely, if the number of task is less, DVFS reduces the frequency of the processor to achieve energy saving. Specifically, Herbert and Marculescu [33] applied DVFS technology to process multi-threaded commercial and scientific workloads, thus illustrating the potential of DVFS for chip multiprocessors. Based on the success of DVFS, Hanson *et al.* [34] considered it a viable option to manage the thermal variations of the processor. Experimental results from the paper confirm that DVFS has a direct influence on processor temperature and could be utilized for thermal management. To improve the resource utilization and efficiency of hardware infrastructure, Wu *et al.* [35] leveraged DVFS to manage the supply voltage and frequency for hosts in a datacenter, thus reducing the energy consumption of the hosts. The method proposed in this paper is effective, especially when a host has a light workload or is in idle mode. Although DVFS is promising, its effect on energy saving needs to be further improved.

TABLE 1
Comparison Between Different Algorithms (Where a Tick Indicates That the Method Supports the Property, and a Cross Indicates That the Method Does not Support the Property)

Ref.	Contribute to Save Power	Energy-Aware	SLA Aware	Energy Efficiency
C_1				
[30]	✓	×	×	×
[31]	✓	×	×	×
[32]	✓	×	×	×
C_2				
[33]	×	✓	×	×
[34]	×	✓	×	×
[35]	×	✓	✓	✓
C_3				
[36]	✓	×	×	×
[37]	×	✓	✓	✓
[38]	×	✓	✓	✓
[39]	×	✓	✓	✓
[40]	×	✓	✓	✓
C_4				
[26]	×	✓	×	✓
[27]	×	✓	×	×
[28]	×	✓	✓	✓
[29]	×	✓	✓	✓
[41]	×	✓	✓	✓
[42]	×	✓	✓	✓
MCCD	×	✓	✓	✓

Virtualization technology [36], [37], [38], [39], [40] provides a novel strategy to reduce energy consumption in VNDCs. Through the use of virtualization technology, workloads located on different servers can be merged to improve the resource utilization and reduce operation costs. Specifically, authors in [36] outlined frameworks that leveraged a VM migration technique and a server consolidation method for efficient resource management. Additionally, the authors believe that exploiting the virtualization can achieve power management, load balancing, fault tolerance and system maintenance. To achieve energy saving, Kumar and Raghunathan [37] put forward a VM placement and consolidation method to address a datacenter energy consumption. To further reduce data centers energy consumption , Beloglazov *et al.* [38] first explored the issue of the VM migration and then proposed several adaptive heuristics to address the energy-performance trade-off. Although the experimental results show the effectiveness in reducing the energy consumption, the authors did not consider the VM reconfiguration, communication, and disk energy consumption. In their later work [39], Beloglazov and Buyya solved the problem of host overloaded detection under specified SLA violation requirements. Additionally, the heuristic proposed in the paper can handle known or unknown workloads using the MSW (multi-size sliding window) estimation technique. To improve energy efficiency, in our previous study [40], based on the linear relationship between processor resource utilization and energy consumption, we put forward five algorithms to reduce energy consumption while maintaining the QoS. The drawback of this method is that it is not suitable for varying workloads.

Combining both DVFS and virtualization technology [26], [27], [28], [29], [41], [42] is an effective approach to lower the energy consumption in VNDCs. Specifically, Urgaonkar *et al.* [26] put forward a strategy called Lyapunov to jointly maximize the energy costs utility and average throughput of an application. The drawback of this method is that it did not consider VM reconfiguration, communication and disk energy consumption. Next, Mathew *et al.* [41] used the IDEAL method to obtain the optimal frequency for the incoming workload for the purpose of saving energy. IDEAL is an idealized algorithm, and it does not highlight VM reconfiguration and disk energy consumption. To decrease energy consumption, Kimura *et al.* [27] proposed a method named Standard to save energy consumption. However, it considers only computational energy consumption. Next, Shojafar *et al.* [42] explored the problem of energy-performance trade-off. Although this approach is promising, it assumes that tasks arrive in each time slot; thus, there is consideration of possibly queued tasks. However, in most cases, the workload contains randomness. In addition, the original paper did not take the energy consumption of the disk into account. Finally, the work in [28], [29] is focused on reducing the computing plus communication energy consumption in VNDCs but did not take into account the energy consumption due to the server disks. Table 1 summarizes these categories.

3 Background

The system model and the related background knowledge are discussed in this section. The main symbols used throughout the paper and their meanings are shown in Table 2.

3.1 System Model

A high level architecture of a VNDC system is shown in Fig. 1. The system is made up of three layers namely: (i) a user layer; (ii) a middleware layer; and (iii) a physical server layer [28], [29], [44]. The system consists of n DVFS-enabled virtualized server nodes $SC = \{S_1, S_2, \ldots, S_n\}$. A single-hop virtual network under a central controller is used to interconnect the server nodes. Each server node $S_i \in SC$ is characterized by a virtual network interface card (VNIC), CPU, amount of RAM, voltage frequency, network bandwidth and disk storage. Each server can work with different CPU frequencies by adjusting the voltage frequency using the DVFS capabilities of the underlying hardware. As in [28] and [29], we assume that each server node $S_i \in SC$ manages local virtualized resources (i.e., computing, network and storage) as well as processes the tasks assigned to it.

End users submit requests to the admission control for VNDC resource provisioning. The admission control subsystem is used to prevent overloading of resources such that SLA satisfaction is not impacted. The user requests specify the CPU performance requirements, RAM, network bandwidth and disk storage. The admission control uses the information specified in the user requests as well as data from the SLA subsystem to admit or reject incoming service requests. Interested readers can refer to the work in [45], which we have used in this paper.

All admitted requests will be maintained in a queue until they are assigned the requested resources. Note that each incoming service request has to be processed within a

TABLE 2
The Symbols and Their Associated Meanings

Symbol	Meaning
E_{total}	Total energy consumption
m	The number of VMs
E_{cpu}	Computational energy consumption
E_{reconf}	VM reconfiguration energy consumption
E_{com}	Communication energy consumption
E_{disk}	Storage energy consumption
P	Dynamic power consumption of a CPU
f	Processing frequency of a CPU
V	Supply voltage of a CPU
$f_k(i)$	The kth discrete frequency of the VM i
$t_k(i)$	Time that VM i operates at frequency $f_k(i)$
Q	Number of the discrete CPU frequencies
$E_{reconf}(Ext)(i)$	VM i reconfiguration external energy consumption
$E_{reconf}(Int)(i)$	The VMs i reconfiguration internal energy consumption
$p_{net}(i)$	The communication energy consumption
k_c	Reconfiguration energy consumption coefficient
η	Gaussian noise
$R(i)$	Communication rate of the ith end to end connection
$B(i)$	Refers to the transmission bandwidth
P_{idle}	Idle power consumption at end to end link connection
S	Task size
b	A parameter used to model the offered workload under combination of parameter S
$t(w)$	Task queue time
$t(e)$	Execution time of a task
T	Maximum allowed waiting-plus-computing time per-job
T_{total}	Job deadline
R_{total}	Aggregate communication rate of VLAN
λ	The average arrival rate per unit time for user jobs
μ	The average service rate per unit time for jobs
E_{CC}	Sum of computation plus communication energy consumption
E_{CRC}	Sum of computation, external VMs configuration, and communication energy consumption
E_{CRCD}	Sum of computation, external VMs configuration, communication and disk energy consumption

deadline time (T_{total}) defined in the SLA. In the paper, we divide the SLA into two aspects: first, the total time (including the queuing time, computation time, and network-related time) for per job should be less than or equal to (T_{total}), second, the waiting-plus-computation time for per job should be less than or equal to T. The scheduler performs the allocation of the available virtual resources to the user requests. The dispatcher dispatches the VMs to the allocated host for processing. The virtual machine manager (VMM) is responsible for handling the operations of both the VMs and the VLAN. We assume that VNDC contains k reconfigurable virtual machines, $VM = \{VM_1, \ldots, VM_{k-1}, VM_k\}$. These VMs are interconnected by a switched VLAN (virtual local area network) with limited throughput and star-like topology [12]. A central node with VNS (virtual network switch) serves as a gather/scatter and guarantees internal communication between the VMs. Additionally, the VMM dynamically allocates the proper resources, such as computational resources, communication resources, and disk resources, to the VMs for the purpose of processing the incoming workload.

In the proposed approach, we consider the consumption of energy by the CPU, network, disk storage and VM reconfiguration. Given a VM i, the overall energy expended (E_{total}) by the various system components of VM i is derived as

$$E_{total} = \sum_{i=1}^{m} \Big(E_{cpu}(i) + E_{reconf}(i) + E_{com}(i) + E_{disk}(i) \Big), \tag{1}$$

where $E_{cpu}(i)$, $E_{reconf}(i)$, $E_{com}(i)$, and $E_{disk}(i)$ represent the energy consumption by the CPU, VM reconfiguration, communication, and disk, respectively. We will discuss how the energy consumption by various system components of VM i are derived in the following subsections.

3.1.1 Computation Energy Consumption

The CPU dynamic power consumption (P) is mainly based on the processor frequency and the supply voltage. Therefore, we can define (P) as [46]

$$P = A \cdot C_{eff} \cdot f \cdot V^2, \tag{2}$$

where A represents the active percentage of the gates, C_{eff} is the effective capacitance load, f is the processing frequency, and V is the CPU supply voltage of the CPU.

Since the relationship between P and f is cubic [47], we can define the energy consumption of the VM i due to computation ($E_{cpu}(i)$) as

$$E_{cpu}(i) = \sum_{k=0}^{Q} AC_{eff} \Big(f_k(i) \Big)^3 t_k(i) \quad \forall i = \{1, 2, \ldots, m\}, \tag{3}$$

where Q denotes the maximum number of CPU frequencies of each VM and VM i is said to be in an idle state when k = 0. The parameter $f_k(i)$ represents the kth processing frequency of VM i, and the parameter $t_k(i)$ represents the instance at which VM i operates at $f_k(i)$ frequency. Fig. 2 shows an example of VM i operating at different frequencies when Q = 5.

3.1.2 VM Reconfiguration Energy Consumption

When a VNDC processes a workload under real-time constraints, it is necessary for the VMs to be able to increase/decrease their processing rate. For this reason, changing from a given $f_k(i)$ processing frequency to another $f_{k+1}(i)$ frequency entails VM reconfiguration costs.

For a given VM i, the energy consumption due to VM reconfiguration, ($E_{reconf}(i)$), has two main parts: an external energy consumption due to VM reconfiguration ($E_{reconf}(Ext)(i)$) and an internal energy consumption due to the VM reconfiguration ($E_{reconf}(Int)(i)$), which satisfy

$$E_{reconf}(i) = E_{reconf}(Ext)(i) + E_{reconf}(Int)(i), \tag{4}$$

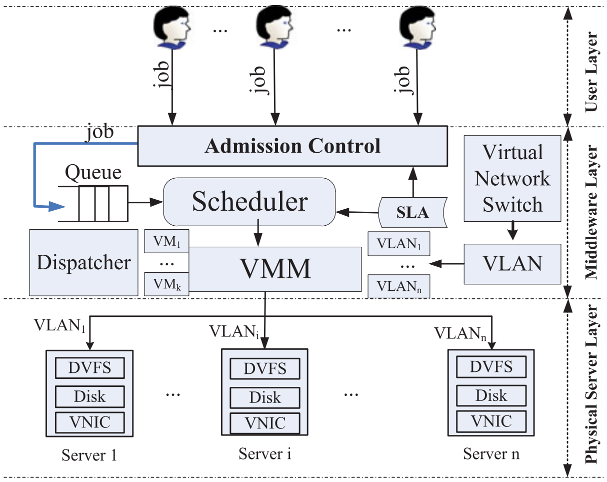


Fig. 1. VNDC architecture for stochastic tasks.

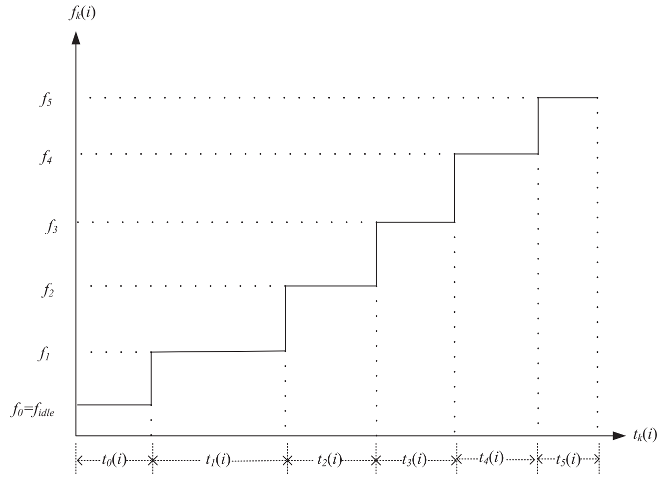


Fig. 2. The range of frequencies for the considered VM *i*.

where $(E_{reconf}(Ext)(i))$ represents the reconfiguration energy consumption and defined as

$$E_{reconf}(Ext)(i) = k_c \times E_{exter}, \tag{5}$$

where parameter k_c is the coefficient of reconfiguration energy consumption and E_{exter} represents the reconfiguration energy consumption of VM i from the final active frequency for the last job processing to the initial active frequency for the incoming job processing. In VNDCs, the value of k_c can be estimated to be a few hundreds of uJ per $[MHZ]^2$ [41].

The internal energy consumption due to VM reconfiguration, $(E_{reconf}(Int)(i))$, refers to the energy consumption by VM i when it switches from the present active distinct frequency to another frequency, so as to process jobs/tasks. For example, when VM i switches from the current active discrete frequency $f_k(i)$ to another frequency $f_{k+1}(i)$, the VM internal reconfiguration energy consumption is $[k_c(f_{k+1}(i) - f_k(i))^2]$. Therefore, $(E_{reconf}(Int)(i))$ can be indicated as

$$E_{reconf}(Int)(i) = k_c \times \sum_{k=0}^{Q} \Big(f_{k+1}(i) - f_k(i)\Big)^2. \tag{6}$$

3.1.3 Energy Consumption Due to Communication

As shown in Fig. 1, the VMs and the VMM communicate through a dedicated virtual link. Therefore, the consumption of power due to one-direction transmission plus switching operations, $(p_{net}(i))$, can be given as

$$p_{net}(i) = p^T_{net}(i) + p^R_{net}(i), \tag{7}$$

where $p^T_{net}(i)$ is the power consumption of transmitting plus switching, and $p^R_{net}(i)$ represents the consumption of energy due to the receiving operation. In general, the power consumption of the channel is the same for transmitting and receiving and can be calculated using the exponential formula of the Shannon-Hartley as

$$p_{net}(R(i)) = \eta\Big(2^{\frac{R(i)}{B(i)}} - 1\Big) + p_{idle}, \tag{8}$$

where parameter η is Gaussian noise and can be considered as a common value, parameter $R(i)$ is the transmission rate, parameter $B(i)$ refers to the transmission bandwidth, and p_{idle} represents the idle power consumption due to end-to-end link connection and can be considered a fixed value. The transmission delay, $(TD(i))$, of the ith link is defined as

$$TD(i) = \sum_{k=1}^{Q} \left(\frac{f_k(i) t_k(i)}{R(i)}\right). \tag{9}$$

Therefore, the one-way communication energy consumption, $(E_{com}(i))$, can be computed as

$$\begin{aligned} E_{com}(i) &= p_{net}(R(i)) \times TD(i) \\ &= p_{net}\Big(R(i)\Big)\left(\sum_{k=1}^{Q}\left(\frac{f_k(i)t_k(i)}{R(i)}\right)\right). \end{aligned} \tag{10}$$

3.1.4 Disk Energy Consumption

Through a storage as a service provided by the cloud service providers, end users store data on the cloud datacentre [48]. For viewing and editing their data, end users have to download the data to his/her system from the cloud and upload to the cloud datacentre any modified files. The overall consumption of power due to the storage service $(E_{disk}(i))$ [49] can be calculated as

$$E_{disk}(i) = \frac{W}{3600}\left(E_T + 1.5\frac{P_{CS}}{C_{CS}}\right) + \frac{2W}{D} \times \frac{P_{hd}}{C_{hd}}, \tag{11}$$

where C_{CS} and P_{CS} denote the capacity and the consumption of energy by the server, respectively. The parameters C_{hd} and P_{hd} denote the capacity and the consumption of energy by the hard disk arrays, respectively. W refers to the downloaded file size, E_T refers to the per bit depletion of energy due to the transmission and switching, and D refers to the number of downloads per hour and can be acquired empirically. The factor of 1.5 signifies the power requirement for cooling and other overheads while the factor of 2 in Eq. (11) is the power requirements for redundancy in storage.

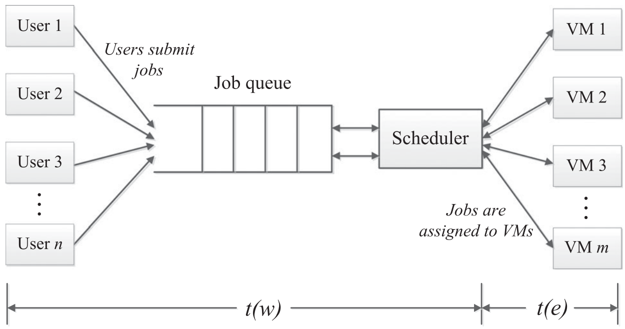


Fig. 3. Queuing model of user tasks in VNDC.

3.2 Workload Model

We consider CPU-intensive jobs with related input data used in [50]. Each accepted job consists of m tasks $\tau = \{t_1, t_2, \ldots, t_m\}$ that will execute in parallel. The m tasks of the job will be allocated to m VMs to execute on a given host. Each task, t_i, is defined by a tuple

$$t_i = \left\langle t(id), S, t(w), t(e), t(dz), t(d) \right\rangle, \tag{12}$$

where $t(id)$ is the unique identification number of the task, S is the task size, $t(w)$ is the time that the task spent in the queue waiting for processing by the VMs, $t(dz)$ is the data size used by the task, $t(d)$ is the task deadline, and $t(e)$ is the task execution time, defined as

$$t(e) = \frac{S}{f}, \tag{13}$$

where parameter f refers to the processing rate of the VM. The completion time of a task (T) is given as

$$T = t(w) + T(e). \tag{14}$$

To evaluate the parallel execution performance, we use the speedup ratio (SR) defined as

$$SR = \frac{E(T_l)}{E(T_p)}, \tag{15}$$

where parameter $E(T_l)$ corresponds to the execution of the tasks on a single VM and $E(T_p)$ is the execution of the tasks on p VMs in parallel. Each admitted job must be completed within an interval $T_{total} = max(t(d))$ seconds. Considering the randomness of jobs, we introduce a stochastic service system theory (M/M/1 queuing model) to handle the issue [28], [29]. Fig. 3 shows the queuing model of user tasks in VNDC. This figure indicates that all admitted requests will be maintained in a queue until they are assigned the requested resource. Each incoming service request has to be processed within a deadline (T_{total}) defined in the SLA. In the paper, we divide the SLA into two aspects. First, the waiting-plus-computation time for per job should be less than or equal to T. Second, the total time (including the queuing time, computation time, and network-related time) for per job should be less than or equal to T_{total}. Let λ be the average arrival rate per unit time for user jobs and μ be the average service rate per unit time for jobs. The average service time for user jobs, ρ, is given by

$$\rho = \frac{\lambda}{\mu}, \tag{16}$$

where $0 \leq \rho \leq 1$ is used to judge whether the queue is busy or not. The closer the value of ρ is to zero, the shorter the waiting time of user jobs in the queue, while the closer the value of ρ is to one, the longer the waiting time for the user jobs in the queue.

Definition 1. *Let H be the expected completion time (i.e., the waiting time in the queue plus the execution time) for the user jobs. Let H_q be the expected waiting time of the job in the queue. H and H_q will satisfy the following:*

$$H = \frac{1}{\mu - \lambda}, \tag{17}$$

$$H_q = \frac{\lambda}{\mu(\mu - \lambda)} = \rho \times \frac{1}{\mu - \lambda}. \tag{18}$$

Proof. Suppose $P_n = P\{N = n\}$ represents the probability that the number of queuing tasks is equal to parameter n in the queue at any one time; specifically, P_0 is the probability that the number of queuing tasks is equal to parameter "0". According to the conclusion in [51], Eqs. (17) and (18) can be obtained. □

4 Optimization Problem and the Proposed Solution

In this section, we discuss the problem overview and formulate the VNDC total energy consumption. We then put forward a solution to address it.

4.1 Optimization Problem

The total energy consumed by a VNDC is formulated as a minimization of E_{total} under real-time constraints T_{total} (T_{total} is the maximum allowed queuing-plus-computing-plus communication time for per job) as follows:

$$Minimize(E_{total}) = Minimize\left(\sum_{i=1}^{m} (E_{cpu}(i) + E_{reconf}(i) + E_{com}(i) + E_{disk}(i)) \right), \tag{19}$$

subject to

$$\sum_{i=1}^{m} \sum_{k=1}^{Q} f_k(i) t_k(i) = S, \tag{20}$$

$$0 \leq t_k(i) \leq t(e) \quad 1 \leq i \leq m, 0 \leq k \leq Q, \tag{21}$$

$$\sum_{k=1}^{Q} (t_k(i)) = t(e), 1 \leq i \leq m, \tag{22}$$

$$t(w) + \sum_{k=1}^{Q} \frac{2 f_k(i) t_k(i)}{R(i)} + t(e) \leq T_{total}, \tag{23}$$

$$\sum_{i=1}^{m} R(i) \leq R_{total}, \tag{24}$$

$$0 \leq R(i) \leq R_{total} \quad \forall i = 1,\ 2,\ \ldots,\ m. \tag{25}$$

TABLE 3
Parameter Values for the Following Experiment

Parameter	Value	Parameter	Value
λ	4	Q	5
μ	5	R_{total}	100 (Gbit/s)
m	10	C_{CS}	604.8 (Tb)
η	0.5(mW)	P_{CS}	4.9 (kW)
k_c	0.05 $J/(GHz)^2$	E_T	$1.42 \times (10)^{-9} (W/b)$
T_{total}	7(s)	D	20
T	5(s)	P_{hd}	225(W)
P_{idle}	0.5(W)	C_{hd}	800(Mb/s)

Eq. (19) is the objective function, which aims to optimise the total consumption of energy, E_{total}, by the VNDC. Eq. (20) guarantees that a job can only be divided into m parallel tasks. Eq. (21) ensures that each computation interval is less than $t(e)$ and that no negative execution time exists for any tasks. Eq. (22) ensures that the computation time does not exceed the execution time of a task, while Eq. (23) ensures that the total time (include the queuing time, computation time, and network-related time) for per job is less or equal to T_{total}. The combination of both Eqs. (22) and (23) guarantees the SLA requirement for users. The bandwidth inequality in Eq. (24) states that the bandwidth of each VM must be lower than the total offered bandwidth by the global network. Eq. (25) guarantees that the communication rate of the channel is less than or equal to the available network capacity.

4.2 Proposed Solution

In order to address the problem discussed in Section 4.1, we propose MCEC (minimize computing-plus-VM reconfiguration-plus-communication-plus-disk energy consumption) algorithm. The key part depends on how to carefully tune the workload fractions $\{f_k(i)t_k(i)), 1 \le i \le m, 0 \le k \le Q\}$, the end-to-end link data transfer rates $\{R(i), i = 1, 2, \ldots, m\}$, and the rates of computing $\{f_k(i), 1 \le i \le m\}$. In contrast with the existing algorithms that optimize one or two energy consuming resources (such as the computational energy consumption E_{cpu}) in a VNDC, our objective of proposing MCEC is to optimise the total energy consumed by the VNDC for processing the input workload.

All in all, MCEC aims to optimize the total energy consumption E_{total} while meeting the user QoS requirements defined in the SLA. The pseudo-code for MCEC is shown in Algorithm 1.

The algorithm takes μ, λ, m, Q, S and b as inputs and outputs $t_k(i)$ and E_{total}. The main idea of MCEC is to optimize E_{total} while meeting the SLA requirements (see Eqs. (19), (20), (21), (22), (23), (24), and (25)), thus providing a solution to the optimization problem. The algorithm first initializes the parameters shown in Table 3. It then computes the task queue time $t(w)$ (see Eq. (14)). The algorithm finds all possible times that VM i operates at frequency $f_k(i)$ while observing Eqs. (20), (21), (22), (23), (24), and (25). Regarding each candidate time, we compare the value of E_{total} and choose the time that makes E_{total} a minimum (Line 9 - Line 16). Finally, MCEC returns the results, including $t_k(i)$ and E_{total}.

Algorithm 1. MCEC Algorithm

Input: The parameters μ, λ, m, Q, S, b
// please refer to Table 3 in Section 5.2
Output: $t_k(i) \quad 1 \le i \le m \quad 0 \le k \le Q; E_{total}$
1: Initialize parameters //such as μ, λ, m, Q, S, b ;
2: Compute $t(w)$ according to Eq. (22); // task queue time
3: $f_k(i) \leftarrow f \leftarrow \{0.15, 1.867, 2.133, 2.533, 2.668\}$;
// variable Δ is an independent identically distributed
//random sequence, in this paper, S=8, b=2
4: **for** (each workload $\Delta \in [S-b, S+b]$) **do**
5: **for** ($i = 1 \;\; to \;\; m$) **do**
6: $E(i) \leftarrow MAX$;
//initial value is the maximum value
7: Find all possible times that VM i operates at frequency $f_k(i) (0 \le k \le Q)$ using Eqs. (18), (19), (20), (21), (22), and (23);
8: Get all solutions for $t_k(i)$ and save to $T_k(i)$;
// $T_k(i)$ is a Vector class for saving candidate value
9: **for** (each Vector $\tau \in T_k(i)$) **do**
10: Calculate $E_{cpu}(i), E_{reconf}(i), E_{com}(i), and\ E_{disk}(i)$ according to Eqs. (3)-(4), Eqs. (10)-(11);
11: $E_{total}(i) \leftarrow (E_{cpu}(i) + E_{reconf}(i) + E_{com}(i) + E_{disk}(i))$;
12: **if** ($E_{total}(i) \; < \; E(i)$) **then**
13: $E(i) \leftarrow E_{total}(i)$;
14: $t_k(i) \leftarrow \tau$; //an optimal value denoted by;
15: **end if**
16: **end for**
17: **end for**
18: $E_{total} \leftarrow \sum_{i=1}^{m}(E(i))$;
//get the total energy consumption
19: **end for**
20: **return** $t_k(i)\ and\ E_{total}$;

4.3 Theoretical Analysis

In this part, we will derive some theoretical conclusions for the proposed MCEC algorithm. Although these conclusions do not guarantee performance, they do give us some guidance and inspiration. We present a total of four Theorems. Theorem 1 is the time complexity of the proposed algorithm, and Theorems 2 and 3 lay the foundation for Theorem 4. The logical relationship of the four Theorems is as follows: Theorem 1 proves the complexity of the proposed algorithm theoretically. Theorem 2 lays a foundation for Theorem 3. Specifically, Theorem 3 needs to use the Eq. (26) from Theorem 2. Similarly, To prove Theorem 4, it needs to use the conclusion from Theorem 3. Theorem 4 confirms that the communication energy consumption is a convex function. Thus, there is an optimal value for communication energy consumption. The Communication energy consumption accounts for a large part of the total energy consumption (the sum of computation energy consumption plus VM configuration energy consumption plus communication energy consumption plus disk energy consumption). In this paper, the MCEC algorithm optimizes the total energy consumption while meeting user QoS requirements defined in the SLA.

Theorem 1. *The complexity of MCEC is $O(n \times m \times Q \times \theta)$, where n refers to the number of workloads, m is the number of VMs, Q is the number of discrete CPU frequencies, and θ is the number of $T_k(i)$.*

Proof. In MCEC, each job is processed by m VMs in parallel. As for each VM, in order to save energy consumption, each VM can operate at Q different discrete frequencies. Furthermore, MCEC needs to find an optimal solution from object $T_k(i)$. Therefore, when the number of workloads is n, the complexity of MCEC is $O(n \times m \times Q \times \theta)$. □

Theorem 2. *The following inequality holds:*

$$\sum_{i=1}^{m}\sum_{k=1}^{Q} f_k(i)t_k(i) \le \frac{1}{2}R_{total}\Big(\Big(T_{total}-t(w)-t(e)\Big)\Big).$$

Proof. According to Eq. (21), we can obtain Eq. (28):

$$\sum_{k=1}^{Q} f_k(i)t_k(i) \le \frac{1}{2}R(i)\Big(T_{total}-t(w)-t(e)\Big). \tag{26}$$

Using the summation on both sides of Eq. (26), we can obtain the following expression:

$$\sum_{i=1}^{m}\sum_{k=1}^{Q} f_k(i)t_k(i) \le \sum_{i=1}^{m}\frac{1}{2}R(i)\Big(T_{total}-t(w)-t(e)\Big). \tag{27}$$

Applying Eq. (23), Eq. (27) can be modified to have the following form:

$$\sum_{i=1}^{m}\sum_{k=1}^{Q} f_k(i)t_k(i) \le \frac{1}{2}R_{total}\Big(\Big(T_{total}-t(w)-t(e)\Big)\Big). \tag{28}$$

Therefore, Theorem 2 has been proven. □

Theorem 3. *The following equation holds:*

$$\begin{aligned}&\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\Big(R(i)\Big)\frac{2f_k(i)t_k(i)}{R(i)}\\ &= (T_{total}-t(w)-t(e))\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\left(\frac{2f_k(i)t_k(i)}{T_{total}-t(w)-t(e)}\right).\end{aligned}$$

Proof.

$$\begin{aligned}&\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\Big(R(i)\Big)\frac{2f_k(i)t_k(i)}{R(i)}\\ &= \sum_{i=1}^{m} p_{net}\Big(R(i)\Big)\left(\sum_{k=1}^{Q}\frac{2f_k(i)t_k(i)}{R(i)}\right).\end{aligned} \tag{29}$$

According to Eq. (26), Eq. (29) can be put in the following form:

$$\begin{aligned}&\sum_{i=1}^{m} p_{net}\Big(R(i)\Big)\left(\sum_{k=1}^{Q}\frac{2f_k(i)t_k(i)}{R(i)}\right)\\ &\le \Big(T_{total}-t(w)-t(e)\Big)\sum_{i=1}^{m} p_{net}\Big(R(i)\Big).\end{aligned} \tag{30}$$

On the other hand, according to Eq. (21), we get

$$R(i) \ge \frac{\sum_{k=1}^{Q} 2f_k(i)t_k(i)}{T_{total}-t(w)-t(e)}.$$

Therefore, Eq. (31) can be put in the following form:

$$\begin{aligned}&\Big(T_{total}-t(w)-t(e)\Big)\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\left(\frac{2f_k(i)t_k(i)}{T_{total}-t(w)-t(e)}\right)\\ &= \Big(T_{total}-t(w)-t(e)\Big)\sum_{i=1}^{m} p_{net}\left(\frac{\sum_{k=1}^{Q} 2f_k(i)t_k(i)}{(T_{total}-t(w)-t(e))}\right)\\ &\le \Big(T_{total}-t(w)-t(e)\Big)\sum_{i=1}^{m} p_{net}\Big(R(i)\Big).\end{aligned} \tag{31}$$

Based on the conclusion from both Eqs. (30) and (31), Theorem 3 has been proven. □

Theorem 4. *The communication energy consumption E_{com} is a convex function.*

Proof. According to the conclusion from Theorem 3

$$\begin{aligned}&\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\Big(R(i)\Big)\frac{2f_k(i)t_k(i)}{R(i)}\\ &= \Big(T_{total}-t(w)-t(e)\Big)\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\left(\frac{2f_k(i)t_k(i)}{T_{total}-t(w)-t(e)}\right).\end{aligned} \tag{32}$$

Eq. (32) is devided by a factor of 2, so we obtain Eq. (33), that is

$$\begin{aligned}&\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\Big(R(i)\Big)\frac{f_k(i)t_k(i)}{R(i)}\\ &= \Big(T_{total}-t(w)-t(e)\Big)\sum_{i=1}^{m}\sum_{k=1}^{Q} p_{net}\left(\frac{f_k(i)t_k(i)}{T_{total}-t(w)-t(e)}\right).\end{aligned} \tag{33}$$

Applying the definitions and properties of convex functions to Eq. (33), we obtain the conclusion: E_{com} is a convex function. Therefore, Theorem 4 has been proven. □

5 Performance Evaluation

We validated the proposed MCEC using extensive performance evaluation and compare the performance of MCEC against several baseline approaches.

5.1 Experimental Setting

We used the CVX solver in Matlab. Table 3 presents the parameters and their values used in the experiment. For example, k_c is the reconfiguration energy consumption coefficient, and it is mainly related to the cost of energy consumption per portion of frequency transferring in the channel. It means if we have more k_c values the cost of channel for energy transferring is higher. $R(i)$ is the communication rate of the ith end to end connection. Regarding to the meanings of other parameters, please refer to Table 2.

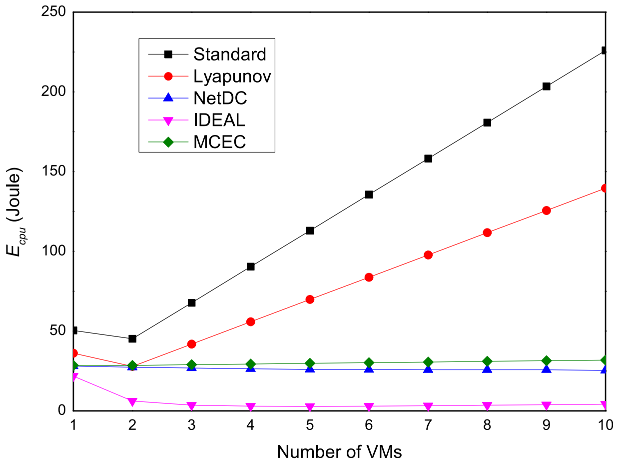


Fig. 4. E_{cpu} value.

The offered workload is modeled as iid random sequence S whose value belongs to the interval $[S-b, S+b]$, with S=8 [Gbit] and b=2 [Gbit] [29]. The discrete frequencies for the experiment are $f_k(i) = f = \{0.15, 1.867, 2.133, 2.533, 2.668\}$ [27]. To ensure the accuracy of experimental data, each experiment has been numerically evaluated in more than 1000 independent runs.

We consider Lyapunov [26], Standard [27], NetDC [28], and IDEAL [41] as baseline algorithms for comparison with MCEC. Note that the Standard algorithm uses the DVFS method to save power, while the Lyapunov algorithm leverages the queuing information to maximize the throughput, so as to achieve energy saving. Although both the Standard and Lyapunov algorithms focus on optimizing the computational energy consumption (E_{cpu}), they do not take into account the VM reconfiguration energy consumption (E_{reconf}), communication energy consumption (E_{com}), and disk energy consumption (E_{disk}). Since the NetDC algorithm depends on the proportion of computed real frequency, it cannot be applied in actual environment. In addition, the NetDC algorithm does not consider the internal VM energy consumption as well as disk energy consumption. In contrast with the NetDC algorithm, IDEAL is an idealized algorithm, and it does not highlight the VM reconfiguration and disk energy consumption. Similarly the IDEAL algorithm impractical in real setting since it requires working on continuous series of frequencies.

5.2 Experimental Results and Analysis

In this part, we evaluate the MCEC algorithm through a series of experiments, including the scenario one used for performance comparison with other algorithms, scenario two used for robust estimation, and scenario three used for performance comparison under real-world workload trace.

5.2.1 Scenario One

(1) Computation Energy Consumption. Fig. 4 shows the average computational energy consumption (E_{cpu}) (y-axis) of the five different algorithms (i.e., Standard, Lyapunov, NetDC, IDEAL, and MCEC) under different number of VMs (x-axis). The results of this experiment shows that IDEAL is the best, NetDC is second, MCEC is third, Lyapunov is fourth, and Standard is the worst. When the energy consumption optimization is solely focused on CPU consumption, IDEAL obtains the optimal frequency at each time slice, which leads to less computational energy consumption. MCEC is close to NetDC and better than the Standard and Lyapunov algorithms. This is because MCEC can find the proper time for the active discrete frequencies. However, the IDEAL and NetDC algorithms cannot be applied in real scenarios. The reason for Lyapunov performing better than Standard is that Lyapunov utilizes the queuing information to reduce energy consumption, while the Standard algorithm reduces energy consumption by reducing the virtual machines CPU frequencies.

Fig. 4 also indicates that, by simply increasing VM numbers (for example, when the number of VMs goes from 1 to 2), quite a lot of energy saving can be achieved. The reason is that when the number of VMs increases, less quota ($f_k(i)t_k(i)$) of S will be assigned to each VM. Therefore, the needed frequency and time for processing the task decrease, as does the computational energy consumption (E_{cpu}). However, when VM numbers is more than 2 ($m > 2$), in terms of the three algorithms (NetDC, IDEAL, and MCEC), there is no influence of increasing number of VMs on the computational energy consumption. This is because the system goes into the idle mode most of the time, and fewer or no time is allocated to the residual frequencies. As for the Lyapunov and Standard algorithms, by increasing the VM numbers (for example, when $m > 2$), the computational energy consumption (E_{cpu}) increases because both the Lyapunov and Standard algorithms are not suited for a varying workload.

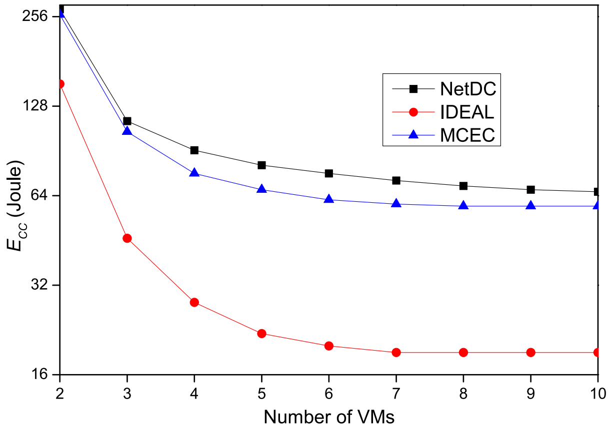
(2) Computation Plus Communication Energy. Figs. 5a, 5b, and 5c show the performance of the NetDC, IDEAL and MCEC algorithms when the number of VMs vary. The sum of computation plus communication energy consumption is defined as follows:

$$E_{CC} = E_{cpu} + E_{com}. \tag{34}$$

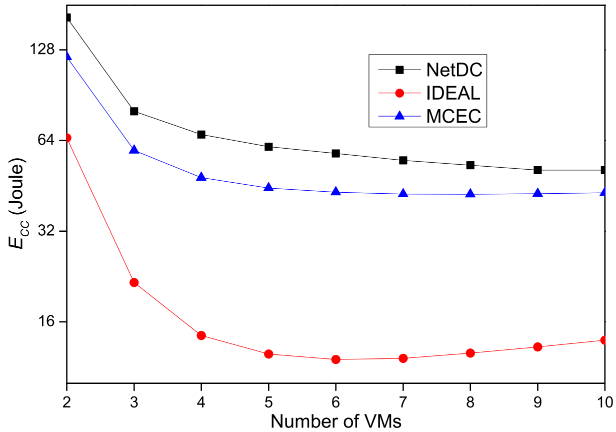
We excluded the Standard and Lyapunov algorithms since they do not consider the optimization of energy consumption due to communication. We also varied η, with $0.2 \leq \eta \leq 0.8$.

Fig. 5b shows that when η = 0.5, the IDEAL algorithm leads to less E_{CC} energy consumption, while NetDC is the lowest performing algorithm. MCEC lies between the two algorithms. Note that the IDEAL algorithm is expected to perform well since it is an ideal algorithm but cannot be used in real environments. MCEC saves an excess of 10% E_{CC} energy consumption compared to NetDC. This is because the NetDC algorithm depends on the likelihoods of the preceding and subsequent active discrete frequencies. This strategy leads to much more communication energy consumption and thus also the E_{CC} energy consumption.

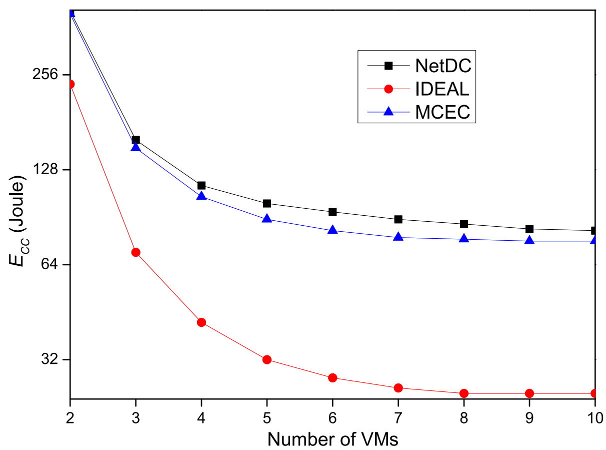
Figs. 5b and 5c investigate the influences of the parameter η on the sum of computation plus communication energy consumption E_{CC}. As a result, Figs. 5a, 5b, and 5c confirm that with the growth of η, E_{CC} increases, proving that η has an impact on energy consumption. Additionally, the results reveal that with an increase in VM numbers, the energy consumption E_{CC} decreases. It can be explained that more VMs means more processing power. Thus, the needed



(a) η=0.5



(b) η=0.2



(c) η=0.8

Fig. 5. E_{CC} in (Joule) results for $\eta = \{0.2, 05, 0.8\}$.

frequency and time for processing the task decreases, lowering the E_{CC} energy consumption.

(3) Three Elements Energy Consumption. Figs. 6, 7, 8, 9, 10, and 11 show the sum of the computation, external VM configuration, and communication energy consumption for the NetDC and MCEC algorithms (y-axis) as the number of VMs vary (x-axis)

$$E_{CRC} = E_{cpu} + E_{reconf}(Ext) + E_{com}. \tag{35}$$

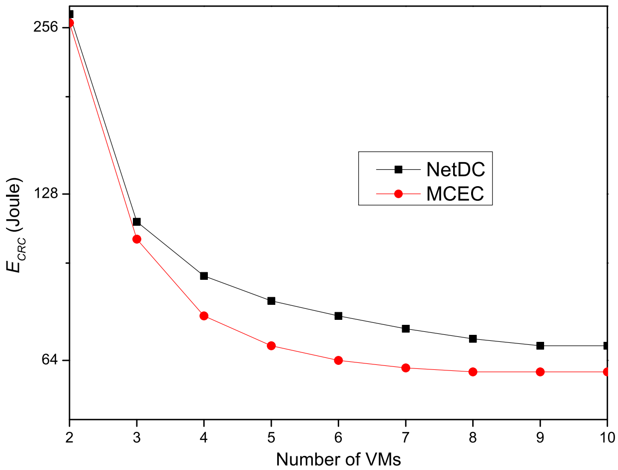


Fig. 6. E_{CRC} value when k_c=0.05.

As the IDEAL algorithm does not consider the VM reconfiguration energy consumption, we excluded it from the experiment. In this experiment, we varied k_c, with $0.005 \leq k_c \leq 0.5$.

Fig. 6 shows that, in comparison with the NetDC algorithm, the MCEC algorithm saves more than 10% E_{CRC} energy consumption. The reason behind this is that MCEC can provide the proper time to discrete frequencies for processing the incoming workload. Judging from Fig. 7, NetDC leads to less external VM configuration energy consumption ($E_{reconf}(Ext)$) compared with MCEC. The reason is that, in contrast to the computational (E_{cpu}) and communication (E_{com}) energy consumptions, VM configuration energy consumption ($E_{reconf}(Ext)$) account for a small proportion of total energy consumption. Therefore, the solution attempts to search the best objective variables to optimise the energy consumed by computation (E_{cpu}) and communication energy consumption (E_{com}), thus optimizing the E_{CRC}. Figs. 6, 7, 8, 9, 10, and 11 also confirm that the VM configuration energy consumption ($E_{reconf}(Ext)$) grows with the increases in the value of k_c.

(4) Four Resources Energy Consumption. In this part, we describe the experimental results regarding the sum of computation energy consumption plus external VM configuration energy consumption plus communication energy consumption plus disk energy consumption for MCEC, which is defined as follows:

$$E_{CRCD} = E_{cpu} + E_{reconf}(Ext) + E_{com} + E_{disk}. \tag{36}$$

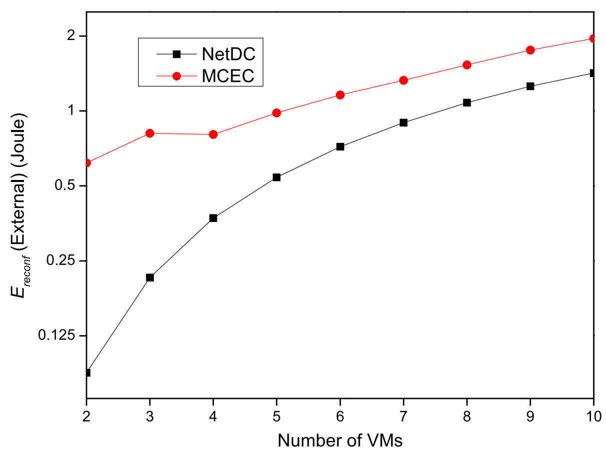


Fig. 7. $E_{reconf}(Ext)$ value when k_c=0.05.

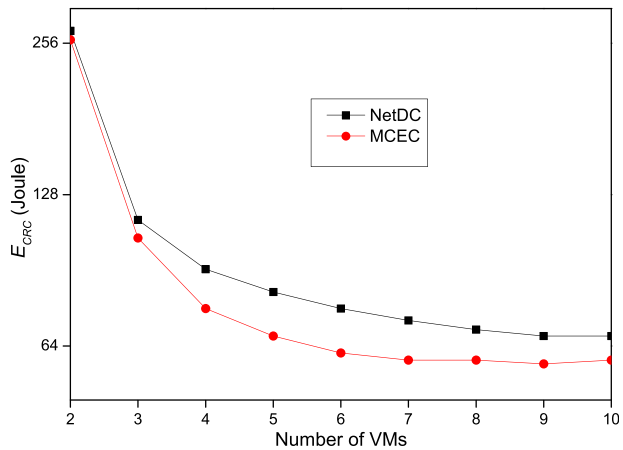


Fig. 8. E_{CRC} value when k_c=0.005.

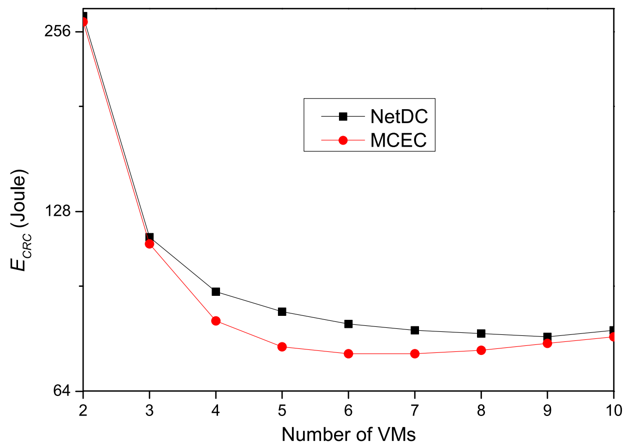


Fig. 10. E_{CRC} value when k_c=0.5.

As Standard, Lyapunov, NetDC, and IDEAL do not consider the disk energy consumption, this experiment did not consider these four algorithms. However, we extended the NetDC such that it considers the disk energy consumption for the purpose of benchmark.

Fig. 12 displays the performance of MCEC and extended NetDC with respect to E_{CRCD} as the number of VMs vary (x-axis). Fig. 12 indicates that MCEC has better performance than the extended NetDC. This can be explained by the fact that MCEC can find a better solution to the optimization problem. It also shows that the E_{CRCD} energy consumption for MCEC can be kept at a stable state when $m \geq 6$ (m is the number of VMs). This is because the MCEC algorithm is capable of assigning the proper time for each active discrete frequency when $m \geq 6$.

Fig. 13 illustrates the performance of MCEC and extended NetDC with respect to E_{disk} as the number of VMs vary (x-axis). Judging from Fig. 13, the extended NetDC leads to the least E_{disk} energy consumption compared with MCEC during the optimization of E_{CRCD}. The reason can be explained as follows: in contrast to computational (E_{cpu}) and communication (E_{com}) energy consumptions as well as the energy consumed by disk (E_{disk}) only account for a small proportion of total energy consumption. Therefore, the solution tries to search the optimal objective variables to reduce the computational (E_{cpu}) and communication (E_{com}) energy consumptions, thus optimizing E_{CRCD}.

(5) Job Execution Time. Fig. 14 indicates the execution time per task for the first 100 tasks under different VM numbers. Judging from this figure, the per task execution time decreases as the VM numbers increases. This is because increasing VMs means more processing power and thus the less execution time per task.

5.2.2 Scenario Two

In Section 5.2.1 (Scenario one), we have selected Lyapunov [26], Standard [27], NetDC [28], and IDEAL [41] as baseline algorithms for comparison with MCEC. We have mainly evaluated the MCEC algorithm in terms of computation energy consumption, computation plus communication energy consumption, computation plus communication energy consumption plus VM configuration energy consumption, and total energy consumption (computation plus communication energy consumption plus VM configuration energy consumption plus disk storage energy consumption).

In this part (Scenario two), we measure the robustness for the proposed algorithm (MCEC) by varying experimental parameters. As Standard, Lyapunov, NetDC, and IDEAL do not consider the disk energy consumption, this experiment did not consider these four algorithms (In Scenario one, we have done a detailed experimental evaluation for the five algorithms). However, we extended

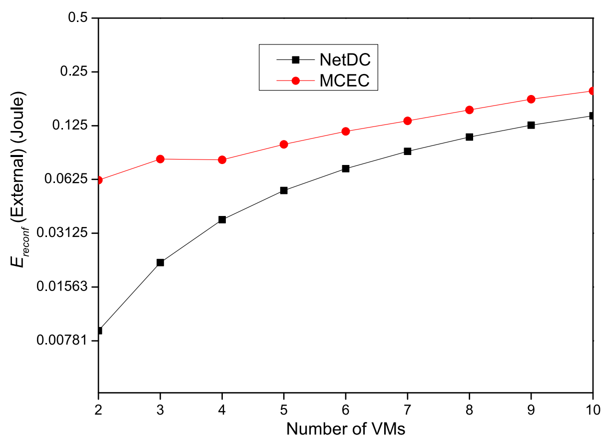


Fig. 9. $E_{reconf}(Ext)$ value when k_c=0.005.

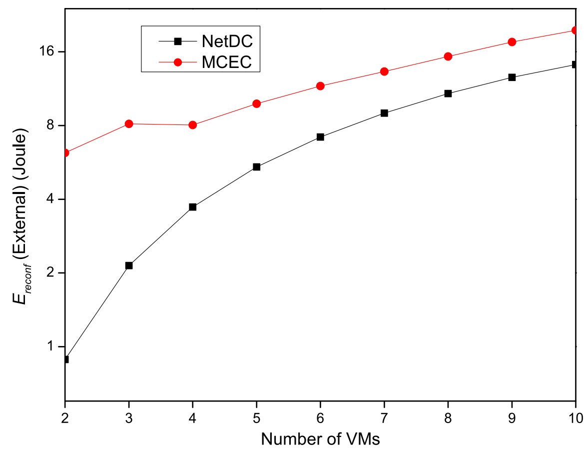


Fig. 11. $E_{reconf}(Ext)$ value when k_c=0.5.

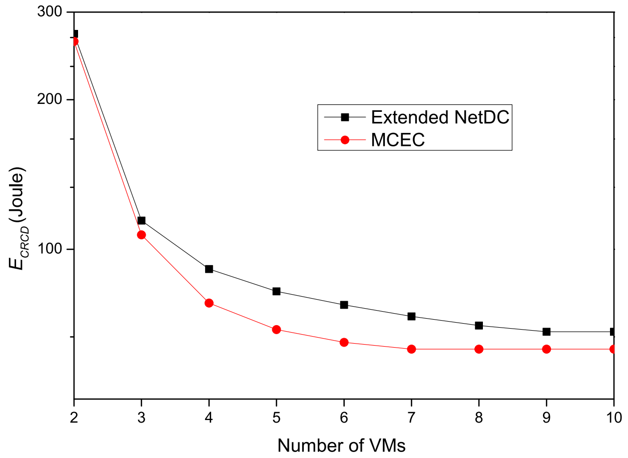


Fig. 12. The value of E_{CRCD}.

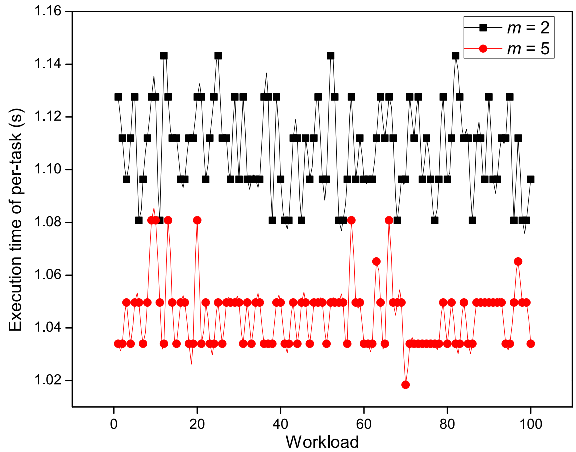


Fig. 14. Execution time per task for the first 100 tasks.

the NetDC such that it considers the disk energy consumption for the benchmark. In scenario two, the offered workload is modelled as an iid random sequence S whose value belongs to the interval $[S-b, S+b]$, with S=8 [Gbit] and b=2 [Gbit] [29]. The discrete frequencies for the experiment are $f_k(i) = f = \{0.3, 0.533, 0.667, 0.800, 0.933\}$, which come from the power-scalable real Crusoe cluster with the TM-5800 CPU in [27].

(1) Performance Comparison Under Different VMs. Fig. 15 illustrates the total energy consumption (computation energy consumption plus VM configuration energy consumption plus communication energy consumption plus disk energy consumption) comparison under different VMs. Fig. 15 indicates that MCEC can obtain a better performance than extended NetDC algorithm under different VMs, the reason can be explained by the fact that MCEC can find a better solution to minimize the total consumption of energy. Fig. 15 also illustrates that the total consumption of energy increases with the growth of VM numbers. The reason is that more VMs means more processing energy consumption.

(2) Performance Comparison Varying the Parameter k_c. As we mentioned above, in Scenario one (Section 5.2.1), we have done a detailed experimental evaluation for the five algorithms (Lyapunov [26], Standard [27], NetDC [28], IDEAL [41], and MCEC). As the Standard, Lyapunov, NetDC, and IDEAL do not consider the disk energy consumption, this experiment did not consider these four algorithms (in scenario one, we did a detailed experimental evaluation). However, we extended the NetDC such that it considers the disk energy consumption for the purpose of benchmark. In scenario two, we measure the robustness of the proposed algorithm (MCEC) by varying experimental parameters in total energy consumption.

In this Section, we explore the impact of the parameter k_c on the total energy consumption. In the paper, the setting of parameter k_c is according to the published [28] and [41]. As the parameter k_c ranges from 0.005 to 0.5 ($0.005 \leq k_c \leq 0.5$), we varied the value of parameter k_c from 0.005 to 0.5, increased by a factor of 10. In other words, $k_c \in \{0.005, 0.05, 0.5\}$. Parameter k_c is the reconfiguration energy consumption coefficient, and it is mainly related to the cost of energy consumption per portion of frequency transferring in the channel. According to the Eq. (6), the more k_c values, the more VM reconfiguration energy consumption. Fig. 16 illustrates that MCEC has a better performance than extended NetDC algorithm under different value of parameter k_c ($0.005 \leq k_c \leq 0.5$). On average, MCEC saves 9 percent of energy consumption compared with Extended NetDC algorithm. The reason lies in MCEC optimizing the overall consumed energy by computation, VM configuration, communication, and disk subsystems. Fig. 16 also confirms that the total energy consumption grows with the increasing of the value of k_c. The reason is that the greater the value of k_c, the greater the value of VM configuration energy consumption and total energy consumption.

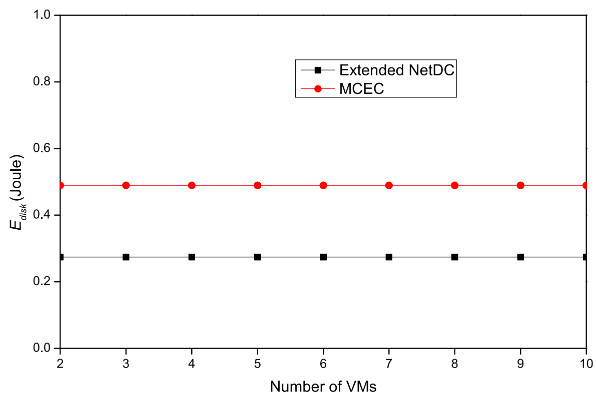


Fig. 13. The value of E_{disk}.

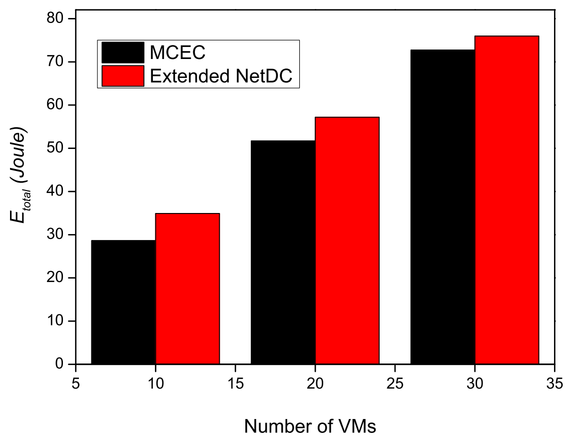


Fig. 15. Energy consumption comparison.

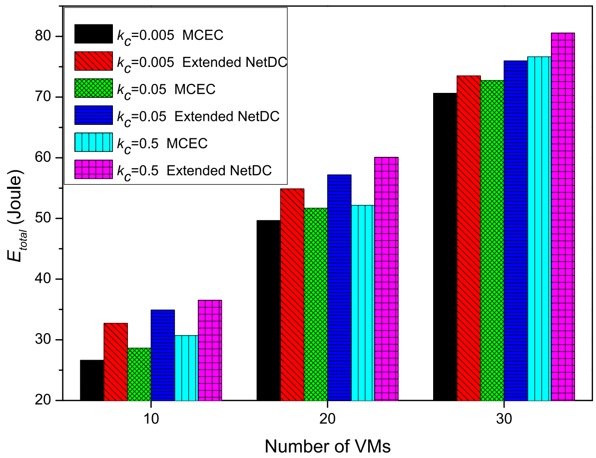


Fig. 16. Energy consumption comparison when varying k_c.

(3) Performance Comparison Varying the Parameter η. Parameter η has an impact on the communication energy consumption. By varying the η $(0.2 \leq \eta \leq 0.8)$, we can obtain the comparison of total energy, as shown in Fig. 17.

Fig. 17 reveals that MCEC can obtain a better performance than extended NetDC algorithm under different value of parameter η $(0.2 \leq \eta \leq 0.8)$. The reason behind this is that MCEC can find a better solution to optimise the total consumption of energy. Fig. 17 also indicates the overall consumption of the energy grows in the value of η. The reason is that the greater the value of η, the greater the value of communication energy consumption and total energy consumption.

5.2.3 Scenario Three

Different from Scenario one and two, in this scenario (Scenario three), we have mainly evaluated the performance for the MCEC algorithm under real-world workload trace. The real-world workload trace is considered in [52]. Under the real-world workload trace, k_c=0.05 $J/(GHz)^2$, T=8 (s), and each job has a mean length of 10 [Mbit]. Fig. 18 indicates the comparison of the total energy consumption under the real-world workload.

The result from Fig. 18 shows that, in terms of the total energy consumption, MCEC leads to the best performance, extended NetDC is the second, Lyapunov is the third, and Standard the worst. The reason can be explained as follows:

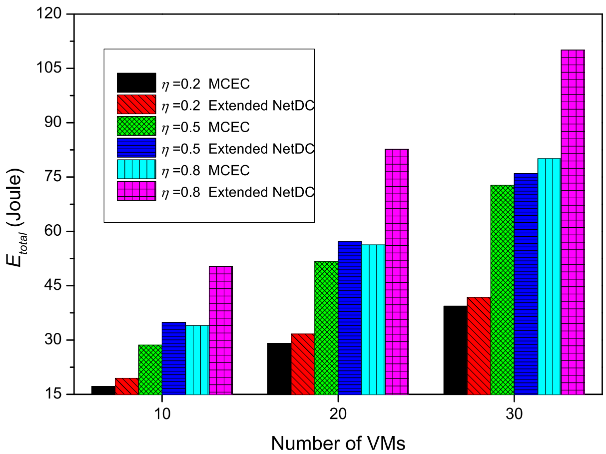


Fig. 17. Energy consumption comparison when varying η.

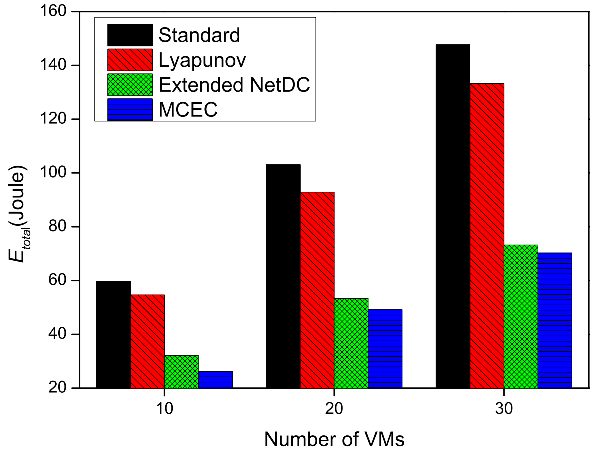


Fig. 18. Energy consumption comparison under real-world workload.

MCEC can find a better solution for the optimization problem by assigning a proper time slot for each frequency. By using this optimization strategy each time, MCEC leads to the least total energy consumption. Extended NetDC is better than Lyapunov and Standard. The reason is that extended NetDC considers the energy consumption by the CPU, network, disk storage, and VM reconfiguration. Different from MCEC and extended NetDC algorithm, Lyapunov and Standard only optimize the computational energy consumption. The reason for Lyapunov performing better than Standard is that Lyapunov utilizes the queuing information to reduce energy consumption, while Standard reduces energy consumption by reducing the virtual machines CPU frequencies.

6 Conclusion and Hint for Future Works

The problem of how to efficiently reduce the high consumption of energy by VNDCs while ensuring QoS has been receiving significant attention among researchers and practitioners. To address this problem, we put forward an adaptive energy-aware algorithm that considers computational energy consumption, VM configuration energy consumption, communication energy consumption, and disk energy consumption. The results of the experiments demonstrate that minimising the total energy consumed by all resources is more efficient than just minimising energy consumed by one or two resources. In addition, the results from simulation and real experiments also confirms that most of the energy is consumed by computation and communication resources, while energy consumed by the VM configuration and disk contributes only a small portion to the overall energy consumption in the system. Moreover, our algorithm, MCEC, performs better than other energy-saving algorithms. MCEC can be used in real-life scenarios or on real-world cloud platforms to minimise the overall consumption of energy of the system while ensuring end-user QoS requirements.

In the future, we will consider more components related to the energy consumption in VNDCs to maximize power saving.

Acknowledgments

The authors would like to express his gratitude to the anonymous reviewers for their criticism and comments on improving the quality of the manuscript.

References

[1] P. Cong, J. Zhou, L. Li, K. Cao, and K. Li, "A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud," *ACM Comput. Surv.*, vol. 53, no. 2, pp. 1–38, 2020.

[2] J. Zhu, X. Li, R. Ruiz, W. Li, H. Huang, and A. Y. Zomaya, "Scheduling periodical multi-stage jobs with fuzziness to elastic cloud resources," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 12, pp. 2819–2833, Dec. 2020.

[3] C. Liu, K. Li, J. Liang, and K. Li, "Service reliability in an HC: Considering from the perspective of scheduling with load-dependent machine reliability," *IEEE Trans. Rel.*, vol. 68, no. 2, pp. 476–495, Jun. 2019.

[4] G. S. Aujla, M. Singh, N. Kumar, and A. Zomaya, "Stackelberg game for energy-aware resource allocation to sustain data centers using RES," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 1109–1123, Oct.–Dec. 2019.

[5] J. Mei, K. Li, Z. Tong, Q. Li, and K. Li, "Profit maximization for cloud brokers in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 1, pp. 190–203, Jan. 2019.

[6] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, and P. Li, "Energy-efficient scheduling for real-time systems based on deep Q-learning model," *IEEE Trans. Sustain. Comput.*, vol. 4, no. 1, pp. 132–141, Jan.–Mar. 2019.

[7] Z. Zhou, F. Liu, S. Chen, and Z. Li, "A truthful and efficient incentive mechanism for demand response in green datacenters," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 1, pp. 1–15, Jan. 2020.

[8] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in SaaS clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 10, pp. 2648–2658, Oct. 2014.

[9] W. Deng, F. Liu, H. Jin, C. Wu, and X. Liu, "Multigreen: Cost-minimizing multi-source datacenter power supply with online control," in *Proc. 4th Int. Conf. Future Energy Syst.*, 2013, pp. 149–160.

[10] X. Guo, M. Zhou, S. Liu, and L. Qi, "Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption," *IEEE Trans. Automat. Sci. Eng.*, vol. 18, no. 2, pp. 804–816, Apr. 2021.

[11] E. A. B. Nieto, S. Rezazadeh, and R. D. Gregg, "Minimizing energy consumption and peak power of series elastic actuators: A convex optimization framework for elastic element design," *IEEE/ASME Trans. Mechatron.*, vol. 24, no. 3, pp. 1334–1345, Jun. 2019.

[12] Z. Zhou, F. Liu, R. Zou, J. Liu, H. Xu, and H. Jin, "Carbon-aware online control of geo-distributed cloud services," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 9, pp. 2506–2519, Sep. 2016.

[13] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "TOFFEE: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing," *IEEE Trans. Cloud Comput.*, early access, Jun. 20, 2019, doi: 10.1109/TCC.2019.2923692.

[14] H. Sun, F. Zhou, and R. Q. Hu, "Joint offloading and computation energy efficiency maximization in a mobile edge computing system," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3052–3056, Mar. 2019.

[15] M. Ghetas, "A multi-objective monarch butterfly algorithm for virtual machine placement in cloud computing," *Neural Appl.*, vol. 33, pp. 11011–11025, 2021.

[16] M. A. Rostami, A. Kavousi Fard, and T. Niknam, "Expected cost minimization of smart grids with plug-in hybrid electric vehicles using optimal distribution feeder reconfiguration," *IEEE Trans. Ind. Informat.*, vol. 11, no. 2, pp. 388–397, Apr. 2015.

[17] Z. Zhou, F. Liu, and Z. Li, "Bilateral electricity trade between smart grids and green datacenters: Pricing models and performance evaluation," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3993–4007, Dec. 2016.

[18] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Gener. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, 2012.

[19] G. Tang, W. Jiang, Z. Xu, F. Liu, and K. Wu, "NIPD: Non-intrusive power disaggregation in legacy datacenters," *IEEE Trans. Comput.*, vol. 66, no. 2, pp. 312–325, Feb. 2017.

[20] S. Azizi, M. Shojafar, J. Abawajy, and R. Buyya, "GRVMP: A greedy randomized algorithm for virtual machine placement in cloud data centers," *IEEE Syst. J.*, vol. 15, no. 2, pp. 2571–2582, Jun. 2021.

[21] S. Omer, S. Azizi, M. Shojafar, and R. Tafazolli, "A priority, power and traffic-aware virtual machine placement of IoT applications in cloud data centers," *J. Syst. Archit.*, vol. 115, 2021, Art. no. 101996.

[22] L. Wang *et al.*, "GreenDCN: A general framework for achieving energy efficiency in data center networks," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 1, pp. 4–15, Jan. 2014.

[23] S. Chen, Z. Zhou, F. Liu, Z. Li, and S. Ren, "CloudHeat: An efficient online market mechanism for datacenter heat harvesting," *ACM Trans. Model. Perform. Eval. Comput. Syst. (TOMPECS)*, vol. 3, no. 3, p. 11, 2018.

[24] W. Jiang, F. Liu, G. Tang, K. Wu, and H. Jin, "Virtual machine power accounting with shapley value," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst.*, 2017, pp. 1683–1693.

[25] W. Jiang, S. Ren, F. Liu, and H. Jin, "Non-IT energy accounting in virtualized datacenter," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst.*, 2018, pp. 300–310.

[26] R. Urgaonkar, U. C. Kozat, K. Igarashi, and M. J. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *Proc. IEEE Netw. Operations Manage. Symp.-NOMS 2010*, 2010, pp. 479–486.

[27] H. Kimura, M. Sato, Y. Hotta, T. Boku, and D. Takahashi, "Emprical study on reducing energy of parallel programs using slack reclamation by DVFs in a power-scalable high performance cluster," in *Proc. IEEE Int. Conf. Cluster Comput.*, 2006, pp. 1–10.

[28] N. Cordeschi, M. Shojafar, and E. Baccarelli, "Energy-saving self-configuring networked data centers," *Comput. Netw.*, vol. 57, no. 17, pp. 3479–3491, 2013.

[29] N. Cordeschi, M. Shojafar, D. Amendola, and E. Baccarelli, "Energy-efficient adaptive networked datacenters for the QOS support of real-time applications," *J. Supercomput.*, vol. 71, no. 2, pp. 448–478, 2015.

[30] S. K. Garg, S. Versteeg, and R. Buyya, "A framework for ranking of cloud computing services," *Future Gener. Comput. Syst.*, vol. 29, no. 4, pp. 1012–1023, 2013.

[31] X. Zhang, J.-J. Lu, X. Qin, and X.-N. Zhao, "A high-level energy consumption model for heterogeneous data centers," *Simul. Model. Pract. Theory*, vol. 39, pp. 41–55, 2013.

[32] G. G. Castañé, A. Nunez, P. Llopis, and J. Carretero, "E-mc2: A formal framework for energy modelling in cloud computing," *Simul. Model. Pract. Theory*, vol. 39, pp. 56–75, 2013.

[33] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proc. Int. Symp. Low Power Electron. Des.*, 2007, pp. 38–43.

[34] H. Hanson, S. W. Keckler, S. Ghiasi, K. Rajamani, F. Rawson, and J. Rubio, "Thermal response to DVFs: Analysis with an intel pentium M," in *Proc. Int. Symp. Low Power Electron. Des.*, 2007, pp. 219–224.

[35] C.-M. Wu, R.-S. Chang, and H.-Y. Chan, "A green energy-efficient scheduling algorithm using the DVFs technique for cloud datacenters," *Future Gener. Comput. Syst.*, vol. 37, pp. 141–147, 2014.

[36] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *J. Netw. Comput. Appl.*, vol. 52, pp. 11–25, 2015.

[37] M. R. V. Kumar and S. Raghunathan, "Heterogeneity and thermal aware adaptive heuristics for energy efficient consolidation of virtual machines in infrastructure clouds," *J. Comput. Syst. Sci.*, vol. 82, no. 2, pp. 191–212, 2016.

[38] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurr. Comput., Pract. Exp.*, vol. 24, no. 13, pp. 1397–1420, 2012.

[39] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1366–1379, Jul. 2013.

[40] Z. Zhou, Z.-G. Hu, T. Song, and J.-Y. Yu, "A novel virtual machine deployment algorithm with energy efficiency in cloud computing," *J. Central South Univ.*, vol. 22, no. 3, pp. 974–983, 2015.

[41] V. Mathew, R. K. Sitaraman, and P. Shenoy, "Energy-aware load balancing in content delivery networks," in *Proc. IEEE INFOCOM*, 2012, pp. 954–962.

[42] M. Shojafar, C. Canali, R. Lancellotti, and S. Abolfazli, "An energy-aware scheduling algorithm in DVFs-enabled networked data centers," in *Proc. 6th Int. Conf. Cloud Comput. Serv. Sci.*, 2016, pp. 387–397.

[43] M. Shojafar, C. Canali, R. Lancellotti, and J. Abawajy, "Adaptive computing-plus-communication optimization framework for multimedia processing in cloud systems," *IEEE Trans. Cloud Comput.*, vol. 8, no. 4, pp. 1162–1175, Oct.–Dec. 2020.

[44] J. Almeida, V. Almeida, D. Ardagna, Í. Cunha, C. Francalanci, and M. Trubian, "Joint admission control and resource allocation in virtualized servers," *J. Parallel Distrib. Comput.*, vol. 70, no. 4, pp. 344–362, 2010.

[45] L. Wu, S. K. Garg, and R. Buyya, "SLA-based admission control for a software-as-a-service provider in cloud computing environments," *J. Comput. Syst. Sci.*, vol. 78, no. 5, pp. 1280–1299, 2012.

[46] S. Azodolmolky, P. Wieder, and R. Yahyapour, "Cloud computing networking: Challenges and opportunities for innovations," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 54–62, Jul. 2013.

[47] Z. Qian *et al.*, "Timestream: Reliable stream computation in the cloud," in *Proc. 8th ACM Eur. Conf. Comput. Syst.*, 2013, pp. 1–14.

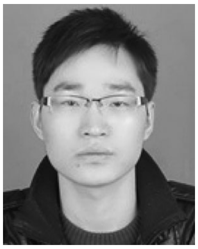
[48] D. Kondo, B. Javadi, P. Malecot, F. Cappello, and D. P. Anderson, "Cost-benefit analysis of cloud computing versus desktop grids," in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, 2009, vol. 9, pp. 1–12.

[49] J. Baliga, R. W. Ayre, K. Hinton, and R. S. Tucker, "Green cloud computing: Balancing energy in processing, storage, and transport," *Proc. IEEE*, vol. 99, no. 1, pp. 149–167, Jan. 2011.

[50] P. K. Sinha, P. Joshi, P. Pundir, M. Negi, and R. Goudar, "Enhanced single-pass algorithm for efficient indexing using hashing in map reduce paradigm," in *Proc. Intell. Comput., Netw., Informat.*, 2014, pp. 1195–1200.

[51] R. H. Möhring, A. S. Schulz, and M. Uetz, "Approximation in stochastic scheduling: The power of LP-based priority policies," *J. ACM*, vol. 46, no. 6, pp. 924–942, 1999.

[52] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi, "Analytic modeling of multitier internet applications," *ACM Trans. Web*, vol. 1, no. 1, pp. 1–36, 2007.

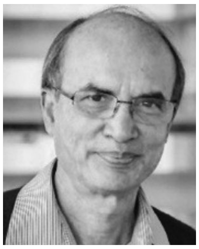


Zhou Zhou received the PhD degree from Central South University, Changsha, China, in 2017, majoring in computer science. Since 2018, he has been a postdoctoral with Hunan University. He is currently an associate professor with Changsha University. He has authored or coauthored more than 20 academic papers in cloud computing and edge computing and also holds several Chinese patents. His research interests include cloud computing, edge computing, energy consumption model, energy-efficient resource management, and virtual machine deployment.

Kenli Li (Senior Member, IEEE) received the PhD degree in compuer science from the Huazhong University of Science and Technology, China, in 2003. He is currently the dean, a full professor of computer science and technology with Hunan University, and the deputy director of National Supercomputing Center, Changsha. He has authored or coauthored more than 150 research papers in international conferences and journals. His research interests include parallel computing, high performance computing, grid, and cloud computing.

Jemal Abawajy (Senior Member, IEEE) is currently a full professor with Faculty of Science, Engineering and Built Environment, Deakin University, Australia. He has delivered more than 50 keynote and seminars worldwide and has been involved in the organization of more than 300 international conferences in various capacity including chair and general co-chair. He has authored or coauthored more than 250 refereed articles and supervised numerous PhD students to completion. He was on the editorial-board of numerous international journals, including the *IEEE Transactions on Cloud Computing*.

Mohammad Shojafar (Senior Member, IEEE) received the PhD degree in information and communication technology from the Sapienza University of Rome, Italy, with an assessment of Excellent, in 2016. He is curently an associate professor with the Network Security and an Intel Innovator working in the 5G and 6G Innovation Centre (5GIC/6GIC) with the University of Surrey, U.K., a PI of AutoTrust, a 750k euro 5G security project funded by European Space Agency in 2021, and a PI of 275k euro Marie Curie EU project PROSNODE, in 2019 and was contributed to some Italian projects in telecommunications like GAUChO, SAMMClouds, and SC2. He is currently an associate editor for the *IEEE Transactions on Network and Service Management*, *IEEE Transactions on Consumer Electronics*, *IEEE Systems Journal* and the *IET Communications*.



Morshed Chowdhury received the PhD degree in computing from Monash University, Australia, in 1999. He is currently a senior lecturer with the School of Information Technology, Deakin University, Burwood, Australia. He has more than 12 years of industry experiences and has authored or coauthored more than 100 research papers. He has organized a number of international conferences and was a member of program committees of several international conferences.

Fangmin Li received the PhD degree from Zhejiang University, Hangzhou, China, in 2001, majoring in computer science. He is currently a professor with Changsha University. He has authored several books on embedded systems and more than 30 academic papers in wireless networks, and also holds ten Chinese patents. His current research interests include cloud computing, wireless communications and networks, and energy-efficient resource management.

Keqin Li (Fellow, IEEE) is currently a SUNY distinguished professor of computer science with the State University of New York and the national distinguished professor with Hunan University, China. He has authored or coauthored more than 800 journal articles, book chapters, and refereed conference papers. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, big data computing, high-performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, and intelligent and soft computing. He holds more than 60 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top ten most influential scientists in distributed computing based on a composite indicator of Scopus citation database. He has chaired many international conferences. He is currently an associate editor for the *ACM Computing Surveys* and the *CCF Transactions on High Performance Computing*. He was on the editorial boards of the *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Services Computing*, and the *IEEE Transactions on Sustainable Computing*. He was the recipient of the several best paper awards.