AFED-EF: An Energy-efficient VM Allocation Algorithm for IoT Applications in a Cloud Data Center

Zhou Zhou, Mohammad Shojafar, *Senior Member, IEEE,* Mamoun Alazab, *Senior Member, IEEE,* Jemal Abawajy, *Senior Member, IEEE* and Fangmin Li

Abstract—Cloud Data Centers (CDCs) have become a vital computing infrastructure for enterprises. However, CDCs consume substantial energy due to the increased demand for computing power, especially for the Internet of Things (IoT) applications. Although a great deal of research in green resource allocation algorithms have been proposed to reduce the energy consumption of the CDCs, existing approaches mostly focus on minimizing the number of active Physical Machines (PMs) and rarely address the issue of load fluctuation and energy efficiency of the Virtual Machine (VM) provisions jointly. Moreover, existing approaches lack mechanisms to consider and redirect the incoming traffics to appropriate resources to optimize the Quality of Services (QoSs) provided by the CDCs. We propose a novel adaptive energy-aware VM allocation and deployment mechanism called AFED-EF for IoT applications to handle these problems. The proposed algorithm can efficiently handle the fluctuation of load and has good performance during the VM allocation and placement. We carried out extensive experimental analysis using a real-world workload based on more than a thousand PlanetLab VMs. The experimental results illustrate that AFED-EF outperforms other energy-aware algorithms in energy consumption, Service Level Agreements (SLA) violation, and energy efficiency.

Index Terms—Cloud data center (CDC), Internet of Thing (IoT), Energy efficiency, Resource provision, Virtual machine allocation (VMA), Service level agreement (SLA).

1 Introduction

Cloud Computing systems enable users on-demand access to a shared pool of configurable resources. These resources can be hardware (e.g., memory, CPUs, GPUs, networks, etc.) or software (i.e., services and applications). Generally, cloud computing systems are classified into Platform as a Service (PaaS) [1], Infrastructure as a Service (IaaS) [2], and Software as a Service (SaaS) [3]. PaaS provides Cloud users only the hardware for development and deployment where the users are responsible for installing and managing the required operating system and applications. In IaaS, the rented hardware comes with an operating system and the necessary libraries installed. In contrast, SaaS provides software on-demand, which means that the users do not have to purchase the software's original license.

Cloud Data Centers (CDCs) have become a popular computing platform for enterprises. With the Internet of Things (IoT) applications increasingly becoming popular, Cloud systems are receiving tremendous processing requests from these IoT applications. These applications generate heterogeneous data traffic with different demands to be analyzed and processed on CDCs. Each CDC consists of several servers, virtual infrastructures, and physical cables managing the informing traffic to the internet. CDCs utilize a set of key technologies, e.g., virtualization and Service Level Agreements (SLA) [4]. Virtualization facilitates the task of sharing cloud computing resources (e.g., dividing one powerful physical machine (PM) into a set of less powerful virtual machine (VM)).

Although virtualization improves the PMs' utilization by creating a series of VMs for providing specific services to meet the users' requirements, it also brings a new problem to cloud computing, namely, mapping VMs to the proper PMs. This problem is known as the VM deployment problem, which is known to be an NP-problem [5]. Many issues further complicate the VM deployment. For example, VMs should be deployed to PMs, and the software components should be deployed to VMs/containers [6]. Furthermore, the existence of heterogeneous VMs and PMs within the CDC complicates the VM deployment problem.

An optimal VM deployment is crucial for a cloud provider to reduce operating costs. At the same time, it important to increase the performance of the cloud user. Several different methods for deploying VMs in PMs on cloud computing systems have been proposed to achieve these conflicting objectives. An approach that focuses on improving the utilization of a given resource (e.g., central processing unit (CPU) utilization or memory have been discussed in [7] and in [8]) respectively. Since bandwidth is vital for efficiently deploying IoT applications on clouds [9], it is imperative to optimize the cloud platform bandwidth [10]. A deterministic method (e.g., best-fit and first-

- *Z. Zhou and F. Li are with the School of Computer Engineering and Applied Mathematics, Changsha University, Changsha, China*
  Z. Zhou is also with the College of Information Science and Engineering, Hunan University, Changsha, China
  F. Li is also with the Hunan Province Key Laboratory of Industrial Internet Technology and Security, Changsha University, Changsha, China
  Email: zhouzhou03201@126.com, lifangmin@whut.edu.cn
- *M. Shojafar is with the 5GIC & 6GIC, Institute for Communication Systems (ICS), University of Surrey, Guildford, GU27XH, United Kingdom*
  E-mail: m.shojafar@surrey.ac.uk
- *M. Alazab is with the College of Engineering, IT and Environment, Charles Darwin University, Australia*
  Email: alazab.m@ieee.org
- *J. Abawajy is with the Faculty of Science, Engineering and Built Environment, Deakin University, Geelong, Australia*
  Email: jemal.abawajy@deakin.edu.au

Corresponding Author: Mohammad Shojafar

fit) finds solutions in the shortest time, but usually, these reported solutions are not close to optimal [11]. Besides, it ignores the communications between VMs [12]. Meta-heuristic based method such as genetic algorithms [13], ant colony optimization (ACO) [14] and ant mating optimization [15] are often utilized to search in the vast search space of the possible solutions. Finally, these different methods can be combined to provide more robust techniques [16]. These algorithms can reduce the energy consumption within data centers to a certain extent. However, they are not suitable for the variable load. Besides, during VM deployment, the SLA violations for the existing algorithms are still high and needed to be further reduced. Nevertheless, none of these approaches considers the QoS constraints to find a joint energy-aware solution for VMA considering SLAs and QoSs using an innovative threshold-based mechanism to tweak the traffics.

Our aim in this paper is to develop a new framework that mutually considers the energy of the PMs/VMs and manage the incoming traffic in the CDCs. Specifically, we address the following questions: *i)* Is it possible to provide an adaptive traffic-aware framework in the CDC dealing with network energy? (see Section 3.1 and Section 3.2.1) *ii)* How can we preserve a set of QoS constraints in the steering traffic of different IoT users? (see Section 3.2.1) And, *iii)* How can we come up threshold-aware solution in such an environment? (see Sections 3.2.2 and Section 3.2.4).

1.1 The goal of the paper and the contributions

The paper's main goal is to decrease energy consumption and ensure the high QoS within cloud data centers (CDCs). To accomplish this, we propose an energy-efficient VM allocation and deployment algorithm based on an adaptive energy-aware framework. Unlike other energy-aware algorithms that only consider energy consumption due to VM deployment's, the proposed algorithm considers VM provision's energy efficiency during VM allocation and deployment. The proposed algorithm can effectively deal with variable load and maintain low energy consumption and SLA violation. All in all, our main contributions can be summarised as:

(1) Proposal of the adaptive four thresholds energy-aware framework that can effectively address the variable load named *AFED*;
(2) Presentation of a novel energy-efficient VM provision policy that considers both the energy consumption and SLA violation during the process of VM allocation and placement named *AFED-EF*;
(3) Carrying out a series of experiments using real-world workload and verifying the correctness and effectiveness of the proposed algorithm.

Compared with other energy-aware algorithms, the proposed AFED-EF can effectively handle the problem of load fluctuation by dividing the servers in CDCs into five classes and achieving proper VMs migration. Besides, different from other energy-aware algorithms that mainly focus on the energy consumption in CDCs, AFED-EF takes both the energy consumption and SLA violation into account during VM allocation and placement. A series of experimental results show that AFED-EF outperforms other energy-saving algorithms in energy consumption, SLA violation, and energy efficiency.

1.2 Organization

We scaffold the paper as the following sections. In Section 2, we present the related work. Section 3 proposes an adaptive four-threshold energy-aware framework for VM deployment. The performance evaluation is explained in Section 4. Finally, Section 5 concludes the paper and outlines areas for future research.

2 Related Work

To achieve much high energy efficiency of the green computing in CDCs, some scholars explore the problem and put forward some solutions. The solutions can be divided into three categories (C_1,C_2,C_3), that is *energy-aware IoT-based techniques* (C_1), *energy consumption model on CDC* (C_2), and *VM deployment/placement techniques on CDC* (C_3) that are reported in Section 2.1, 2.2, and 2.3, respectively.

2.1 Energy-aware IoT-based techniques

This category's main idea is to utilize the load characteristics and VM consolidation technology to achieve more energy saving. In this category, we survey the energy-aware techniques [17]–[25] used IoT applications try to save the energy within a CDC. Specifically, in [17], Cao and Dong raised a novel heuristic framework for VM consolidation to reduce energy consumption with a data center. Under the framework, authors develop a decision algorithm and minimize energy consumption and maximize the utilization rate. Similarly, in [18], Beloglazov and Buyya designed a two-threshold energy-aware framework. Its main idea is to set two thresholds to ensure that all CPU utilization of servers within a data center should fall between these two thresholds. To further saving energy consumption, in our previous work [19], a novel three thresholds framework was proposed used for VM deployment to achieve energy saving within data centers. Some of them use the *weighted sum* [20] and *normalization* [21], [22] approaches to obtain a single optimal solution for addressing the energy-aware IoT-based techniques in CDC. Recently, the work [23] introduces a new priority, power and traffic-aware VM placement algorithm that aim to minimize the energy usage of the servers of incoming IoT traffics. Also, the authors [24] designs an energy-efficient mechanism inspired by reinforcement algorithm adopt resource management over the VM resources associated with the servers in CDC. Finally, the authors in [25] design a secure method that controls the effects of IoT device demands over the network. Our method presented in this paper comparing this category papers not only monitors the VM provisioning to minimize the energy usage over the servers efficiently but also utilizes some intrinsic thresholds that could control the data centre's behaviour.

2.2 Energy Consumption model on CDC

This category includes several energy models applied on CDCs such as [26]–[29]. An energy-aware algorithm is closely related to the energy consumption model, and any

energy-saving algorithm is dependent on a certain power model. The power model's accuracy directly reflects the pros and cons of an energy-saving algorithm. Precisely, in [26], the authors design a novel energy consumption model by leveraging the deep learning technology. Their model takes 12 energy-related features into account and leverages deep neural network architecture to build an energy consumption model. In Reference [27], based on the combination of PCA (principal component analysis) and regression methods, the authors design a power model by considering multiple components (CPU, memory, disk, and network) of servers within a cloud data center. Their experimental results point out that the proposed model can achieve more than 95% prediction accuracy. In [28], the authors introduce several power models, including mixed load and I/O-intensive load. Recently, the authors in [29] introduce an energy-efficient task priority model to develop a fairness task scheduling algorithm in the CDC. Nevertheless, this paper's proposed model mainly addresses the energy model considered threshold-based provisioning policy and validated through a real-world workload.

2.3 VM deployment/placement algorithms on CDC

This category pays close attention to VM deployment algorithms applied in CDCs for green computing. Prior work such as [30]–[37] considered the energy consumption during the VM deployment despite ignored the SLA violation. The SLA violation and energy consumption factor are the same important for the cloud users, which obtain the cloud services through virtualization technology. Specifically, in [30], to achieve further energy saving, Fu and Zhou designed a novel VM selection strategy and placement method, which considered the degree of resource satisfaction. To meet the rapid increasing in demand for computational power, the paper [31] raised a novel resource allocation policy based on resource utilization of VMs. Their results confirmed that the proposed policy could reach a balance between QoS and energy consumption within a cloud data center. To achieve the maximum energy saving within a multimedia cloud computing, Han et al. [32] put forward a remaining utilization-aware algorithm for VM deployment. Its main idea is to find proper servers to shut down for energy saving. In [33], Beloglazov and Buyya presented an adaptive algorithm for energy-efficient and energy-consumption in CDCs. To further achieve the high energy efficiency, we can consider the VM deployment problem as "constrained programming problem " [34] and "integer programming problem " [35], [36]. For example, in [34], Dong et al. proposed a two-stage VM deployment scheme to strike a balance between "energy efficiency" and "network performance" in CDCs. In [35], a method based on game theory is applied to estimate the price of virtual resources and develop an integer linear programming for the VM system. To improve the energy efficiency of VM deployment, in [36], the authors mapped the VMs deployment problem into the integer linear programming problem under the condition of considering the communication between VMs. This way, they maximize system resource utilization and minimize system overhead, thereby saving the system energy consumption and default rate. Finally, GRVMP [37] is a greedy randomized VM placement algorithm in CDC using multidimensional resources. GRVMP used the "power of two choices" model and places VMs on the more power-efficient servers to optimize CDC energy and resource demands. Table 1 summarizes these categories.

TABLE 1: Comparison between different algorithms (where a ✓ implies that the reference admits the property, and a × implies that the reference does not admit the property). ML/DL:= Machine/Deep Learning; Thr:= Threshold; Math.:= Mathematical Formulation.

Ref.	Energy-Aware	Thr-Aware	SLA Aware	Type	VM Allocation
C_1					
[17]	✓	×	✓	Heuristic	×
[18]	✓	✓	✓	Heuristic	✓
[19]	✓	✓	×	Heuristic	×
[20]	✓	×	×	Math.	✓
[23]	✓	×	✓	Math. Heuristic	✓
C_2					
[26]	✓	✓	×	DL	×
[27]	✓	×	×	ML	×
[28]	✓	✓	×	Heuristic	×
[29]	✓	✓	×	Math. Heuristic	×
C_3					
[30]	✓	×	×	Heuristic	×
[31]	✓	✓	×	Math.	×
[32]	✓	✓	×	Heuristic	✓
[33]	✓	✓	×	Heuristic	✓
[36]	✓	✓	×	Math.	✓
[38]	✓	✓	✓	Math. Heuristic	✓
[39]	✓	✓	✓	Math. Heuristic	✓
[37]	✓	✓	✓	Math. Heuristic	✓
AFED-EF	✓	✓	✓	**Math. Heuristic**	✓

3 Adaptive Four-threshold energy-aware framework for VM deployment (AFED)

In here, we will explorer the system architecture (section 3.1), adaptive four-threshold framework for VM deployment (section 3.2). The framework's content involves an adaptive four thresholds framework, determination of four thresholds, energy-efficient VM allocation strategy, VM selection strategy, and energy-efficient VM deployment strategy. Table 2 lists the main symbols and notations used in the paper.

3.1 System architecture

Energy-efficient VM provision aims to reduce energy consumption while guaranteeing a high QoS defined by cloud users. Fig. 1 illustrates the system architecture for supporting energy-efficient service provision for IoT applications in a CDC. In this figure, the IoT devices connect to the Internet and transfer data demands through it. Internet is connecting to the Global Manager and Server manager components as an entry system to the CDC. These components are described as follows:

(1) **Global manager:** it obtains the information from the "server manager" and analyzes the collected information. Based on the obtained information feedback, it gives optimization commands.

TABLE 2: Main symbols and their meanings

Symbol	Description
	Server's Thresholds
T_L	Little loaded server threshold
T_N	Normally loaded server threshold
T_M	Medium loaded server threshold
T_H	heavy loaded server threshold
M	The number of servers
N	The number of VMs
T_S	The current CPU utilization of a server
S	Set of Servers
V	Set of VMs
c	Safety coefficient
T	CPU utilization dataset of a server for a period of time in the past
C	Cluster set where $\forall k \in K\ C = \{C_1, \ldots, C_k\}$
$MAD(C_j)$	Mean Absolute Deviation of cluster C_j
$MC[j]$	The j-th value of array MC
$IQR(MC)$	The interquartile range of array MC
EF	Total energy efficiency within a CDC
P_{total}	Total energy consumption consumed by servers
V_{SLA}	Total SLA violations
$SVTH$	SLA violations time per active server
$PDCM$	Performance degradation induced by excessive VMs migration
$R^{CPU}_{VM_i}$, $R^{mem}_{VM_i}$	Requested CPU and memory amount, respectively
$R^{disk}_{VM_i}$, $R^{bw}_{VM_i}$	Requested disk and bandwidth amount, respectively

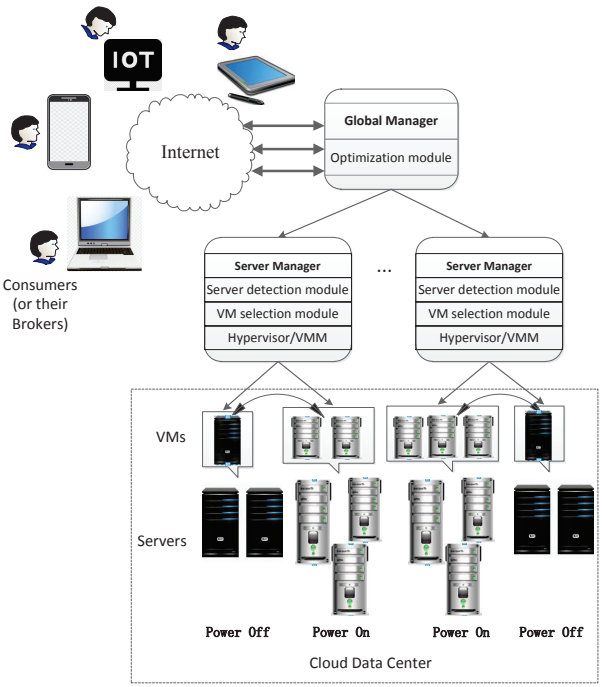


Fig. 1: System architecture

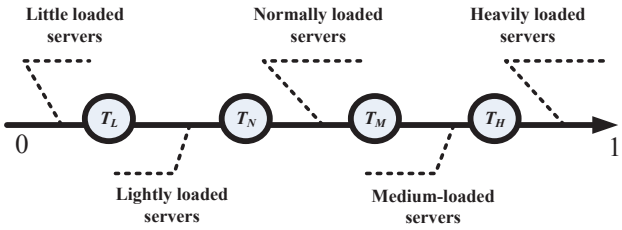


Fig. 2: Five classes servers within a data center

(2) **Server manager:** each server has a "server manager", and it is in charge of communication with the global manager and virtual machine monitor (VMM).
(3) **Server detection module:** it is responsible for monitoring the status of the server and determining whether the server is a "little loaded server" or "lightly loaded server" or "normally loaded server" or "medium-loaded server" or "heavily loaded server".
(4) **VM selection module:** when a server is judged as a "heavily loaded server", some VMs belonging to the server should be selected and migrated into other servers.
(5) **Hypervisor/VM Management (VMM):** VMM means the VM monitor, hypervisor component/VMM plays an essential role in CDC. It performs the VM placement and task allocation on the servers and manages the server consolidation (such as server ON/OFF).

3.2 Adaptive four-threshold energy-aware framework

In a CDC, there are plenty of servers that connect and provide various cloud services to cloud users. These servers within a data center have different resource utilization (For instance, CPU utilization, memory utilization, and disk utilization). As the servers' CPU utilization accounts for a large proportion of the energy consumption [19], [33] based on a load of servers. We divide the servers within a data center into *five* classes by setting *four* thresholds. The four thresholds are T_L, T_N, T_M, and T_H ($0 \leq T_L < T_N < T_M < T_H \leq 1$), and the five classes servers are "little loaded server", "lightly loaded server", "normally loaded server", "medium-loaded server", and "heavily loaded server", respectively. Denote T_S as the current CPU utilization of a server. We have the following definition:

(1) if $T_S \leq T_L$, the server is considered as *little loaded server*;
(2) if $T_L < T_S \leq T_N$, the server is regarded as *lightly loaded server*;
(3) if $T_N < T_S \leq T_M$, the server is treated as *normally loaded server*;
(4) if $T_M < T_S \leq T_H$, the server is viewed as *medium-loaded server*; and,
(5) if $T_S > T_H$, the server is seen as "*heavily loaded server*".

Fig. 2 shows the *five* classes serves within a data center.

To achieve more energy efficiency within data centers, we purpose a framework named *Adaptive four-threshold energy-aware framework for VM deployment (AFED)*. The flow chart of AFED presents in Fig. 3.

We give detail on the basic idea of AFED as below. First, AFED classifies the servers in a data centers into *five* classes by setting four thresholds T_L, T_N, T_M, and T_H that indicate little IoT application demands loaded on the server, lightly loaded server, normally loaded server, medium-loaded server, and heavily loaded server, respectively. Then, regarding each server within a data center, AFED checks the load status of the server. When the server is judged as *heavily loaded server*, to lower the SLA violations within a data center

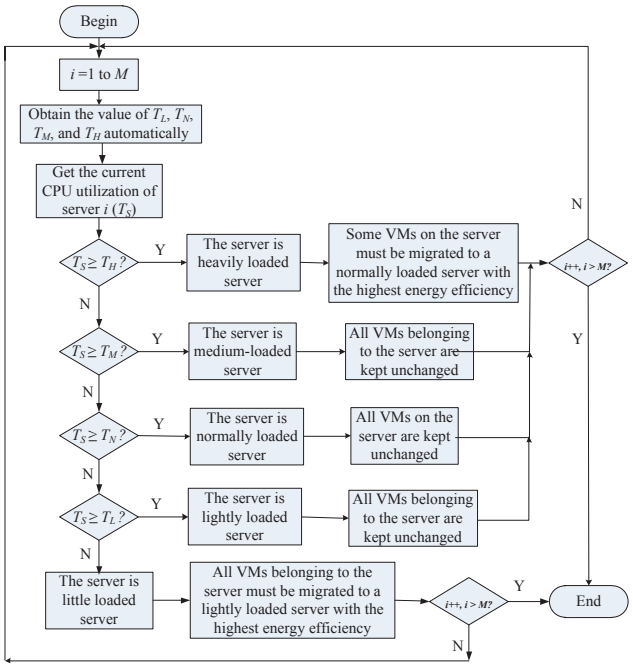


Fig. 3: The flowchart of adaptive four-threshold energy-aware framework for VM deployment

for sustainable computing, some VMs on the server must be migrated to a *normally loaded server* with the highest energy efficiency. Same wise, when the server is determined as *medium-loaded server*, to avoid excessive VM migration lead to high energy consumption and SLA, all VMs belonging to the server are kept unchanged. Also, when the server is decided as *normally loaded server*, all VMs on the server is kept unchanged. This is because there is no need to migrate any other VMs from the server with a normal load. When the server is judged as *lightly loaded server*, it means the server is in a state with reasonable load, and there is no need to migrate any other VMs from the server. When the server is determined as *little loaded server*, to save the energy consumption within a data center, all VMs belonging to the server must be migrated to a *lightly loaded server* with the highest energy efficiency. AFED repeats the previous process until all VMs in the data center are judged.

AFED reasonably divides the server within a data center based on its load and provides different migration policies to achieve a much higher energy efficiency of data centers and IoT apps' loads. However, regarding the AFED, some remaining issues need to be addressed. The first issue is how to obtain the value of T_L, T_N, T_M, and T_H automatically. This problem will be handled at the Section 3.2.1. According to the idea of AFED, when a server is determined as "heavily loaded server", some VMs belonging to the server must be migrated to other servers. The second issue is that there are a lot of VMs on the server, which VM should be migrated. This problem will be addressed at Section 3.2.4.

Based on the idea of AFED when a server accommodates the VMs that have been migrated from elsewhere, it should owns the highest energy efficiency. The third issue is how to decide the server with the highest energy efficiency. This question will be handled at Sections 3.2.2 and 3.2.3.

Algorithm 1 KMIR algorithm (K-Means-Mad-IQR)

Input: The past CPU utilization set $T = \{T_1, T_2, ..., T_p, ..., T_n\}(1 \leq p \leq n)$ and k
Output: Four thresholds T_L, T_N, T_M, and T_H

```
for i = 1 to M do
    Obtain the past CPU utilization set T = {T_1, T_2, ..., T_n} and parameter k;
    Get the clusters C = {C_1, C_2, ..., C_k} by using the K-Means algorithm, that is C=KMeans(T, k) ;
    //parameter k is the size of Cluster C
    for j = 1 to k do
        MC[j] ← Mad(C_j);
        // using the eq. (1) to calculate
    endfor
    Calculate IQR(MC) by using eq. (2);
    Obtain the four thresholds T_L, T_N, T_M, and T_H according to eqs. (3)-(6);
endfor
return T_L, T_N, T_M, and T_H.
//Return the four thresholds.
```

3.2.1 Determination of four thresholds

As we discussed recently, this section is dealing with the first problem we have issued.

To handle the previous problem, we develop a new algorithm named *K-Means-Mad-IQR* (KMIR) to determine the four thresholds (T_L, T_N, T_M, and T_H) of AFED automatically. Let S be a set of servers within a data center, that is $S = \{S_1, S_2, \ldots, S_i, \ldots, S_M\}(1 \leq i \leq M)$, and V is a set of VMs, that is $V = \{VM_1, VM_2, \ldots, VM_j, \ldots, VM_N\}(1 \leq j \leq N)$. Regarding a server S_i ($S_i \in S$) within a data center at time n, for a period of time in the past, its CPU utilization set can be can be expressed as $T = \{T_1, T_2, \ldots, T_p, \ldots, T_n\}(1 \leq p \leq n)$. The pseudo-code of KMIR is characterized as follows:

KMIR takes servers' past CPU utilization set $T = \{T_1, T_2, \ldots, T_p, \ldots, T_n\}$ as inputs and outputs T_L, T_N, T_M, and T_H. First, KMIR adopts K-Means clustering technology to divide T into k clusters (i.e., the value of k can be obtain by adopting the empirical method) such that:

- C=$\{C_1, C_2, C_3, \ldots, C_q, \ldots, C_k\}$ where $(1 \leq q \leq k)$
- $C_q = \{T_{R_{q-1}+1}, T_{R_{q-1}+2}, \ldots, T_{R_q} \in T\}$
- $0 = R_0 < R_1 < R_2 < R_3 < \ldots < R_q = n$ and
- $R_q \bigcap R_e = O$ where $(1 \leq q, e \leq k)$

Then, after getting the clusters set C=$\{C_1, C_2, C_3, \ldots, C_q, \ldots, C_k\}$, we calculate the Mean Absolute Deviation (MAD) of cluster C_q $(1 \leq q \leq k)$. The definition of MAD can be defined as

$$MAD(C_q) = median(|T_{R_{q-1}+i} - median(C_q)|), \quad (1)$$

where function $median(C_q)$ is to return the median value of cluster C_q. Next we calculate the Interquartile Range (IQR) for the array MC. The IQR can be obtained using the following equation:

$$IQR(MC) = Q_3(MC) - Q_1(MC), \quad (2)$$

where $Q_3(MC)$ is the 75-th percentile of array MC, whereas $Q_1(MC)$ is the 25-th percentile of array MC. Finally, the T_L, T_N, T_M, and T_H of KMIR are defined as

$$T_H = (1 - c \times IQR), \quad (3)$$

$$T_M = 0.9(1 - c \times IQR), \tag{4}$$

$$T_N = 0.8(1 - c \times IQR), \tag{5}$$

$$T_L = 0.5(1 - c \times IQR), \tag{6}$$

where parameter c means safety coefficient, and its value reflects how the CDC consolidates VMs. In general, the greater the value c is, the more frequently that the VMs are merged, which also means less energy consumption and higher SLA violations. On the contrary, if the parameter c is set to a small value, it means the less frequently VMs consolidation, the higher energy consumption, and the lower SLA violations. The CDC use T_M, T_N and T_L thresholds with accompanying numeric coefficient values like 0.9, 0.8, and 0.5 respectively. These coefficients indicate the weight of each threshold to preserve the energy usage in the CDC.

3.2.2 Energy-efficient VM allocation strategy

VM allocation strategy plays a fundamental role in reducing the energy consumption and SLA violations within a CDC. Unlike other energy-aware algorithms only focusing on minimizing the energy consumption within a data center, AFED optimizes both energy consumption and SLA violations during VM allocation and placement. Assume EF is the total energy efficiency (i.e., energy consumption and SLA violations) within a CDC, P_{total} is the total energy consumption consumed by servers within a data center, and V_{SLA} is the total SLA violations. Energy efficiency is formally expressed as below:

$$EF = \frac{1}{P_{total} \times V_{SLA}}. \tag{7}$$

The formal description of energy-efficient VM allocation strategy can be concluded as

$$\begin{aligned} &Maximize \left\{ EF = \frac{1}{P_{total} \times V_{SLA}} \right\} \\ &Subject\ \ to \\ &P_{total} = \sum_{i=1}^{M} x_i \cdot P_i, \\ &\sum_{i=1}^{N} R_{VM_i}^{CPU} = \sum_{j=1}^{M} x_j \cdot CPU_j, \\ &\sum_{i=1}^{N} R_{VM_i}^{mem} = \sum_{j=1}^{M} x_j \cdot Mem_j, \\ &\sum_{i=1}^{N} R_{VM_i}^{disk} = \sum_{j=1}^{M} x_j \cdot D_j, \\ &\sum_{i=1}^{N} R_{VM_i}^{bw} = \sum_{j=1}^{M} x_j \cdot B_j, \end{aligned} \tag{8}$$

where x_i is the number of type i server, P_i is the energy consumption of type i server, M means the number of servers within a data center, N represents the number of VMs, $R_{VM_i}^{CPU}$, $R_{VM_i}^{mem}$, $R_{VM_i}^{disk}$, and $R_{VM_i}^{bw}$ are the requested CPU amount, memory amount, disk amount, and bandwidth amount of VM i, respectively. The para metres CPU_j, Mem_j, D_j, and B_j are the CPU amount, the memory amount, the disk amount, and the bandwidth amount of a server j, respectively.

Algorithm 2 AFED

Input: T_L, T_N, T_M, and T_H
Output: migrateMap
1: serverList ← getServerlist( );
// getServerlist( ) is used to get servers list
2: For server to serverlist do
3: T_S ← server.getUtilizationOfCPU();
//obtain the CPU utilization of the server;
4: If ($T_S \leq T_L$) Then
5: vmList ← getAllVmToMigrate(server);
//get all VMs of the server
6: OneMap ← getVMPlacement(vmlist);
7: migrateMap.add(OneMap);
8: Else If ($T_S \leq T_H$) Then
9: continue;
10: Else If ($T_S > T_H$) Then
11: vmList ← getVmToMigrate(server);
//get some VMs of the server to migrate
12: TwoMap ← getVMPlacement(vmlist);
13: migrateMap.add(TwoMap);
14: End If
15: End For
16: **return** migrateMap.

3.2.3 VM selection strategy

Herein, we explain the VM migration, VM selection strategies that aim to reduce the potential energy consumption and SLA violations.

We adjusted the MMT policy [33] to choose a VM from the heavily loaded server. The main idea of MMT policy is to select a VM with the least migration time compared with other VMs belonging to the server. Let VM_j is the VM set belonging to the server j, $Mem(r)$ is the memory amount that is being utilized by VM r, BW_j is available bandwidth for server j. MMT selects a VM v that meets the following requirement:

$$v \in VM_j \ and \ \forall r \in VM_j, \frac{Mem(v)}{BW_j} \leq \frac{Mem(r)}{BW_j}. \tag{9}$$

Algorithm 2 presents the pseudo-code of AFED. The main idea of AFED can be summarized as follows: AFED takes the *four* thresholds (T_L, T_N, T_M, and T_H) as inputs and outputs the VM migration map. First, AFED obtains the server list (line 1). Then, we gain the CPU utilization of the current server denoted by T_S (lines 2-3). After that, we should make a comparison with the four thresholds. If ($T_S \leq T_L$), all VMs belong to the server should migrate to other place (lines 4-7); otherwise, if ($T_S \leq T_H$), all VMs on the server are kept unchanged (lines 8-9); otherwise, if ($T_S > T_H$), some VMs on the server need to be migrated to other place to avoid potential SLA violations (lines 10-14). At last, the algorithm returns the migration map (lines 15-16).

3.2.4 Energy-efficient VM deployment strategy

In this part, we mainly focus on management and decide on the server action when facing the highest energy efficiency. To deal with this problem, we propose an energy-efficient VM deployment strategy based on AFED framework called *AFED-EF*. Unlike other energy-aware algorithms, AFED-EF optimizes energy consumption and energy efficiency within

Algorithm 3 AFED-EF

Input: T_L, T_N, T_M, and T_H
Output: VMs placement
1: vmList ← getVmlist( ); // Get VM list
2: serverListOne ← getLightlyLoadedServerlist( );
3: serverListTwo ← getNormallyLoadedServerlist( );
4: vmlist.sortByCPUDecreasing (); // Sorting VMs
5: For vm to vmlist() do
6: minPower ← Min_VALUE;
//assign a maximum value
7: allocatedServer ← Null;
8: For server to serverListOne or serverListTwo do
9: If (server.isSuitableForVm(vm)) Then
10: PowerServer ← server.getPower();
//obtain the power of the server
11: PowerServerAfterVM ← getPower(server, vm);
12: PowerDiff ← PowerServerAfterVM-PowerServer;
13: SLA ← server.getSLA();
14: SLAAfterAllocation ← getSLA(server, vm);
15: SLADiff ← SLAAfterAllocation-SLA;
16: EF ← 1/(PowerDiff∗SLADiff);
17: If (EF>minPower) Then // Get the maximum
18: minPower ← EF;
19: allocatedServer ← server;
20: End if
21: End If
22: End for
23: allocate the VM to server and achieve placement
24: End for
25: **return** VMs placement

a data center. The pseudo-code of AFED-EF is shown in Algorithm 3.

The main idea of AFED-EF can be summarized as follows: AFED-EF takes the *four* thresholds (T_L, T_N, T_M, and T_H) as inputs and outputs the VM placement. First, AFED-EF obtains the VM list and server list (lines 1-3). Then, VMs are ranked in descending order according to its CPU utilization (line 4). For each VM in VM list and candidate server (lines 5-8), AFED-EF calculates the energy consumption difference (lines 10-12), SLA difference (lines 13-15), and energy efficiency difference (line 16) before and after placement of the VM. After that, AFED-EF selects a server with high energy efficiency to achieve VM placement (lines 17-20) and return the final results (line 23).

Theorem 1. *The time complexity of AFED-EF is $O(M \times N)$, M is the number of servers, N is the number of VMs within a data center.*

Proof. AFED-EF comprises *three* parts. The first part is the AFED algorithm, and its time complexity is $O(M)$. The second part is the KMIR algorithm, and its time complexity is $O(k \times M)$, where k is the number of clusters; The third part is the energy-efficient VM allocation and placement, and its time complexity is $O(M \times N)$. Therefore, the time complexity of AFED-EF is $O(M + k \times M + M \times N)$, that is $O(M \times N)$. Theorem 1 has been proven.

4 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithms; AFED and AFED-EF. We first explain the experimental setup (Section 4.1). Then, give detail on the IoT workload data traffic model (Section 4.2). Afterwards, we list the energy consumption and efficiency model that are reported in Section 4.3. In the end, we present the results (Section 4.4).

4.1 Experimental setup

energy consumption, number of VM migration, SLA violation, SVTH (Section 4.4), PDCM (Section 4.4), and energy efficiency. The experiments are composed of *three* parts: (1) Decide the optimal parameter value of c; (2) Test the performance of AFED algorithm; (3) Test the performance of AFED-EF algorithm. The program is coded on CloudSim toolkit [18], [33], [40]. We create a CDC, which is composed of 800 servers. There are two types of servers (HP Proliant G4 and G5) in the CDC. We list the server configurations in Table 3. The Java program implementation of AFED-EF is available in [41].

TABLE 3: Configuration of servers

Servers	Type	Frequency	Cores	RAM
HP G4	Intel Xeon 3040	1.86 (GHz)	Double	4 (GB)
HP G5	Intel Xeon 3075	2.66 (GHz)	Double	4 (GB)

Similar to Amazon EC2, there are four types of VMs in the CDC. Hence, we adopt them and report their configurations in Table 4.

TABLE 4: Configuration of VMs

VM type	CPU (MIPS)	RAM (GB)
Micro instance	500	0.61
Small instance	1000	1.7
Large instance	2000	3.75
High performance instance	2500	0.85

4.2 Workload data

It is essential to leverage the real workload to evaluate the performance of AFED-EF algorithm. In this experiment, we used real workload data to make a comparison. The real workload trace comes from a CoMon project [33], which is used to monitor infrastructure and IoT data. The collected trace data is derived from 500 places and more than 1000 VMs' resource utilization. Table 5 displays the characteristics of workload data. In Table 5, "Mean" represents the average value, "Quartile 1" (Q1) means the 25-th percentile of the array, "Quartile 3" (Q3) is the 75-th percentile of the array.

TABLE 5: Workload specifications

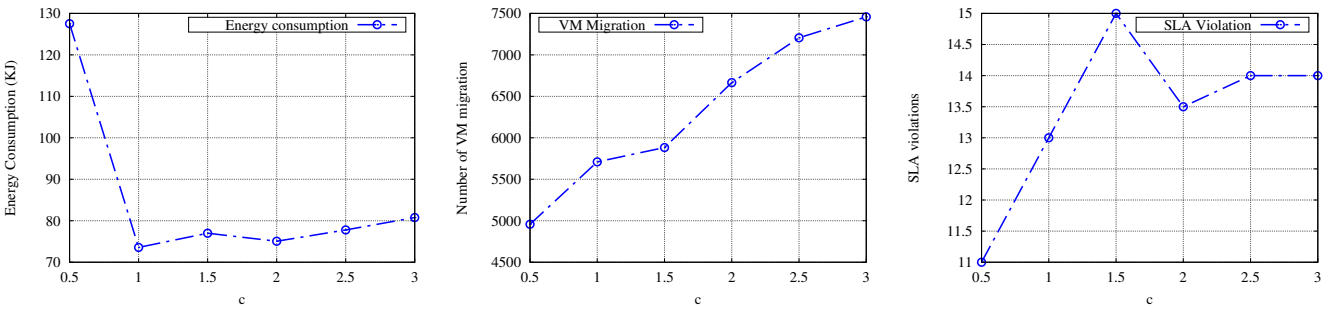
Date	VM Numbers	Average	Q1	Q3
3-Mar-11	1052	12.31%	2%	15%
6-Mar-11	898	11.44%	2%	13%
...	...	...	...	...
20-Apr-11	1033	10.43%	2%	12%

In the following experiments, we choose "3-Mar-11" as dataset and compare their traffic values among the proposed algorithms and state-of-the-art. As for "3-Mar-11" dataset, the number of VMs is 1052, mean value is 12.31%, "Quartile 1" (Q1) is 2%, and "Quartile 3" (Q3) is 15%.

4.3 Energy consumption and efficiency models

Energy consumption model lays a foundation for energy-aware algorithms. In this paper, we leverage the real energy consumption data which is come from the SPECpower benchmark[1]. Table 6 shows the energy consumption data

1. http://www.spec.org/powerssj2008/



(a) Energy Consumption. (b) Number of VM migration. (c) SLA violations.

Fig. 4: The AFED-EF variation results using c values for energy consumption (4a), VM migration (4b) and SLA violations (4c).

TABLE 6: Result of energy consumption on real servers.

CPU Utilization (%)	Power Consumption (W)	
	HP G4	HP G5
0	86	93.7
10	89.4	97
20	92.6	101
30	96	105
40	99.5	110
50	102	116
60	106	121
70	108	125
80	112	129
90	114	133
100	117	135

of servers under different resource utilization, i.e., CPU utilization as referred in [19].

In Section 3.2.2, we presented the formal description of energy efficiency (see eq. (7)). Based on this definition, energy efficiency is related to energy consumption and SLA violations. SLA violations are associated with two factors. First, SLA violation time per active server (SVTH) [33], and performance degradation caused by VM migration (PDCM) [33]. As the two methods are equally effective, we use the two methods to define the SLA violation (denoted by V_{SLA}).

$$V_{SLA} = SVTH \times PDCM. \quad (10)$$

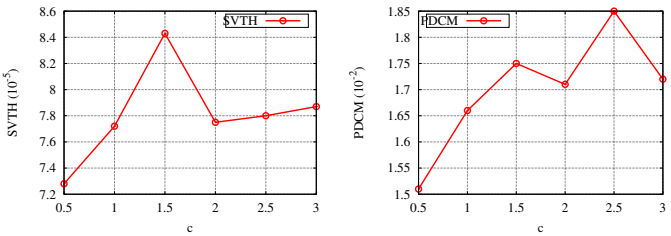
4.4 Experimental results

In the rest of this section, we will test the performance of the AFED-EF algorithm by performing a series of experiments. To embody the advantages of the proposed algorithm, the algorithms KAI-1.0 [19], THR-0.8 [33], MAD-2.5 [33], and IQR-1.5 [33] are selected to make a comparison in terms of energy efficiency, energy consumption, SLA violation, SVTH, PDCM, and a number of VM migration. For VM selection policy, the MMS is chosen to make a comparison.

4.4.1 *The optimal value of c for AFED-EF*

In the first set of experiment, we test the AFED-EF results for various metrics with the aid of varying the value of parameter c. We change the c value from 0.5 to 3.0 increased by 0.5. The key outcome of this experiment helps us to respond to how to decide the optimal value of parameter c for AFED-EF.

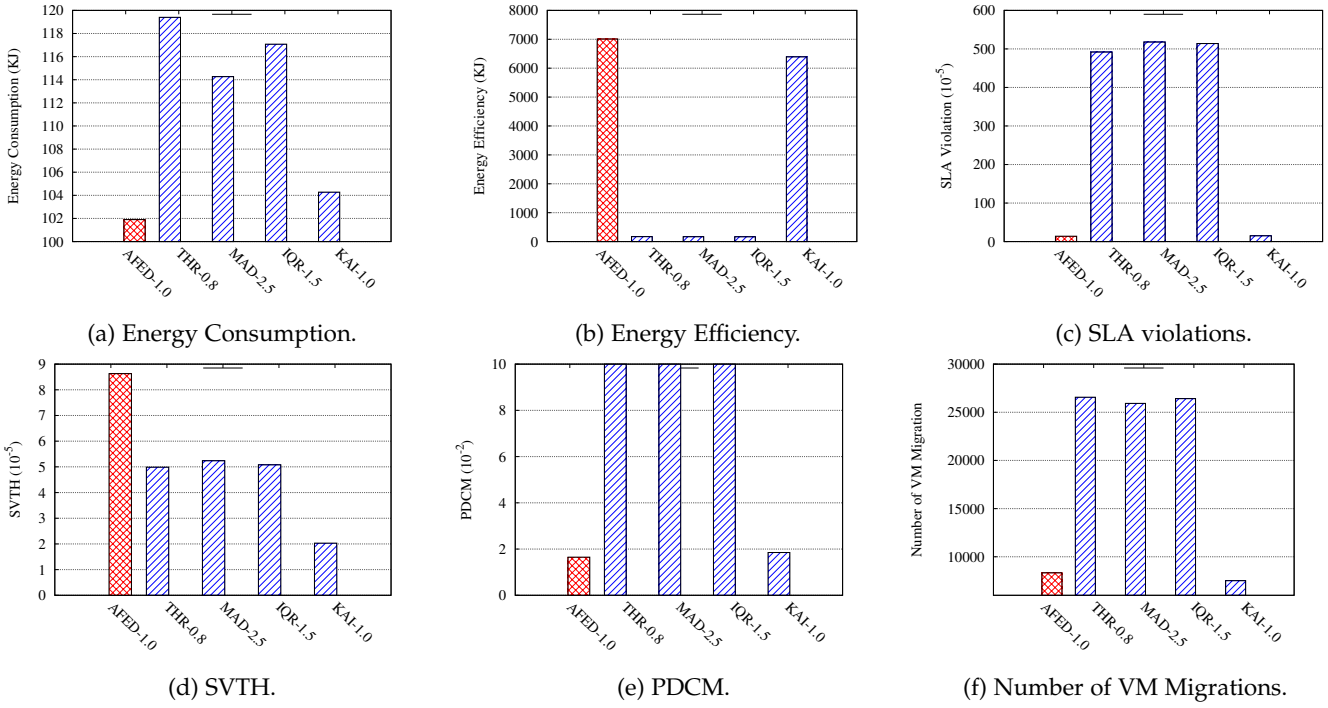
Fig. 4 present the energy consumption, VM migration and SLA violation of AFED-EF algorithm for various c values which are within $c = \{0.5, 1, 1.5, 2, 2.5, 3\}$. Focusing on Fig. 4a, we can confirm that the energy consumption decreases with an increase of c. However, when parameter c increases to a certain extent, the energy consumption increases instead. This can be explained that a proper value of parameter c can bring a reasonable VMs migration. Thus it brings more energy consumption saving. In Fig. 4b the number of VM migration growths with the increasing of parameter value c. Because the greater value of c, leads to the more frequently the VMs merge and it increases the number of VM migration. Finally, Fig. 4c illustrates the SLA violation under different value of c. This figure points out that when parameter $c = 0.5$ or $c = 1.0$, the SLA violation of AFED-EF is at a low level. In the next set of simulation, we will test various SLS metrics; SVTH and PDCM. Fig. 5



(a) SVTH. (b) PDCM.

Fig. 5: The AFED-EF variation results based on c values for different SLA violation metrics.

shows the results of AFED-EF algorithm for various c values on different SLA metrics. According to Fig. 5a describes that when the parameter $c = 0.5$ or $c = 1.0$, the SVTH of AFED-EF is at a low level. The reason is that a proper value of c can bring a lower SVTH. Regarding Fig. 5b confirms that when $c = 0.5$ or $c = 1.0$, the PDCM of AFED-EF is at a lowest level. The reason could be explained by the fact that a proper value c can bring a reasonable VM migration, thus leading to less PDCM. Based on the results of Figs. 4 and 5, We can conclude that when we consider $c = 1$ AFED-EF has a better performance. Therefore, in the following experiments, we keep $c = 1.0$ as the optimal value along with AFED-EF.



(a) Energy Consumption. (b) Energy Efficiency. (c) SLA violations.

(d) SVTH. (e) PDCM. (f) Number of VM Migrations.

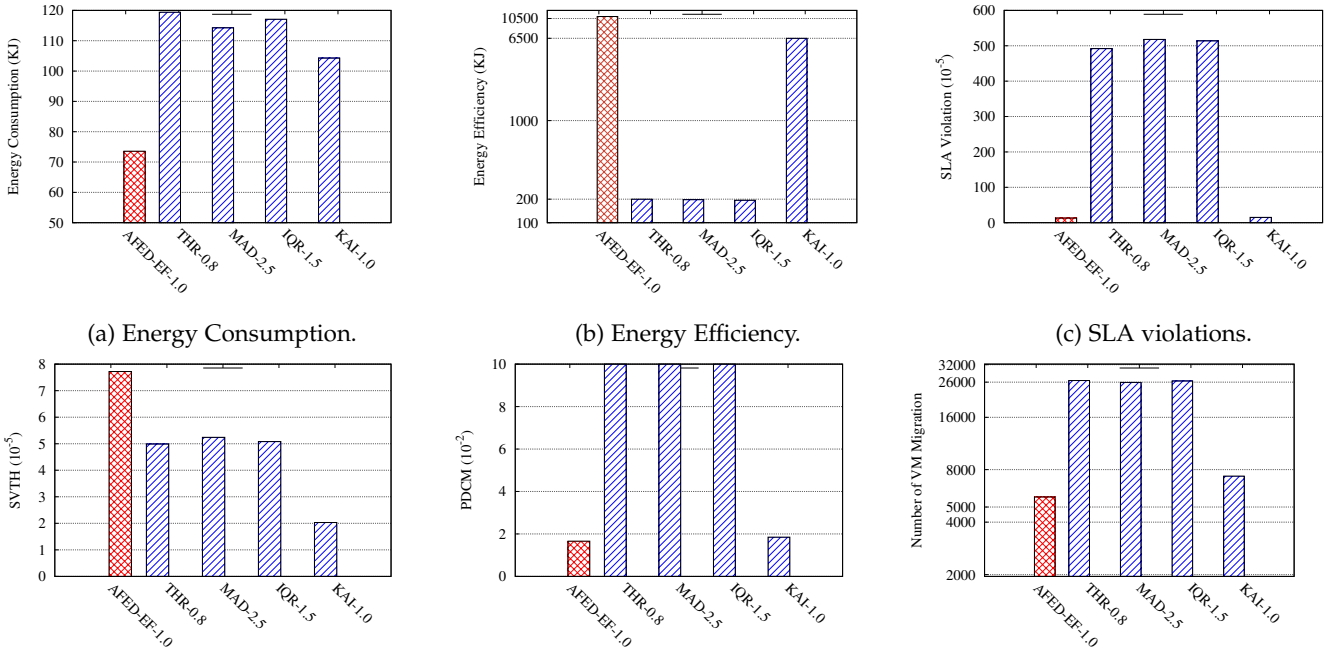
Fig. 6: The AFED results when comparing with THR, MAD, IQR, and KAI approaches.

4.4.2 The evaluation and analysis of AFED

In the next experiment, we test the performance of the proposed algorithm, AFED, without applying energy-efficient VM allocation and placement algorithm. Figs. 6a to 6f display the energy efficiency, energy consumption, SLA violation, SVTH, PDCM, and number of VM migration, respectively. In particular, Fig. 6b indicates the energy efficiency comparison for the five algorithms. The experimental results show that AFED-1.0 leads to the best performance, KAI-1.0 the second, THR-0.8 the third, MAD-2.5 the fourth, and IQR-1.5 the last one. It is because as presented in eq. (7), the value of energy consumption is directly related to energy consumption and SLA violation. Hence, it affects energy consumption, and our method consumes less energy compared to other solutions. Fig. 6a points out that AFED-1.0 leads to the best performance, KAI-1.0 the second, MAD-2.5 the third, IQR-1.5 the fourth, and THR-0.8 the fifth. The reason behind this is that AFED-1.0 adopts the four thresholds framework and reducing energy consumption. Besides, Fig. 6c reveals the SLA violations comparison for the five algorithms, and results illustrate that AFED-1.0 leads to the best performance. It confirms that AFED-1.0 adopts the four thresholds framework to consolidate VM effectively, thus decreasing the SLA violations. Fig. 6d and 6e present the results for SVTH and PDCM metrics for the five algorithms.Both SVTH and PDCM metrics are related to SLA violation (eq. (10)). Thus, Focusing on PDCM, AFED-1.0 has a better performance than other algorithms. Focusing on SVTH, AFED-1.0 has a normal performance. The reason is that AFED-1.0 has to consolidate VM aggressively to reduce energy consumption, and it rises the SVTH values. Finally, Fig.6f describes the number of VM migration for the five algorithms. This figure confirms that excessive VM migration can bring performance degradation. Similarly, less VM migration can also bring high energy consumption. Therefore, keeping an amount of VMs migration is necessary for the energy efficiency of any VM placement algorithms.

4.4.3 The evaluation and analysis of AFED-EF

In this experiment, we validate the performance of our AFED-EF which is the joint of energy-efficient VM allocation and placement strategies. Figs. 7a to 7f show the energy efficiency, energy consumption, SLA violation, SVTH, PDCM, and number of VM migration, respectively. Specifically, Fig. 7a indicates the energy consumption of five algorithms including our AFED-EF algrithm. From this figure we can understand that AFED-EF-1.0 can adopt the four thresholds framework and energy-efficient VM allocation and deployment policy to decrease the energy consumption. Focusing on energy efficiency as reported in Fig. 7b, our algorithm can exponentially save the energy of the network since it is SLA sensitive and could provide a reasonable solution for the incoming traffic to the cloud system. In Fig. 7c we present the ratio of SLA violation among the algorithms. It is obvious that AFED-EF algorithm has very low SLA violation when it compares among other cases and its violation rate is 13 while KAI is 15. The reason is that AFED-EF adopts the four thresholds framework and energy-efficient VM allocation and deployment policy to consolidate VM effectively, thus reducing the SLA violations. Focusing on Figs. 7d and 7e which are the metrics of SLA violation, our algorithm provides an acceptable and reasonable values compared to the other methods. It confirms that AFED-EF has to consolidate VM aggressively to reduce the energy consumption, and it also gives rise to high SVTH but this value is much less than the same value for other algorithms. At last, Figs. 7f shows the number of VM migration for the five



(a) Energy Consumption. (b) Energy Efficiency. (c) SLA violations.

(d) SVTH. (e) PDCM. (f) Number of VM Migrations.

Fig. 7: The AFED-EF results when comparing with THR, MAD, IQR, and KAI approaches.

algorithms and enables excessive VM migration can bring performance degradation. Note that less VM migration can bring the high energy consumption. Therefore, keeping an amount of VMs migration is necessary for energy efficiency of any VM placement algorithms. AFED-EF has lower VM migration when compares to other methods and this rate is exponentially lower than others.

4.4.4 *Discussion on limitation*

Although the present work provides an efficient solution for VM allocation for the huge processing request gathering from IoT applications, there are still several aspects concerning this problem. First, the proposed method can be extended to include the allocation of network bandwidth to provide high QoS for critical applications and avoid traffic congestion for normal ones. Second, in this work, we assume uniform random traffic among VMs of a requested application. However, to improve it, we can incorporate learning mechanisms for traffic prediction among VMs of requested applications in our future work. Third, this work is focused on the VM allocation and SLA violation management. However, to improve the energy efficiency of a CDC more and more, an efficient micro-service IoT demand request phase also can be added to this work. Finally, the proposed method cannot be directly applied in Fog/Edge-Cloud computing environments [42], [43]. To cope with such environments, we are planing to extend our proposed method and compare it with different learning methods such as those presented in [44], [45].

5 Conclusion and Future Directions

Prolonging IoT devices' lifetime and enhancing the quality of processing of IoT applications on the back-end servers are quite challenging problem. To effectively address the variable load and achieving the maximum energy efficiency for IoT applications in a CDC, this paper proposes a novel VMallocation and placement algorithm, which we referred to as *AFED-EF*. The results of the experiments show that, in contrast to KAI-1.0 and AFED-EF-1.0, improves energy efficiency by more than 64%. In terms of energy consumption and SLA violations, AFED-EF reduces up to 29% and 13% compared to KAI-1.0, AFED-EF-1.0, respectively. AFED-EF can deal with the variable load problem in cloud data centre and reduce both the energy consumption and SLA violations for IoT applications. The deployment of AFED-EF on different cloud-based platforms can significantly reduce the cost of energy consumption and SLA violations, and thus improves return on investment (ROI) for the cloud operators. Therefore, We plan to implement AFED-EF as an extension of VM optimization and performance evaluation within the OpenStack Cloud platform in future work. This work will also incorporate the microservices IoT demands running on the cloud-based back-end servers and gateways to specify the heterogeneous Spatio-temporal IoT requests. This helps the designed algorithm to dynamically control the servers and trigger actuators to migrate some portions of the demands to nearby servers to save time and energy and avoid violation of the users' SLA.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 61772088), and in part by the Hunan Province Key Laboratory of Industrial Internet Technology and Security (nos. 2019TP1011). Mohammad Shojafar is supported by Marie Curie Global Fellowship funded by European Commission with grant agreement MSCA-IF-GF-839255.

REFERENCES

[1] X. Liu, B. Cheng, and S. Wang, "Availability-aware and energy-efficient virtual cluster allocation based on multi-objective optimization in cloud datacenters," *IEEE Transactions on Network and Service Management*, 2020.

[2] Y. Al-Dhuraibi, F. Paraiso, N. Djarallah, and P. Merle, "Elasticity in cloud computing: State of the art and research challenges," *IEEE Transactions on Services Computing*, vol. 11, no. 99, pp. 430–447, 2018.

[3] H. Wu, Y. Sun, and K. Wolter, "Energy-efficient decision making for mobile cloud offloading," *IEEE Transactions on Cloud Computing*, 2018.

[4] C. Liu, K. Li, and K. Li, "A game approach to multi-servers load balancing with load-dependent server availability consideration," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2018.

[5] M. R. Rahimi, N. Venkatasubramanian, S. Mehrotra, and A. V. Vasilakos, "On optimal and fair service allocation in mobile cloud computing," *Computer ence*, vol. 6, pp. 815–828, 2018.

[6] I. Mavridis and H. Karatza, "Combining containers and virtual machines to enhance isolation and extend functionality on cloud computing," *Future Generation Computer Systems*, vol. 94, pp. 674–696, 2019.

[7] S. Sotiriadis, N. Bessis, and R. Buyya, "Self managed virtual machine scheduling in cloud systems," *Information Sciences*, vol. 433, pp. 381–400, 2018.

[8] C. M. McDermott, "Optimizing virtual machine memory sizing for cloud-scale application deployments," Apr. 17 2018, uS Patent 9,946,573.

[9] T. Verbelen, T. Stevens, F. De Turck, and B. Dhoedt, "Graph partitioning algorithms for optimizing software deployment in mobile cloud computing," *Future Generation Computer Systems*, vol. 29, no. 2, pp. 451–459, 2013.

[10] H. Yu, J. Yang, H. Wang, and H. Zhang, "Towards predictable performance via two-layer bandwidth allocation in cloud datacenter," *Journal of Parallel and Distributed Computing*, vol. 126, pp. 34–47, 2019.

[11] P. H. Raj, P. R. Kumar, and P. Jelciana, "Load balancing in mobile cloud computing using bin packing's first fit decreasing method," in *International Conference on Computational Intelligence in Information System*. Springer, 2018, pp. 97–106.

[12] Y. Huang, R. Yang, L. Cui, T. Wo, C. Hu, and B. Li, "Vmcsnap: Taking snapshots of virtual machine cluster with memory deduplication," in *2014 IEEE 8th International Symposium on Service Oriented System Engineering*. IEEE, 2014, pp. 314–319.

[13] K. Gai, M. Qiu, and H. Zhao, "Cost-aware multimedia data allocation for heterogeneous memory using genetic algorithm in cloud computing," *IEEE transactions on cloud computing*, 2016.

[14] M. Diallo, A. Quintero, and S. Pierre, "An efficient approach based on ant colony optimization and tabu search for a resource embedding across multiple cloud providers," *IEEE Transactions on Cloud Computing*, 2019.

[15] S. Ghanavati, J. H. Abawajy, and D. Izadi, "An energy aware task scheduling model using ant-mating optimization in fog computing environment," *IEEE Transactions on Services Computing*, pp. 1–1, 2020.

[16] W. Zhong, Y. Zhuang, J. Sun, and J. Gu, "A load prediction model for cloud computing using pso-based weighted wavelet support vector machine," *Applied Intelligence*, vol. 48, no. 11, pp. 4072–4083, 2018.

[17] Z. Cao and S. Dong, "An energy-aware heuristic framework for virtual machine consolidation in cloud computing," *The Journal of Supercomputing*, vol. 69, no. 1, pp. 429–451, 2014.

[18] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 7, pp. 1366–1379, 2014.

[19] Z. Zhou, Z. Hu, and K. Li, "Virtual machine placement algorithm for both energy-awareness and sla violation reduction in cloud data centers," *Scientific Programming*, vol. 2016, pp. 1–11, 2016.

[20] E. Parvizi and M. H. Rezvani, "Utilization-aware energy-efficient virtual machine placement in cloud networks using nsga-iii meta-heuristic approach," *Cluster Computing*, pp. 1–23, 2020.

[21] X. Xu, Q. Liu, Y. Luo, K. Peng, X. Zhang, S. Meng, and L. Qi, "A computation offloading method over big data for iot-enabled cloud-edge computing," *Future Generation Computer Systems*, vol. 95, pp. 522–533, 2019.

[22] F. Ruan, R. Gu, T. Huang, and S. Xue, "A big data placement method using nsga-iii in meteorological cloud platform," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–13, 2019.

[23] 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," *Journal of Systems Architecture*, vol. 115, p. 101996, 2021.

[24] R. Shaw, E. Howley, and E. Barrett, "Applying reinforcement learning towards automating energy efficient virtual machine consolidation in cloud data centers," *Information Systems*, p. 101722, 2021.

[25] M. Ghahramani, R. Javidan, M. Shojafar, R. Taheri, M. Alazab, and R. Tafazolli, "Rss: An energy-efficient approach for securing iot service protocols against the dos attack," *IEEE Internet of Things Journal*, 2020.

[26] Y. Liang, Z. Hu, and K. Li, "Power consumption model based on feature selection and deep learning in cloud computing scenarios," *IET Communications*, vol. 14, no. 10, pp. 1610–1618, 2020.

[27] Z. Zhou, J. H. Abawajy, F. Li, Z. Hu, M. U. Chowdhury, A. Alelaiwi, and K. Li, "Fine-grained energy consumption model of servers based on task characteristics in cloud data center," *IEEE Access*, vol. 6, no. 1, pp. 27 080–27 090, 2018.

[28] W. Lin, Y. Zhang, W. Wu, S. Fong, L. He, and J. Chang, "An adaptive workload-aware power consumption measuring method for servers in cloud data centers," *Computing*, 2020.

[29] M. Hussain, L.-F. Wei, A. Lakhan, S. Wali, S. Ali, and A. Hussain, "Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing," *Sustainable Computing: Informatics and Systems*, p. 100517, 2021.

[30] F. U. Xiong and C. Zhou, "Virtual machine selection and placement for dynamic consolidation in cloud computing environment," *Frontiers of Computer Science*, vol. 9, no. 2, pp. 322–330, 2015.

[31] A. Horri, M. S. Mozafari, and G. Dastghaibyfard, "Novel resource allocation algorithms to performance and energy efficiency in cloud computing," *The Journal of Supercomputing*, vol. 69, no. 3, pp. 1445–1461, 2014.

[32] G. Han, W. Que, G. Jia, and L. Shu, "An efficient virtual machine consolidation scheme for multimedia cloud computing," *Sensors*, vol. 16, no. 2, p. 246, 2016.

[33] 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," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.

[34] J. Dong, H. Wang, and S. Cheng, "Energy-performance tradeoffs in iaas cloud with virtual machine scheduling," *China communications*, vol. 12, no. 2, pp. 155–166, 2015.

[35] S. K. Addya, A. K. Turuk, B. Sahoo, A. Satpathy, and M. Sarkar, "A game theoretic approach to estimate fair cost of vm placement in cloud data center," *IEEE Systems Journal*, vol. 12, no. 4, pp. 3509–3518, 2018.

[36] L. Zhang, T. Han, and N. Ansari, "Energy-aware virtual machine management in inter-datacenter networks over elastic optical infrastructure," *IEEE Transactions on Green Communications and Networking*, vol. 2, no. 1, pp. 305–315, 2018.

[37] S. Azizi, M. Shojafar, J. Abawajy, and R. Buyya, "Grvmp: A greedy randomized algorithm for virtual machine placement in cloud data centers," *IEEE Systems Journal*, 2020.

[38] Z. Zhou, J. Abawajy, M. Chowdhury, Z. Hu, K. Li, H. Cheng, A. A. Alelaiwi, and F. Li, "Minimizing sla violation and power consumption in cloud data centers using adaptive energy-aware algorithms," *Future Generation Computer Systems*, vol. 86, pp. 836–850, 2018.

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

[40] Z. Miao, P. Yong, Y. Mei, Y. Quanjun, and X. Xu, "A discrete pso-based static load balancing algorithm for distributed simulations in a cloud environment," *Future Generation Computer Systems*, vol. 115, pp. 497–516, 2021.

[41] 2021, "AFED-EF source code," https://github.com/mshojafar/sourcecodes/tree/master/Zhou2021AFED-EF_Sourcecode.

[42] C.-H. Hong and B. Varghese, "Resource management in fog/edge computing: a survey on architectures, infrastructure, and algorithms," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–37, 2019.

[43] B. Omoniwa, R. Hussain, M. A. Javed, S. H. Bouk, and S. A. Malik, "Fog/edge computing-based iot (feciot): Architecture, applications, and research issues," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4118–4149, 2018.
[44] S. Tuli, S. Ilager, K. Ramamohanarao, and R. Buyya, "Dynamic scheduling for stochastic edge-cloud computing environments using a3c learning and residual recurrent neural networks," *IEEE Transactions on Mobile Computing*, 2020.
[45] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning *et al.*, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *International Conference on Machine Learning*, 2018, pp. 1407—-1416.

Zhou Zhou received the Ph.D. degree from Central South University, Changsha, China, in 2017, majoring in computer science. He is currently an Associate Professor at Changsha University. He has published over 20 academic papers in cloud computing and green 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.

Mohammad Shojafar (M'17-SM'19) is a Senior Lecturer (Associate Professor) in the network security and an Intel Innovator, Profassional ACM member, and a Marie Curie Alumni, working in the 5G & 6G Innovation Centre (5GIC/6GIC) at the University of Surrey, UK. Before joining 5GIC/6GIC, he was a Senior Researcher and a Marie Curie Fellow in the SPRITZ Security and Privacy Research group at the University of Padua, Italy. Also, he was CNIT Senior Researcher at the University of Rome Tor Vergata contributed to 5G PPP European H2020 "SUPERFLUIDITY" project. Dr. Mohammad was a PI of PRISENODE project, a 275k euro Horizon 2020 Marie Curie global fellowship project in the areas of Fog/Cloud security collaborating at the University of Padua. He also was a PI on an Italian SDN security and privacy (60k euro) supported by the University of Padua in 2018. He was contributed to some Italian projects in telecommunications like GAUChO, SAMMClouds, and SC2. He received his Ph.D. degree from Sapienza University of Rome, Rome, Italy, in 2016 with an "Excellent" degree. He is an Associate Editor in *IEEE Transactions on Consumer Electronics*, *IEEE Systems Journal* and IET Communications. For additional information: http://mshojafar.com

Mamoun Alazab (SM'15) is an Associate Professor at the College of Engineering, IT and Environment at Charles Darwin University, Australia. He received his PhD degree in Computer Science from the Federation University of Australia, School of Science, Information Technology and Engineering. He is a cyber security researcher and practitioner with industry and academic experience. Alazab's research is multidisciplinary that focuses on cyber security and digital forensics of computer systems with a focus on cybercrime detection and prevention. He has more than 150 research papers in many international journals and conferences. He is a Senior Member of the IEEE. He is the founding chair of the IEEE Northern Territory (NT) Subsection. For more information: https://sites.google.com/view/alazabm

Jemal Abawajy (SM'11) is a full professor at Faculty of Science, Engineering and Built Environment, Deakin University, Australia. Professor Abawajy has served on the editorial-board of numerous international journals and currently serving as associate editor of the International Journal of Big Data Intelligence and International Journal of Parallel, Emergent and Distributed Systems. Prof. Abawajy 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 also served on the editorial-board of numerous international journals including *IEEE Transactions on Cloud Computing*. Prof. Abawajy is the author/co–author of more than 250 refereed articles and supervised numerous PhD students to completion. For additional information:https://www.deakin.edu.au/about-deakin/people/jemal-abawajy

Fangmin Li received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 2001, majoring in computer science. He is currently a Professor at Changsha University. He has published over 30 academic papers in wireless networks and energy-efficient resource management, and also holds ten Chinese patents. His current research interests include cloud computing, wireless communications and networks, and energy-efficient resource management.