Journal of Parallel and Distributed Computing 135 (2020) 191–202

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Managing renewable energy and carbon footprint in multi-cloud computing environments

Minxian Xu [a,b,1], Rajkumar Buyya [b,*]

[a] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China
[b] Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems, University of Melbourne, Australia

ARTICLE INFO

Article history:
Received 24 April 2019
Received in revised form 8 September 2019
Accepted 25 September 2019
Available online 9 October 2019

Keywords:
Cloud data centers
Renewable energy
Workload shifting
Carbon footprint
Brownout

ABSTRACT

Cloud computing offers attractive features for both service providers and customers. Users benefit from the pay-as-you-go model by saving expenditures and service providers are deploying their services to cloud data centers to reduce their maintenance efforts. However, due to the fast growth of cloud data centers, the energy consumed by the data centers can lead to a huge amount of carbon emission with environmental impacts, and the carbon intensity of different locations are varied among different power plants according to the sources of energy. Thus, in this paper, to address the carbon emission problem of data centers, we consider shifting the workloads among multi-cloud located in different time zones. We also formulate the energy usage and carbon emission of data centers and model the solar power corresponding to the locations. This helps to reduce the usage of brown energy and maximize the utilization of renewable energy at different locations. We propose an approach for managing carbon footprint and renewable energy for multiple data centers at California, Virginia, and Dublin, which are in different time zones. The results show that our proposed approaches that apply workload shifting can reduce around 40% carbon emission in comparison to the baseline while ensuring the average response time of user requests.

1. Introduction

As cloud computing environment is able to provide on-demand resources to a variety of applications, it emerges as a successful model for delivering utility-oriented computing services. By providing on-demand resources across the world, cloud computing is viewed as a new paradigm in IT industry. Cloud computing provides the pay-as-you-go model and can reduce the management complexity from the users' perspective. Nowadays, more and more service providers are migrating their workloads to clouds. The clouds can consist of multiple data centers across geographical locations, and each data center can have thousands of servers. The diversity of geographical locations brings the benefits of high reliability, disaster resistance and transparency for users in different time zones.

Although the cloud data centers have attractive features, such as pay-as-you-go model and low costs for users, the large amount of energy consumed by them has become a major issue. According to [12], the US data center consumed 91 billion kWh electricity in 2013, which is equivalent to the two years energy consumption of New York City households. The energy consumption is predicted to grow up to 140 billion kWh in 2020, which will generate 150 million tons of carbon emission. Since the underutilization and overloading of resources in infrastructure (e.g. computing, storage, networking, and cooling), the energy usage in cloud data centers is not efficient enough. The power is consumed while some of the resources are idle, which increases the management costs of data centers.

To relieve the high energy consumption and carbon footprint from data centers, a promising approach is improving resource utilization. This can reduce the number of active servers in data centers, thus the total energy consumption can be decreased when servicing the same amount of request. One dominant way to improve resource utilization is via virtualization technique [43]. With virtualization, multiple virtual machines (VMs) can be allocated to a single physical server. The VMs share the hardware resources and maximize server utilization. Additionally, operational costs are reduced by applying VM management to optimize cloud resources usage via dynamically provisioning resources. Resource utilization can also be improved via microservices [25], which are a set of self-contained application components and enable the fine-grained control on resource management.

* Corresponding author.
E-mail addresses: mx.xu@siat.ac.cn (M. Xu), rbuyya@unimelb.edu.au (R. Buyya).

[1] Minxian Xu was with the School of Computing and Information Systems, University of Melbourne; he is now with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China.

https://doi.org/10.1016/j.jpdc.2019.09.015
0743-7315/

Another approach for reducing carbon footprint is taking advantage of renewable energy (e.g. solar and wind) instead of coal-based brown energy. By applying renewable energy as another energy source, the brown energy usage and carbon footprint can be significantly reduced [29]. Enabling the data center to be partially or completely powered by renewable energy supports the cloud provider to reduce their dependency on coal-based energy sources. In different locations, the carbon intensity and availability of renewable energy are different, thus, the challenge is how to manage them in a global view, which is the objective of this work. We aim to apply the workload shifting to maximize the usage of renewable energy, however, as the unpredictability in supply of renewable energy, it is still needed to utilize the hybrid design of energy supply to support the full availability of cloud services.

The dominant cloud providers, like Amazon and Google, often manage geographically distributed data centers. The geographical distribution not only enhances the availability of the whole system, but it can also give the service provider more options to allocate requests based on different preferences to improve resource utilization for workloads execution. A concept called "*follow the renewable*" was proposed to encourage cloud providers to establish their data centers closer to the sources of renewable energy and the workloads can be distributed geographically [32]. In this work, we consider the preference related to the availability of renewable energy and impacts of carbon footprint. The problem is challenging due to the heterogeneous resources, varying availability of renewable energy and user define QoS. Some issues are required to be solved to achieve the objective, such as:

(a) How to provision resources for workloads execution in an energy efficient manner

(b) Where to shift the workloads in geographically distributed data centers so that renewable energy usage can be maximized

(c) When to shift the workloads thus the brown energy usage can be reduced while the users' QoS is ensured

By addressing these issues, this work makes following key **contributions**:

- Introduced a system model considering data center energy consumption, renewable energy and carbon footprint.
- Proposed workload shifting algorithm to manage the renewable energy and carbon footprint for data centers.
- Evaluated the impacts of requests distributed and processed in different time zones via renewable energy usage.

The rest of the paper is organized as follows. In Section 2, the related work are discussed. Section 3 introduces our system model, constraints and optimization objective. Our proposed workload shifting approach is presented in Section 4. The simulation configurations and results are demonstrated in Section 5. The summary and future directions are concluded in Section 6.

2. Related work

There has been extensive research on improving energy efficiency in cloud data centers. There are three levels of optimization that can be investigated for energy efficiency purposes, including software level, hardware level, and intermediate level [32]. Previously, the dominant energy efficient approaches can be mainly categorized as Dynamic Voltage Frequency Scaling (DVFS) [20] and VM consolidation [5]. However, both these approaches cannot function well when the whole data center is overloaded. Therefore, some complementary approaches, like brownout [34] have been proposed.

2.1. DVFS

DVFS is an energy efficient power management technique, which dynamically adjusts the frequency and voltage of machine components, e.g CPU, memory and storage. The DVFS-based approaches are designed to manage the energy according to operational frequency and voltage scaling, which can save power when the system is at the idle state and has less load. Wu et al. [39] proposed a DVFS-based approach to improve resource utilization and reduce energy consumption while ensuring system performance. The scheduled jobs are prioritized based on resource demand and specific SLA requirement. Wang et al. [38] introduced an approach in DVFS-enable cluster for precedence-constrained parallel tasks. The proposed approach can reduce energy consumption without increasing task processing time. Guerout et al. [16] introduced a methodology to simulate the DVFS process for energy efficiency purpose and applied a scientific application as a use case. Although DVFS-based approaches can reduce energy consumption, generally response time and service delay can be increased due to the switch between different frequency modes.

2.2. VM consolidation

Resource usage can be harnessed by VM consolidation, which aims to consolidate VMs to fewer machines, thus more active machines can be turned into the low power mode. With VM consolidation, data can also be transferred from one server to the other. Beloglazov et al. [7] proposed an energy efficient system based on OpenStack via VM consolidations to save power usage while ensuring QoS, which implemented several heuristics based on VM consolidation. Rossi et al. [30] presented an energy efficient cloud orchestrator combining VM consolidation and DVFS to improve the trade-offs between power savings and application performance. The proposed orchestrator has been validated under real testbed, and the results showed that energy consumption can be saved significantly while only leading to small portion extra costs. When applying energy efficient VM consolidation, generally there are trade-offs between energy and migration time, especially for the migration among geographically distributed data centers.

Nguyen et al. [26] introduced a virtual machine consolidation algorithm with multiple usage prediction based on local history to improve the energy efficiency of cloud data centers. The current and predicted resource usage are used to identify the overloaded or underloaded servers to find the best place for VM consolidation. Chen et al. [10] presented workload placement and migration approach under a distributed cloud computing environment, which considers the availability of renewable energy to maximize the throughput of the whole system. Different from our work, it applies for batch workloads and does not consider the carbon emission.

Virtual Graphics Processing Units (GPUs) are used in data centers to improve the resource utilization and reduce the energy consumption of Clouds [33]. Iserte et al. [18] analyzed the cluster equipped with remote virtual GPUs and the results showed that virtual GPUs can improve resource utilization while ensuring energy constraints. Varghese et al. [37] investigated virtual GPUs for financial application, which showed that the application efficiency can benefit from GPUs. Prades et al. [27] applied remote GPU virtualization framework to accelerate scientific applications executed on VMs. A task migration approach for virtual GPUs was proposed to demonstrate the possibilities to improve resource utilization [28]. Unlike these work, we consider to distribute workloads among data centers located in different time zones. Our CPU based scheduling can be extended easily to such heterogeneous architectures.

Table 1
Comparison of related work.

Approach	Technique					Environment	Metrics			
	DVFS	VM consolidation	Brownout	Green energy	Workload shift	Multiple clouds	Energy	Electricity costs	SLA	Carbon footprint
Wu et al. [39]							✓		✓	
Wang et al. [38]	✓						✓		✓	
Guerout et al. [16]	✓						✓		✓	
Beloglazov et al. [7]		✓					✓		✓	
Rossi et al. [30]	✓	✓					✓		✓	
Nguyen et al. [26]		✓					✓		✓	
Xu et al. [42]		✓	✓				✓		✓	
Hasan et al. [17]			✓	✓			✓		✓	
Liu et al. [22]				✓		✓	✓	✓		
Toosi et al. [36]				✓		✓	✓	✓		
Chen et al. [9]				✓		✓	✓	✓		
Adnan et al. [2]					✓	✓	✓	✓	✓	
Neglia et al. [24]				✓	✓	✓	✓	✓		
Khosravi et al. [19]		✓		✓		✓	✓	✓		✓
Chen et al. [10]				✓	✓	✓	✓		✓	
Goiri et al. [15]				✓	✓		✓		✓	
Our Approach			✓	✓	✓	✓	✓		✓	✓

2.3. Brownout

Brownout-based approach manages resource usage by dynamically controlling the running status of optional parts of applications in the cloud computing system [41]. Brownout can also be applied to microservices for fine-grained control on resources. Hasan et al. [17] investigated an adaptive application management approach based on dynamically switching application modes to improve the trade-offs among multiple optimization objectives. The optimization objectives include multiple metrics such as energy, user experience and performance for the interactive cloud application. In our previous work [40,42,44], we have proposed brownout-based approaches to manage application components in the system to reduce energy consumption while satisfying QoS. However, brownout approach has not been investigated in geographical data centers and the carbon emission is not considered yet.

2.4. Multi-cloud management

Some existing research has proposed approaches for managing resources in multi-cloud environments. Liu et al. [22] proposed geographical load balancing method by using renewable energy, and the method can reduce the brown energy usage. Toosi et al. [36] proposed a framework to balance loads of web application among multiple data centers based on the availability of renewable energy and aimed to reduced total electricity costs. Chen et al. [9] introduced a workload and energy management mechanism to reduce the network operational cost and energy costs. Adnan et al. [2] presented a dynamic workloads deferral algorithm for multi-cloud to fit into the dynamic electricity prices in different locations while ensuring the workloads deadline. Neglia et al. [24] proposed a workload scheduling approach based on Markov Chain to dispatch workloads to geographical data centers with renewable energy. In these articles, they focused on reducing the total electricity costs while the carbon emission was not considered. Our work takes advantage of brownout-based methodology and aims to reduce the carbon emission.

2.5. Carbon footprint

There are also some works combining energy consumption and carbon footprint for cloud data centers. Khosravi et al. [19] proposed a VM placement approach for reducing energy and carbon costs in geographically distributed cloud data centers, while all the locations are in the same country. Doyle et al. [13] presented a method for managing carbon emissions, while its objective is load balancing and renewable energy is not considered. Goiri et al. [15] proposed Parasol and GreenSwitch as the prototype system, which enables to dynamically schedule workloads and select different sources of energy. Unlike our work, all the servers in this work are in the same site.

Table 1 shows the comparison of the related work. Compared to existing works, we apply our proposed workload shifting approach to schedule workloads to different data centers, and our objective is minimizing the total carbon emission while ensuring the average response time of requests. Moreover, we consider geographically distributed data centers in different time zones (e.g. US and Europe) with different carbon intensities and availability of renewable energy.

3. System model

In this section, we present the system model of our proposed approach. Table 2 defines the symbols that are used throughout this paper.

The target system is demonstrated in Fig. 1. Multiple data centers are located geographically and are connected via the network. The main entities in this system contain User, Cloud Scheduler and Data centers.

- **User:** The users are from different locations (e.g. Europe, US and etc.) and submit their requests to the cloud for execution.
- **Cloud Scheduler:** The cloud scheduler is responsible for assigning the received requests to different data centers for execution based on scheduling policies. The scheduling process falls into the MAPE-K [3] loop, which has resource monitoring (Monitor), resource analysis (Analyze), scheduling plan (Plan) and policy execution (Execute) phases. The modeling and policies are managed by the Knowledge module.
- **Data Centers:** The data centers are providing the physical and virtualization resources for the system, and they can be distributed at different locations.

3.1. Data centers

The whole system consists of n data centers that are located at different locations with different time zones, denoted as $D = d_1, d_2, \ldots, d_n$. Each data center has multiple servers, e.g. m servers, $S = s_1, s_2, \ldots, s_m$. The data center can use two energy

Table 2
Symbols and definitions.

Symbol	Definition
D	The set of data centers
d_j	The nth data center in D
S	Physical machine list in a data center
s_i	The ith physical machine in the list
B	The brown energy
G	The green energy
P_{idle}	Power consumption when physical machine is idle
P_{max}	Maximum power when physical machine is running
R_E^G	Carbon intensity of green energy source
R_E^B	Carbon intensity of brown energy source
C_T	Total cost for executing all requests
W	The set of requests
r_k	The kth request in W
C_k, i, j	The cost to execute request r_k on physical host s_i in data center d_j
C_F	Total carbon footprint
C_R	Total cost of response time
E_{d_j}	Energy consumption of data center d_j
t	Time interval
T	Total time intervals
$P_s(t)$	Server power at time interval t
$P_c(t)$	Cooling power at time interval t
$u_{s_i}(t)$	Utilization of physical machine s_i at time interval t
vm_l	The lth Virtual machine
V_{s_i}	The list of virtual machines on s_i
$u_{vm_l}(t)$	The utilization of vm_l at time interval t
A_{vm_l}	The set of microservices on vm_l
ms_o	The o_{th} microservices
$u_{ms_o}(t)$	The utilization of ms_o at time interval t
CoP	The function to calculate the cooling efficiency of cold air
T_{sup}	Cooling air supply temperature
$E_{d_j}^B$	Brown energy usage of data center d_j
$E_{d_j}^G$	Green energy usage of data center d_j
$C_{d_j}^U$	Carbon intensity of data center d_j
$C_{d_j}^F$	Carbon emission of data center d_j
T_{k,d_j}	VM Response time of r_k allocated to d_j
$T_{d_j,d_j'}$	Extra response time when request is forwarded from d_j to d_j'
$C_{r_k}^R$	Response time of r_k
M	The size of data centers
N	The maximum number of physical machines in all data centers

sources, brown energy and green energy to supply for servers, network devices, cooling systems, and other devices. The brown energy B comes from the coal-based facility and green energy G comes from renewable energy, like solar or wind. We denote the energy sources as $E = \{B, G\}$.

3.2. Cloud scheduler

Cloud Scheduler is connecting the physical resources in data centers and requests submitted by users. It receives user requests and dispatches them to different data centers for processing. The cloud scheduler makes decisions based on resource requirement of requests, energy consumption, and carbon footprint to achieve an optimized objective. The cloud scheduler fits into the MAPE-K model as below:

Cloud Users submit their requests to cloud scheduler. The requests can have arrival time and execution time. The requests should be executed within a required time. In the Monitor phase, the Cloud Scheduler monitors the requests and system running status, then provides the collected information to Analyze phase for analysis. After analyzing, the Cloud Scheduler plans the scheduling policies based on optimization objectives. In the Execute phase, the scheduling policies are executed to adjust system status.

3.3. Data center energy consumption

Energy consumption is the dominant factor that has an impact on carbon footprint. In our data center energy consumption model, we mainly consider server power consumption and cooling consumption. As noted in [31,32], these two parts can make up to 60% energy consumption of the data center.

3.3.1. Server power model

We adopt the server power model derived from [45], which includes P_{idle} and P_{max}. The utilization comes from the virtual machines deployed on physical machines. And we mainly consider the utilization as CPU utilization. The power consumption of the server is linear to server utilization.

At time interval t, the power from the server side is

$$P_s(t) = \begin{cases} P_{idle} + u_{s_i}(t) \times (P_{max} - P_{idle}), & u_{s_i}(t) > 0 \\ 0, & u_{s_i}(t) = 0 \end{cases} \quad (1)$$

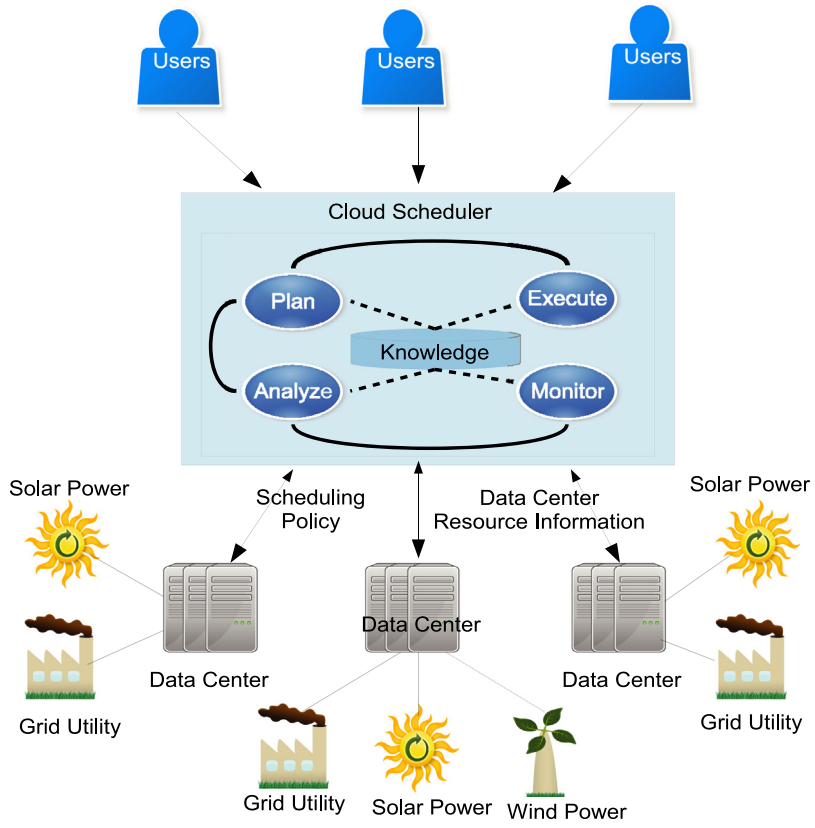

Fig. 1. Target system.

where $u_{s_i}(t)$ is the utilization of server s_i at time interval t, which is equal to the sum of the utilization of VMs running on s_i, and can be represented as:

$$u_{s_i}(t) = \sum_{vm_l \in V_{s_i}} u_{vm_l}(t) \tag{2}$$

where V_{s_i} is the set of VMs assigned on s_i, and the utilization of vm_l at time interval t is $u_{vm_l}(t)$ that is represented as the utilization sum of microservices composed of the application, thus,

$$u_{vm_l}(t) = \sum_{ms_o \in A_{vm_l}} u_{ms_o}(t) \tag{3}$$

where A_{vm_l} is the set of microservices on vm_l.

3.3.2. Cooling power model

For cooling power P_c, we use the model from HP lab data center [23] as follows:

$$CoP(T_{sup}) = 0.0068T_{sup}^2 + 0.0008T_{sup} + 0.458 \tag{4}$$

The CoP (Coefficient of Performance) is a method to calculate the cooling efficiency based on cold air supply temperature T_{sup} maintained by cooling equipment.

We consider the data center thermal control is managed by Computer Room Air Condition (CRAC) system [21], which can contain multiple CRAC units to transfer cold air to the hosts to reduce hotspots. Based on server power consumption and cooling efficiency, we can calculate the power consumed by cooling equipment $P_c(t)$ as:

$$P_c(t) = \frac{P_s(t)}{CoP(T_{sup})} \tag{5}$$

Total energy consumption $E_{s_i}(t)$ of a single server s_i consists of server power $P_s(t)$ and cooling power $P_c(t)$, which can be represented as:

$$E_{s_i}(t) = \int_t^{t+\Delta t} (P_s(t) + P_c(t))\Delta t \tag{6}$$

Then the total energy consumption E_{d_j} of data center d_j can be represented as the sum of energy from all the servers and cooling equipments:

$$E_{d_j} = \sum_{t \in T} \sum_{s_i \in S} E_{s_i}(t) \tag{7}$$

3.4. Renewable availability

In different locations, the availability of renewable energy can vary significantly. For instance, in some locations, the solar irradiance is quite sufficient, while in some other locations, winds are the main renewable energy sources. As renewable energy availability is dependent on the weather, at the same moment while in different time zones, the availability can be quite different. We aim to coordinate the available renewable energy to handle the users' requests, thus, the total carbon footprint can be reduced. In this paper, we consider solar energy as renewable energy to power the data centers. We also consider that renewable energy is used with higher priority than brown energy, which means as long as renewable energy is available, it will be used first.

3.5. Carbon intensity

Carbon intensity can also vary vitally according to the source of power. We denote R_E^G, R_E^B as the sources of green energy and brown energy, which is evaluated as grams per kilowatt-hour

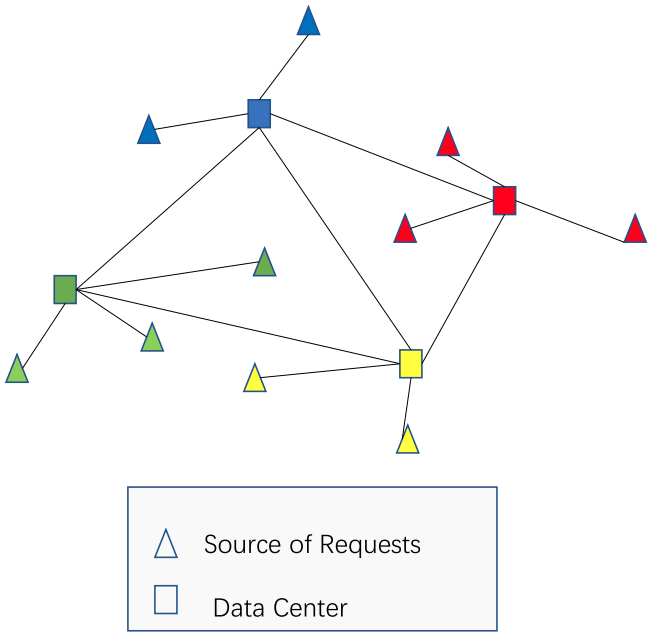

Fig. 2. An example of data centers and sources of requests.

used electricity (g/kWh). This value is related to the type of coal-based fuel to generate the electricity, while for the green energy, the value is 0. Also, in different locations, the carbon intensity can be different, for example, in Norway, its carbon intensity is 6 g/kWh, while in Australia, the value is 870 g/kWh [35]. We aim to dispatch requests to the data center with more renewable energy, however, the response time of requests can be increased. Thus, the average time is taken into consideration when designing scheduling policies. According to [13], the average response time can be increased if requests are routed from a data center region to a geographical region.

As shown in Fig. 2, we consider the data center and source of requests are as a undirected graph. The data centers and sources of requests are considered as nodes, and the edge is the connection between request and data center. In our model, the requests from a source will be initially assigned to the data center based on response time, which means the requests are processed by the nearest data center. To enable the requests to be forwarded when green energy is not enough, the data center nodes are connected.

3.6. Objective function

The objective function is minimizing the combination of average response time and carbon footprint. The total cost to execute the requests is:

$$C_T = \sum_{r_k \in W} \sum_{s_i \in D_j} C_{k,i,j} \tag{8}$$

where r_k is the request, and W is the total number of requests.

This objective function can be divided into two parts: the carbon footprint C_F and the response time C_R.

$$C_T = C_F + C_R \tag{9}$$

The carbon footprint is related to the brown energy used for executing the requests and the carbon intensity at the site. Therefore, we need to calculate the total energy consumption of data centers, including the energy from servers and cooling equipment.

The brown energy usage is defined as:

$$E^B_{d_j} = \max(E_{d_j} - E^G_{d_j}, 0) \tag{10}$$

if the required energy is more than the available green energy $E^G_{d_j}$, the brown energy $E^B_{d_j}$ usage will be a positive value, otherwise, if green energy is enough, then brown energy usage is 0.

Total carbon emission of data center d_j is calculated as:

$$C^F_{d_j} = E^B_{d_j} \times C^I_{d_j} \tag{11}$$

where $C^I_{d_j}$ is the carbon intensity at data center d_j

Total carbon emission is the sum of the carbon emission from all the data centers in D:

$$C_F = \sum_{d_j \in D} C^F_{d_j} \tag{12}$$

As for the response time, the request k allocated to d_j is denoted as T_{k,d_j}, however, if the request needs to be forward to another data center d'_j, there is incurred response time as T_{d_j,d'_j}, thus the response time is $C^R_{r_k} = T_{k,d_j} + T_{d_j,d'_j}$, and the average response time is:

$$C_R = \frac{1}{|W|} \sum_{r_k \in W} C^R_{r_k} \tag{13}$$

The following constraints should be satisfied:

The total VM utilization for executing requests should not exceed the capacity of physical machines. We consider the full capacity of a single physical machine is 1.0.

$$\sum_{vm_l \in V_{s_i}} u_{vm_l} \leq 1.0, \forall s_i \in d_j \tag{14}$$

The total running microservices on a single VM should not exceed the capacity of the VM.

$$\sum_{ms_o \in A_{vm_l}} ms_o \leq 1.0, \forall vm_l \in s_i \tag{15}$$

Finally, the optimization problem becomes to minimize the total costs while satisfying the physical machine and VM capacity requirements.

$$\begin{aligned} &\min \quad C_T \\ &s.t. \quad \textit{constraints} \quad (14)\text{–}(15) \end{aligned} \tag{16}$$

4. Workload shifting algorithm

In this section, we propose a workload shifting algorithm with green energy usage to optimize our objective in Eq. (16). Then we derive another two algorithms by changing the priorities of different parameters to investigate their impacts on brown energy usage, carbon emission and average response time.

4.1. Workload shifting with green-energy (WSG)

Algorithm 1 shows the pseudocode of our proposed workload shifting algorithm. The objective the algorithm is shifting the workloads to the data center with sufficient green energy while ensuring the average response time. The algorithm mainly has the following steps:

(1) Assign request $r_k \in W$ to the nearest data center d_j to reduce the response time of the request (lines 2–3). The source of the request has been assigned a default data center with the lowest latency.[2] The default data center is considered as the nearest one for the requests from a specific source.

(2) Calculate the increased energy if r_k is allocated to the s_i in d_j (line 4). For s_i, it should be the physical machine which has the

[2] In this work, latency and response time are used interchangeably.

Algorithm 1: Workload Shifting Algorithm.

Input: the set of data center D with size n, host list in each data center S, virtual machine list V, microservices list A, request list W, time interval t, response time threshold T_r
Output: request allocated destination

1: At time interval t, collect data center information (energy consumption, carbon intensity and renewable energy amount) from cloud scheduler
2: **for** r_k in W **do**
3: Allocate the r_k to the nearest data center d_j
4: Assign r_k to the s_i that has the least increased energy consumption $E^{\Delta}_{r_k}$
5: Check the amount of available renewable energy $E^{R}_{d_j}$ of data center d_j
6: **if** $E^{R}_{d_j} > E^{B}_{d_j} + E^{\Delta}_{r_k}$ **then**
7: Allocate r_k to d_j
8: Update $E^{B}_{d_j} = E^{B}_{d_j} + E^{\Delta}_{r_k}$
9: Update C_R, C_F, C_T
10: **else**
11: Sort all the data centers in data center set based on available green energy in descending order
12: Check the increased response time ΔC_R
13: Find the data center that can increase the least response time while ensuring $C_R + \Delta C_R < T_r$
14: **if** data center d_j is found **then**
15: Allocate r_k to d_j
16: Update C_R, C_F, C_T
17: **else**
18: Trigger brownout by deactivating lowest utilization component
19: Allocate r_k to current data center
20: Update C_R, C_F, C_T
21: **end if**
22: **end if**
23: **end for**
24: **return** destination

least increased energy consumption. The idea is based on the Best Fit Decreasing algorithm in [6], in this way, the increased energy consumption of the data center can be minimized.

(3) Check the availability of green energy in the allocated data center (line 5). The algorithm would allocate the request to the data center with sufficient green energy with higher priority.

(3a) If renewable energy is sufficient, allocate it. The best destination for this request has been found. Go to step 5.

(3b) If green energy is not sufficient, forward it to another data center with available green energy. Go to Step 4.

(4) Find another suitable data center with enough renewable energy, while the average response time is not exceeding the threshold (lines 11–13). The data centers are sorted based on available green energy.

(4a) If the data center is found, allocate request to the data center. The destination for this request has been found. Go to step 5.

(4b) If no suitable data center, which means the response time constraint will be violated or no data center has sufficient green energy. In this situation, the brownout mechanism is triggered (deactivating some microservices temporarily) to reduce energy and carbon footprint. The request will not be forwarded. Go to step 5.

(5) Update data center information, including carbon emission and available green energy.

(6) Return the allocated destination of the request.

Algorithm Complexity Analysis: We assume the number of data centers as N and the maximum number of physical machine in all the data centers as M. The time to find a host in the nearest data center with sufficient renewable energy is $\Theta(NlogN)$, which can be found by a sorting algorithm. Then the time to find the physical machine with the least increased energy is $\Theta(MlogM)$, which can also be searched by sorting algorithm. Thus, the time to find a physical machine in a data center with sufficient renewable energy is $\Theta(NlogN)+\Theta(MlogM)$. While if the nearest data center does not have sufficient renewable energy, this process will be executed for other data centers, and the maximum execution is the number of data center M. Therefore, the final algorithm complexity is $M \times (\Theta(NlogN) + \Theta(MlogM))$, which is equal to $\Theta(MNlogN + M^2logM)$.

4.2. Workload shifting non brownout (WSNB)

WSNB is very similar to WSG, the only difference compared with WSG is that WSNB omits the line 18 in Algorithm 1, which represents that brownout mechanism is not applied, thus although another suitable data center is not found, the optional components are not deactivated to reduce brown energy usage. This algorithm can be applied to the applications that have no optional components. Due to the minor modification, the algorithm complexity of WSNB is as same as WSG.

4.3. Workload shifting with time (WST)

The WST algorithm differs from WSG in the way that WST cares more about average response time rather than green energy usage. Compared with Algorithm 1, in line 11, the candidate data centers are sorted based on latencies in ascending order, and in line 13, the algorithm will find the first data center with sufficient green energy. In this case, the workloads are shifted to the data center with the lowest latency and green energy is sufficient. Although the priority of data center selection is changed, the complexity of WST remains the same as WSG.

5. Performance evaluation

In this section, we evaluate our proposed workload shifting algorithms in Section 4 to investigate the impacts on brown energy usage and carbon emissions. To make the simulations as realistic as possible, we consider the data centers located at US and Europe, and the workload is derived from Facebook. We use CloudSim [8] simulation toolkit to evaluate the performance of proposed algorithm.

5.1. Experimental settings

5.1.1. Data center and source of requests

To simulate the multi-cloud environment with data centers in different time zone, we select 3 locations (California, Dublin, and Virginia) as data centers (square symbols) and 10 locations as the sources of requests (circle symbols). These locations are distributed in the US and Europe and in different time zones. The reason why we choose the three locations as data centers is that we simulate the Amazon deployment, which has data centers in these locations. Each source can be connected to three data centers, which is demonstrated in Fig. 3. We use the same color of data center and source of requests to represent that the latency between them is the lowest.

The average latency between the data centers and sources is listed in Table 3. This data comes from [13], which is collected via a real cluster within two days and the interval is 15 min. In our algorithm, we use the latency data to find the nearest data center and calculate the extra latency when requests are forwarded to another data center. For example, for requests from Miami, the California data center has the lowest latency.

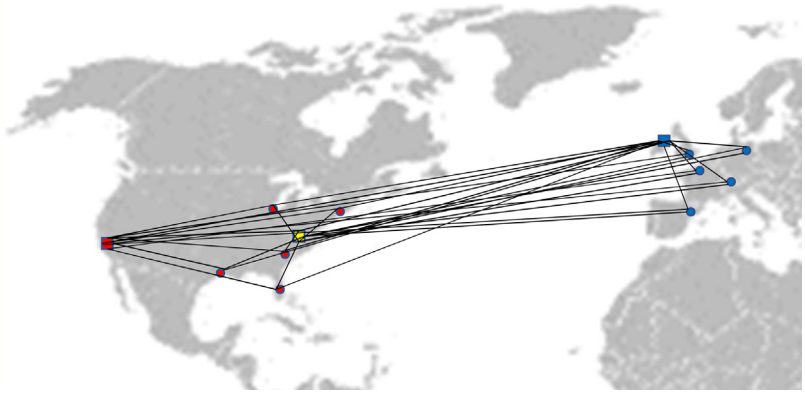

Fig. 3. Locations of data centers and request sources in simulation settings.

Table 3
Average latency between data center and sources of requests and daily number of requests at sources.

Region	California, US (ms)	Dublin, Ireland (ms)	Virginia, US (ms)	Number of Requests (Millions)	Time Zone (GMT)
Miami, US	54.26	171.87	98.21	1.97	−5
Paris, France	192.44	21.24	184.75	9.14	+1
Berlin, Germany	177.74	40.89	157.68	0.68	+1
Chicago, US	63.81	142.78	102.08	1.62	−6
Milan, Italy	188.71	44.71	167.3	0.94	+1
New York, US	96.45	78.71	134.11	14.108	−5
Los Angeles, US	27.66	213.48	153.28	7.14	−8
Barcelona, Spain	194.55	35.83	172.47	2.61	+1
Houston, US	37.61	151.93	98.96	0.99	−6
London, UK	175.59	17.62	163.25	8.53	0

5.1.2. Workload pattern

To simulate the number of requests, we also estimate the daily active users from different locations based on Facebook data [14]. Facebook data is used because it provides a wide range of applications and has users from various locations. We assume each user submits 1 request daily, and the number of request per day from the sources is shown in Table 3. For instance, New York is the most active city with 14.108 million active users while Berlin is the least active city among all sources with 0.66 million active users.

After we obtain the total number of daily requests, we need to convert it to follow a typical pattern that represents the fluctuations in a day. It has been analyzed that the realistic workloads follow the diurnal cycle that has peak and bottom during the day and night. We partitioned the daily data into 24-hour time intervals based on the Facebook workloads pattern analyzed in [4], and adjusted the time zones to match all the 10 sources of requests. The change of the number of requests from different sources is shown in Fig. 4. For example, Miami city has the larger number of requests compared to other cities, its requests reach the bottom and peak at hours 4 and 13 respectively.

5.1.3. Carbon intensity

We also obtain the one-day carbon intensity of the three data center from [35],[3] which contains the hourly carbon intensity data as depicted in Fig. 5. The figure shows that the carbon intensity varies slightly during the observed time and there are some differences between the three data centers. The carbon intensity of California ranges from 254 g/kWhr to 333 g/kWhr, and Virginia has carbon intensity from 338 g/kWhr to 375 g/kWhr. Ireland has a higher carbon intensity between 391 g/kWhr and 433 g/kWhr.

[3] The data was obtained on 9 Jan, 2019

Table 4
Host/VM types and capacity.

Name	CPU	Cores	Memory	Bandwidth	Storage
Host Type 1	1.86 GHz	2	4 GB	1 Gbit/s	100 GB
Host Type 2	2.66 GHz	2	4 GB	1 Gbit/s	100 GB
VM Type 1	2.5 GHz	1	870 MB	100 Mbit/s	1 GB
VM Type 2	2.0 GHz	1	1740 MB	100 Mbit/s	1 GB
VM Type 3	1.0 GHz	1	1740 MB	100 Mbit/s	1 GB
VM Type 4	0.5 GHz	1	613 MB	100 Mbit/s	1 GB

5.1.4. Solar power

We consider solar power as renewable energy. Fig. 6 shows the solar power data obtained from [11], which has been adjusted based on time zones. In the night time, the solar power is 0, and during the day time, different locations have different solar power. California has more powerful solar energy i.e. about 1000 W/m^2 compared with the other two sites. We can also notice from Fig. 6, by taking advantage of multi-cloud at different time zones, the duration of green energy availability can be extended. In most time periods, the green energy can be utilized except for time period from hour 4 to hour 5.

5.2. Compared algorithms

To evaluate the effects of workload shifting, we use cloud simulation toolkit CloudSim [8]. 450 physical machines in total are used, and each data center contains 150 physical machines. As summarized in Table 4, we use two types of physical machines and four types of VMs according to the offering from EC2. The power model of physical machines is based on IBM System x3550 M3 with CPU Intel Xeon X5670 and X5675 [1]. The different utilization levels with the corresponding power are shown in Table 5. As for the utilization of VMs that are allocated on physical machines, it can be simulated via the utilization of

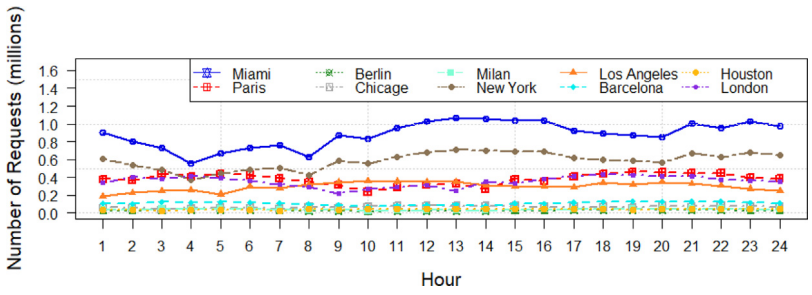

Fig. 4. Number of requests from different sources.

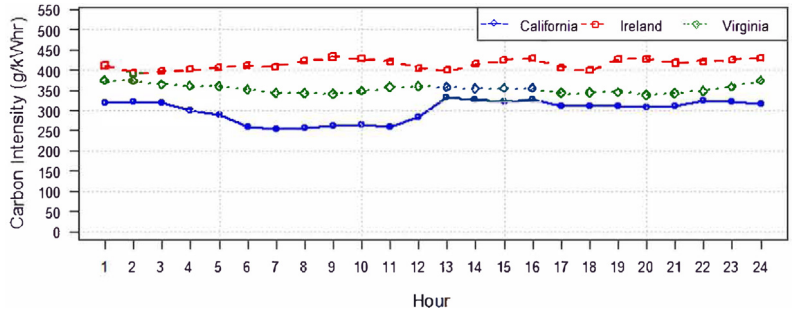

Fig. 5. Carbon intensity of three data centers in one-day time.

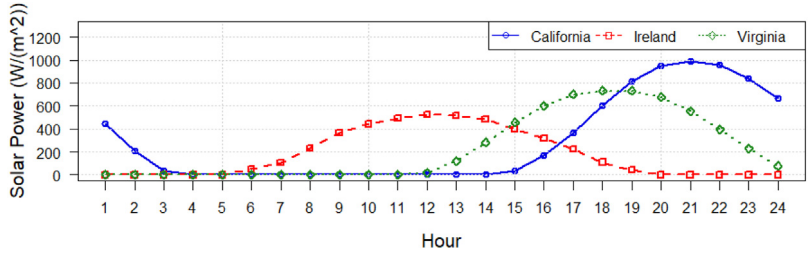

Fig. 6. Solar energy of three data centers in one-day time.

Table 5
Power consumption of servers in Watts.

Servers	0% (sleep mode)	10%	20%	30%	40%	50%
IBM x3550 M3 (Intel Xeon X5670 CPU)	66	107	120	131	143	156
IBM x3550 M3 (Intel Xeon X5675 CPU)	58.4	98	109	118	128	140
Servers	60%	70%	80%	90%	100% (max)	
IBM x3550 M3 (Intel Xeon X5670 CPU)	173	191	211	229	247	
IBM x3550 M3 (Intel Xeon X5675 CPU)	153	170	189	205	222	

application components (microservices) that are running on the VMs as shown in Eq. (3).

We also implement four algorithms for performance comparison as below:

- **NWS (Non Workload Shifting):** the algorithm does not apply the workload shifting. The requests are processed by the nearest data center based on average response time.
- **WSNB(Workload Shifting Non Brownout):** the algorithm that applies workload shifting in Algorithm 1 while the brownout mechanism is not applied.
- **WSG (Workload Shifting with Green-energy):** the algorithm we proposed in Algorithm 1, which aims to maximize the green energy usage while ensuring the average response time.
- **WST (Workload Shifting with Time):** the algorithm is very similar to WSG only with the change of data center selection. This algorithm cares more about average response time than carbon emission. When workloads are needed to be shifted, the algorithm finds the first data center with sufficient green energy and the lowest average response time to execute the workloads.

To support the brownout feature for applications, we extend the cloudlet model in CloudSim to model web application with optional components, e.g. the online shopping application with recommendation engine as an optional component. Each component has its CPU utilization, and when the component is deactivated based on brownout, the amount of corresponding CPU utilization is reduced. For WSG and WST, we configure the amount of optional utilization of application as 20%, which represents how

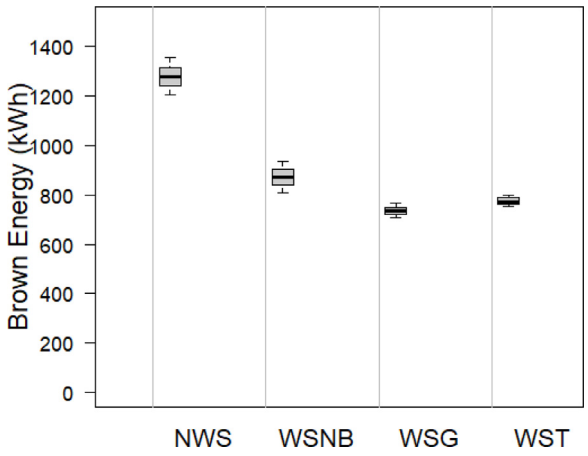

Fig. 7. Comparison of brown energy usage.

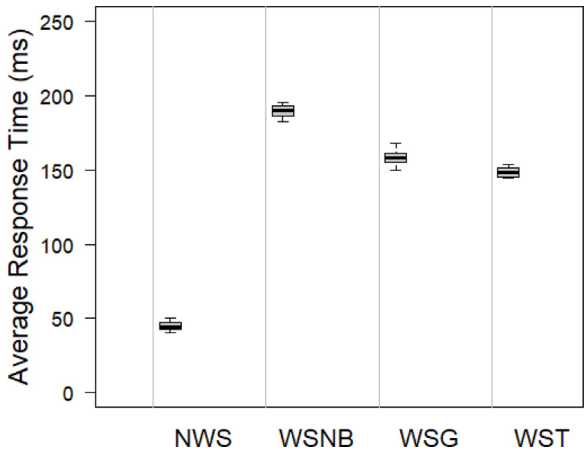

Fig. 9. Comparison of average response time.

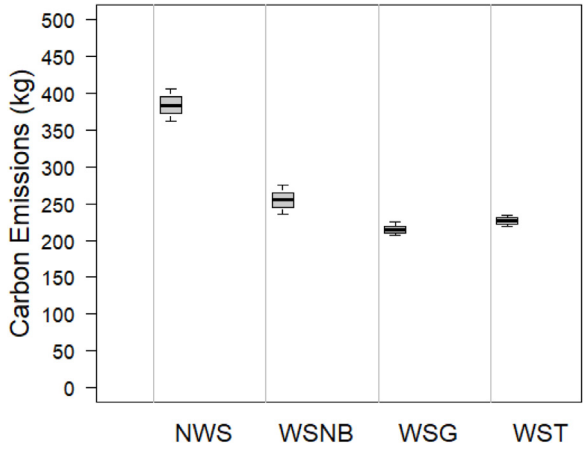

Fig. 8. Comparison of carbon emission.

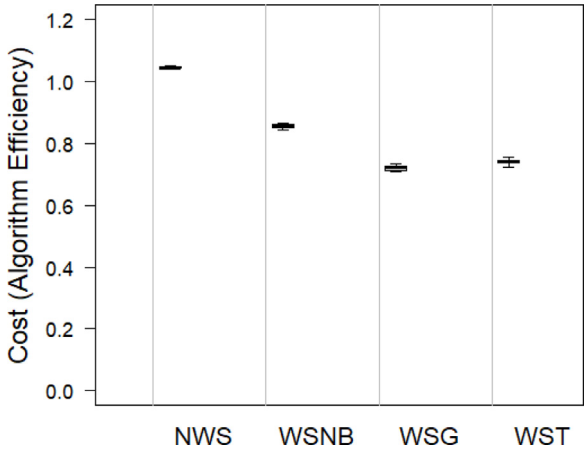

Fig. 10. Comparison of cost (algorithm efficiency).

much utilization can be deactivated in application. We configure this utilization ratio as it has been evaluated in [42] and shows to be effective for the brownout-based approach.

5.3. Results

To evaluate the performance of compared algorithms, we conducted experiments to evaluate brown energy usage, carbon emission and average response time for all three data centers in 24 h. The experiments were repeated 7 times.

Fig. 7 shows the comparison of brown energy usage for the compared algorithms. In NWS, the green energy is not applied, therefore it has the highest brown energy usage compared to other algorithms. The brown energy usage of NWS is 1276.8 KWh with 95% confident interval (CI): (1237.4, 1316.2). By applying workload shifting, in WSNB reduces the brown energy usage to 869.9 kWh with 95% CI: (835.4, 904.3). By using the brownout mechanism in WSG and WST while with different priorities in carbon emission and average response time, both of these algorithms decrease the brown energy usage. WSG reduces 43% to 735.6 kWh with 95% CI: (719.9, 751.3), and WST lowers 40% to 774.2 kWh with 95% CI: (760.6, 787.8). Since the results of WSG and WST are quite close, we conduct the paired t-test, the $p < 0.05$, which means there are significant differences between the two algorithms. Based on the results, we can see the workload shifting can vitally reduce brown energy usage.

In Fig. 8, we compare the carbon emission resulted from the compared algorithms. As NWS consumes the largest amount of brown energy, it also leads to more carbon emission than other algorithms. NWS has the carbon emission as 383.6 kg with 95% CI (371.7, 395.4). For other algorithms that apply workload shifting, WSNB has 255.1 kg carbon emission with 95% CI (244.8, 265.4) by 33.5% reduction, WSG reduces 44% carbon emission to 215.1 kg with 95% CI (210.5, 219.8), and WST decreases the carbon emission to 226.6 kg with 95% CI (222.5, 230.6). We also conduct the paired t-test for WSG and WST, where the $p < 0.05$ and it means there are significant differences between the WSG and WST. WSG outperforms other algorithms in carbon emission.

Fig. 9 demonstrates the comparison of average response time, different from the above comparison, as workload shifting is not applied, NWS has the shortest average response time as 45.2 ms. WSNB has the longest average response time as 189.3 ms, with brownout, WSG, and WST slightly reduce the average response time to 158.3 ms and 148.4 ms respectively. Although the algorithms with workload shifting have longer average response time than NWS, the average response time are in the acceptable range, which are less than 1 s.

As the carbon emission and average response time are in different units, we normalize the values for comparison to measure the total cost in Eq. (9). We set the carbon emission of NWS as the baseline and 1000 ms as the baseline of average response time. Then we calculate the algorithm efficiency (the ratio compared

with baseline) of total cost as demonstrated in Fig. 10. NWS has the cost as the average of 1.045, and WSG achieves the best algorithm efficiency as an average of 0.719.

As a result, we can conclude that our proposed algorithm based on workload shifting and brownout performs better in terms of brown energy and carbon emission than baselines, while the average response time can be optimized within the acceptable range.

6. Conclusions and future work

In this paper, we investigated the approach to reduce brown energy and carbon emission in cloud data centers by utilizing green energy. To maximize green energy usage, we consider applying workload shifting among the multiple data centers located in different time zones. Through the modeling of carbon intensity and solar energy, we propose the workload shifting algorithm to balance the total carbon emission and the average response time of requests. By modeling the data centers located at California, Virginia, and Dublin, we conduct the simulation-based experiments, and the results demonstrate that our approach can significantly reduce the carbon emission while ensuring the average response time.

As future work, we plan to consider to apply this approach with other application models, such as Map-reduce and bag-of-task applications. We would also like to consider delaying workload execution to maximize renewable energy usage if QoS permits. In addition, the network impact on workload shifting can be modeled with a more comprehensive model. For the renewable energy sources, we would like to explore the wind mills as the complementary of solar power.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.jpdc.2019.09.015.

Acknowledgments

This work is supported by China Scholarship Council, Australian Research Council Discovery Project (DP160102414), Shenzhen Basic Research Program (No. JCYJ20170818153016513) and National Natural Science Foundation of China (No. 61802387). We thank Shashikant Ilager for his suggestions on improving this paper. We thank Editor-in-Chief (Prof. Viktor Prasanna), Guest Editors (Prof. Carlos Reano, Prof. Blesson Varghese and Prof. Federico Silla), and anonymous reviewers for their excellent comments on improving the paper.

References

[1] Standard Performance Evaluation Corporation, 2015, http://www.spec.org/power-ssj2008/results/res2010q2/.
[2] M.A. Adnan, R. Sugihara, R.K. Gupta, Energy efficient geographical load balancing via dynamic deferral of workload, in: 2012 IEEE Fifth International Conference on Cloud Computing, IEEE, 2012, pp. 188–195.
[3] P. Arcaini, E. Riccobene, P. Scandurra, Modeling and analyzing MAPE-K feedback loops for self-adaptation, in: Proceedings of the 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 2015, pp. 13–23.
[4] B. Atikoglu, Y. Xu, E. Frachtenberg, S. Jiang, M. Paleczny, Workload analysis of a large-scale key-value store, in: ACM SIGMETRICS Performance Evaluation Review, vol. 40, ACM, 2012, pp. 53–64.
[5] A. Belog1azov, 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. 24 (7) (2013) 1366–1379.
[6] A. Beloglazov, R. Buyya, Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers, Concurr. Comput.: Pract. Exper. 24 (13) (2012) 1397–1420.
[7] A. Beloglazov, R. Buyya, Openstack neat: a framework for dynamic and energy-efficient consolidation of virtual machines in openstack clouds, Concurr. Comput.: Pract. Exper. 27 (5) (2015) 1310–1333.
[8] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. De Rose, R. Buyya, Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw. - Pract. Exp. 41 (1) (2011) 23–50.
[9] T. Chen, Y. Zhang, X. Wang, G.B. Giannakis, Robust geographical load balancing for sustainable data centers, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2016, pp. 3526–3530.
[10] D. Cheng, C. Jiang, X. Zhou, Heterogeneity-aware workload placement and migration in distributed sustainable datacenters, in: IPDPS, 2014, pp. 307–316.
[11] E. Commission, Photovoltaic geographical information system, 2017, http://re.jrc.ec.europa.eu/pvg_tools/en/tools.html.
[12] P. Delforge, Data center efficiency assessment - scaling up energy efficiency across the data center industry: evaluating key drivers and barriers, 2014, https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IP.pdf.
[13] J. Doyle, R. Shorten, D. O'Mahony, Stratus: load balancing the cloud for carbon emissions control, IEEE Trans. Cloud Comput. 1 (1) (2013) 1.
[14] Fastbooking, Top 50 cities on facebook, 2014, https://www.fastbooking.com/newsfeeds/cities-likes-facebook-top-50/.
[15] Í. Goiri, W. Katsak, K. Le, T.D. Nguyen, R. Bianchini, Parasol and greenswitch: managing datacenters powered by renewable energy, in: ACM SIGARCH Computer Architecture News, vol. 41, ACM, 2013, pp. 51–64.
[16] T. Guérout, T. Monteil, G. Da Costa, R.N. Calheiros, R. Buyya, M. Alexandru, Energy-aware simulation with DVFS, Simul. Model. Pract. Theory 39 (2013) 76–91.
[17] M.S. Hasan, F. Alvares, T. Ledoux, J.-L. Pazat, Investigating energy consumption and performance trade-off for interactive cloud application, IEEE Trans. Sustain. Comput. 2 (2) (2017) 113–126.
[18] S. Iserte, J. Prades, C. Reaño, F. Silla, Increasing the performance of data centers by combining remote GPU virtualization with slurm, in: 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid, 2016, pp. 98–101.
[19] A. Khosravi, L.L. Andrew, R. Buyya, Dynamic vm placement method for minimizing energy and carbon cost in geographically distributed cloud data centers, IEEE Trans. Sustain. Comput. 2 (2) (2017) 183–196.
[20] S. Kim, S. Park, Y. Kim, S. Kim, K. Lee, VNF-EQ: dynamic placement of virtual network functions for energy efficiency and QoS guarantee in NFV, Cluster Comput. 20 (3) (2017) 2107–2117.
[21] X. Li, X. Jiang, P. Garraghan, Z. Wu, Holistic energy and failure aware workload scheduling in cloud datacenters, Future Gener. Comput. Syst. 78 (2018) 887–900.
[22] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, C. Hyser, Renewable and cooling aware workload management for sustainable data centers, in: ACM SIGMETRICS Performance Evaluation Review, vol. 40, ACM, 2012, pp. 175–186.
[23] J.D. Moore, J.S. Chase, P. Ranganathan, R.K. Sharma, Making scheduling cool: temperature-aware workload placement in data centers, in: Proceedings of the USENIX annual technical conference, General Track, 2005, pp. 61–75.
[24] G. Neglia, M. Sereno, G. Bianchi, Geographical load balancing across green datacenters: a mean field analysis, ACM SIGMETRICS Perform. Eval. Rev. 44 (2) (2016) 64–69.
[25] S. Newman, Building Microservices, O'Reilly Media, Inc., 2015.
[26] T.H. Nguyen, M.D. Francesco, A. Yla-Jaaski, Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers, IEEE Trans. Serv. Comput. (ISSN: 1939-1374) (2018) 1–14.
[27] J. Prades, C. Reaño, F. Silla, CUDA acceleration for xen virtual machines in infiniband clusters with rCUDA, in: Proceedings of the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, in: PPoPP '16, ACM, New York, NY, USA, ISBN: 978-1-4503-4092-2, 2016, pp. 35:1–35:2, http://doi.acm.org/10.1145/2851141.2851181.
[28] J. Prades, F. Silla, Turning GPUs into floating devices over the cluster: the beauty of GPU migration, in: 2017 46th International Conference on Parallel Processing Workshops, ICPPW, 2017, pp. 129–136, ISSN 1530-2016.
[29] S.S. Raza, I. Janajreh, C. Ghenai, Sustainability index approach as a selection criteria for energy storage system of an intermittent renewable energy source, Appl. Energy 136 (2014) 909–920.
[30] F.D. Rossi, M.G. Xavier, C.A. De Rose, R.N. Calheiros, R. Buyya, E-eco: performance-aware energy-efficient cloud data center orchestration, J. Netw. Comput. Appl. 78 (2017) 83–96.

[31] J. Shuja, K. Bilal, S.A. Madani, M. Othman, R. Ranjan, P. Balaji, S.U. Khan, Survey of techniques and architectures for designing energy-efficient data centers, IEEE Syst. J. 10 (2) (2016) 507–519.
[32] J. Shuja, A. Gani, S. Shamshirband, R.W. Ahmad, K. Bilal, Sustainable cloud data centers: a survey of enabling techniques and technologies, Renew. Sustain. Energy Rev. 62 (2016) 195–214.
[33] F. Silla, J. Prades, S. Iserte, C. Reaño, Remote GPU virtualization: is it useful? in: 2016 2nd IEEE International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era, HiPINEB, 2016, pp. 41–48.
[34] L. Tomás, C. Klein, J. Tordsson, F. Hernández-Rodríguez, The straw that broke the camel's back: safe cloud overbooking with application brownout, in: Proceedings of the 2014 IEEE International Conference on Cloud and Autonomic Computing, 2014, pp. 151–160.
[35] Tomorrow, Electricitymap — live CO2 emissions of electricity consumption, 2019, https://www.electricitymap.org.
[36] A.N. Toosi, C. Qu, M.D. de Assunção, R. Buyya, Renewable-aware geographical load balancing of web applications for sustainable data centers, J. Netw. Comput. Appl. 83 (2017) 155–168.
[37] B. Varghese, J. Prades, C. Reaño, F. Silla, Acceleration-as-a-service: exploiting virtualised GPUs for a financial application, in: 2015 IEEE 11th International Conference on e-Science, 2015, pp. 47–56.
[38] L. Wang, G. Von Laszewski, J. Dayal, F. Wang, Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with DVFS, in: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, IEEE Computer Society, 2010, pp. 368–377.
[39] C.-M. Wu, R.-S. Chang, H.-Y. Chan, A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters, Future Gener. Comput. Syst. 37 (2014) 141–147.
[40] M. Xu, R. Buyya, Energy efficient scheduling of application components via brownout and approximate markov decision process, in: M. Maximilien, A. Vallecillo, J. Wang, M. Oriol (Eds.), Service-Oriented Computing, Springer International Publishing, Cham, ISBN: 978-3-319-69035-3, 2017, pp. 206–220.
[41] M. Xu, R. Buyya, Brownout approach for adaptive management of resources and applications in cloud computing systems: a taxonomy and future directions, ACM Comput. Surv. (ISSN: 0360-0300) 52 (1) (2019) 8:1–8:27, http://doi.acm.org/10.1145/3234151.
[42] M. Xu, A.V. Dastjerdi, R. Buyya, Energy efficient scheduling of cloud application components with brownout, IEEE Trans. Sustain. Comput. 1 (2) (2016) 40–53.
[43] M. Xu, W. Tian, R. Buyya, A survey on load balancing algorithms for virtual machines placement in cloud computing, Concurr. Comput.: Pract. Exper. 29 (12) (2017) 4123–4138.
[44] M. Xu, A.N. Toosi, R. Buyya, Ibrownout: an integrated approach for managing energy and brownout in container-based clouds, IEEE Trans. Sustain. Comput. (ISSN: 2377-3782) 4 (1) (2019) 53–66.
[45] K. Zheng, X. Wang, L. Li, X. Wang, Joint power optimization of data center network and servers with correlation analysis, in: Proceedings of the 2014 IEEE Conference on Computer Communications, INFOCOM, 2014, pp. 2598–2606.

Minxian Xu received the B.Sc. degree in 2012 and the M.Sc. degree in 2015, both in software engineering from University of Electronic Science and Technology of China. In 2019, he got his Ph.D. degree from the University of Melbourne, Australia. His research interests include resource scheduling and optimization in cloud computing with special focus on energy efficiency. He has co-authored several peer-reviewed papers published in prominent international journals and conferences, such as ACM Computing Surveys, IEEE Transactions on Sustainable Computing, IEEE Transactions on Automation Science and Engineering, Journal of Systems and Software, Concurrency and Computation: Practice and Experience, International Conference on Service-Oriented Computing. His Ph.D. Thesis was awarded the 2019 IEEE TCSC Outstanding Ph.D. Dissertation Award.

Rajkumar Buyya is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is also serving as the founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in Cloud Computing. He served as a Future Fellow of the Australian Research Council during 2012–2016. He has authored over 625 publications and seven text books including "Mastering Cloud Computing" published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese and international markets respectively. He is one of the highly cited authors in computer science and software engineering worldwide (h-index=123, g-index=271, 79,800+ citations). Dr. Buyya is recognized as a "Web of Science Highly Cited Researcher" in 2016, 2017 and 2018 by Thomson Reuters, a Fellow of IEEE, and Scopus Researcher of the Year 2017 with Excellence in Innovative Research Award by Elsevier for his outstanding contributions to Cloud computing. He served as the founding Editor-in-Chief of the IEEE Transactions on Cloud Computing. He is currently serving as Co-Editor-in-Chief of Journal of Software: Practice and Experience, which was established over 45 years ago. For further information on Dr. Buyya, please visit his cyberhome: mailto:www.buyya.com