Elastic Power Utilization in Sustainable Micro Cloud Data Centers

Tuhin Chakraborty, Adel N. Toosi, *Member, IEEE,* and Carlo Kopp, *Senior Member, IEEE*

Abstract—Efficient utilization of renewable energy when powering Cloud Data Centers is a challenging problem due to the variable and intermittent nature of both workload demand and renewable energy supply. This work aims to develop an innovative dynamic resource management algorithm to provide energy flexibility to data center operators for shaping their energy demand to match renewable energy supply. We present a novel framework, called *Elastic Power Utilization* (*EPU*), to serve this purpose. *EPU* utilizes energy source information to dynamically manage data center resources for matching the renewable energy supply with the energy demand to serve the workload. We propose a resource management algorithm that exploits overbooking, consolidation and migration of virtual machines (VMs) to implement the power elasticity required by the *EPU* framework. We compare our approach to a state-of-the-art algorithm and baseline approaches with three different workloads. The results from extensive simulations show that our proposed algorithm outperforms the state-of-the-art approach in saving brown energy by 23.1%, 21.3%, and 27.0% for *Google*, *Wikipedia*, and *Nectar* workloads, respectively.

Index Terms—Green computing, renewable energy, cloud computing, data centers, overbooking, VM consolidation.

1 Introduction

Utility computing, exemplified by Cloud Computing technology, continues to displace traditional methods of deploying centralized computing services. This reflects the elasticity properties inherent in clouds that can flexibly and transparently accommodate large transient variations in workload. As with all technologies, benefits usually incur costs, and for Cloud Data Centers (CDCs), this cost is in power consumption. In this work, we introduce a generalized model for *Elastic Power Utilization* (*EPU*), which we define as the ability to elastically scale power consumption by managing CDC resource usage.

Globally, CDCs are estimated to consume 8000 TWh of electrical energy by 2030 [1] and account for 3.2% of the total worldwide carbon emission by 2025 [2]. Thus, energy-efficient resource management in CDCs and the replacement of brown energy sources with clean energy represent a promising approach to decrease recurring operating expenditures and environmental impacts.

Power management in data centers has been a high priority research topic over the last decade, in areas such as managing cooling systems [3] [4] and servers [5] [6] [7]. One of the primary causes of poor power efficiency in CDCs is the sizing of infrastructure resources like servers and switches, as these are typically provisioned for peak workloads and, hence, remain under-utilized at other times. In addition, clients also request more resources than needed leading to server utilization being as low as 10-30%, reported by many data centers globally [8] [9]. This results in significant wasted power. While different techniques such as VM migration and consolidation have been applied to improve power efficiency in servers, these were not devised for mixed sources of power (i.e., renewable and grid) and seldom produced convincing results [10], [11].

Powering data centers with energy derived from clean (renewable) sources to reduce carbon footprint has been the subject of many research studies. Since the technology for generating renewable electric power continues to improve and rapidly growing, many IT service providers have shifted to using clean energy sources, often using on-site renewable power generation [12]. However, the time-variant and intermittent nature of both workload demand and renewable energy supply make the use of renewable energy sources challenging [12] [13] [14]. CDC operators can try to minimize their utilization of brown energy by maximizing the usage of renewable energy when available.

This work promotes on-site renewable generation without local energy storage in CDCs to accelerate decarbonization and greener services [15]. Deploying such an on-site renewable setup better fits small-scale (micro) data centers (μCDCs) at the edge locations. So, in particular, we focus on μCDCs at edge locations with on-site renewable generation without local energy storage (e.g., battery storage). The use of such storage incurs non-recurring initial and recurring cost overheads. While battery backup uninterruptible power supplies are widely employed, the cost of expanding such an installation to support many hours of CDC operation can be significant and will scale with the size of the CDC. So the option of using less environmentally friendly non-renewable sources is an economic choice intended to minimise CDC installation non-recurring and recurring costs. Note that, research by Karimi et al. [14] and Goiri et al. [16] specifically avoided battery usage due to its high cost and potential for adverse impacts on the environment which brings new sustainability challenges. Thus, we avoided batteries in this investigation and as the renewable energy supply is variable and intermittent, we considered purchasing power from the grid (brown energy) with a

• *T. Chakraborty, A. N. Toosi and C. Kopp are with the Faculty of Information Technology, Monash University, VIC, Australia.*
E-mail: tuhin.chakraborty@monash.edu

Manuscript received April dd, 202x; revised Month dd, 202x.

flexible rate (peak, off-peak) as a backup support when not enough renewable energy is available. We consider inverters to mitigate the challenges of combining different energy sources. These inverters can supply the power generated from renewable sources, send back the excess power to the grid when production exceeds the requirement, and utilize grid power when renewable energy is not enough to handle load.

To adapt to the variable nature of renewable energy supply, we apply *EPU* to favor renewable over non-renewable power sources. This differs from many existing methods [7] [10] [9], which solely focus on reducing the overall power consumption without taking into account the source of the power. To implement *EPU*, we propose a resource management scheme that shapes workload power demand by dynamically adapting the overbooking level to match the available clean energy supply. *EPU* increases the overbooking level, the ratio of overcommitting resources when renewable energy is scarce, and minimizes overbooking when renewable energy is abundant.

Overbooking is defined as the capability to accommodate more virtual machines on a physical server, where the total capacity of the physical server is below the sum of the requested resources by hosted VMs. Overbooking is a common practice in the CDC domain. Recently, VMWare, one of the giants of virtualization technology, reported that considerable levels of CPU over-commitment is possible without significantly impacting VM performance [17]. For instance, a general guide for performance sizing as per best practice recommendations permits allocation up to three times of the total available CPU resources in a host (200% overbooking) [18]. Following industry practice, we focus on overbooking of CPU resources since processors are the main power consumers in a server [19]. We also assume that enough memory and storage are available for the targeted overbooking level.

To achieve our objectives, we exploit overbooking to favor green energy, where CDC operators and CDC clients maximize their individual and collaborative gain. In other words, CDC clients can receive incentives (e.g., discount, or carbon tax benefits) when they are exposed to overbooking, and CDC operators can exploit overbooking and consolidation techniques to favor renewable power and reduce brown energy use. When power from renewable sources is scarce, clients will be allocated more VMs per single physical server (share fewer resources than the actual demand), which might affect their Quality of Service (QoS), impacting end users. The compromised QoS due to overbooking is acceptable as part of their Green Service Level Agreements (SLA). A Green SLA can be considered as a joint agreement between CDC clients and CDC operators to accept overbooking (lower QoS) while the data center uses energy from brown sources. Accepting incentives such as discounts for a lower QoS is acceptable for many use cases, like when they are probable to remain underutilized or for applications that can tolerate some delay or performance loss. For example, stateless web servers, big data analytics applications, rendering workloads, and other flexible workloads can use discounted spot instances of AWS that are terminated anytime, i.e., a lower QoS compared to on-demand instances [20].

To further clarify, the key impact of this research is to favor green energy through an efficient resource management scheme by exploiting overbooking. To achieve our objectives, the main contributions are summarized thus:

- An energy-source-aware architecture for CDC resource management, based on the concept of EPU.
- Innovative dynamic resource management and overbooking algorithms to provide energy flexibility to μCDCs which shape energy demand to match renewable energy supply.
- Evaluation of the proposed algorithms through simulations using real-world renewable energy and workload traces.
- Analysis of the efficiencies of the proposed algorithms, in terms of reducing carbon footprint, and demonstration of their superiority over several baseline and state-of-the-art approaches.

The remainder of the paper is organized thus: we explain the background for the green-energy-aware resource management of sustainable CDCs with a typical example scenario and system architecture in Section 2. Section 3 presents the energy models employed and the formalization of the problem, followed by the proposed algorithm and the baselines in Section 4. The results are presented in Section 5, followed by conclusions and future directions in Section 6.

2 Related work and Motivation

Many different approaches have been investigated to address the problems inherent in the utilization of renewables in CDCs [21] [16]. However, the primary challenge when using renewable energy remains - it is by its nature a time variant power source. Some previous studies target load redirection among multiple data centers [22] [23] to address this challenge. Although significant attention has been paid to the energy efficiency of data center [24] and Power Usage Effectiveness (PUE) of 1.21 has been reported by Google [25], little attention has been given to elastically shaping the data center load to align it with renewable energy availability. In recent years, significant attention has been given to the development of VM placement and consolidation strategies to reduce CDC power consumption [31] [9]. VM consolidation effectively improves resource utilization and energy efficiency by gathering virtual machines into a minimal subset of physical hosts. Moreover, powering off physical host machines that would be otherwise idle will also save cooling power.

Most power consumption in a CDC network fabric arises from switching components when powered on [32]. Powering off only unused ports saves very little energy, at best 1-2 Watts per port [32]. The concept of ElasticTree [32] is based on this idea, but this concept did not consider hypervisors and VMs and did not power off edge switches as we do in this work. Hence, it still suffers from some unnecessary power consumption in both its computing and network stacks.

Liu et al. [22] explored geographic load balancing and storing the energy from renewable sources to minimize brown energy consumption. Lin et al. [33] extended this work to achieve a net-zero brown energy system by combining brown and green energy sources. Aiming to solve the

TABLE 1: A brief comparison with existing literature

Work	Brown energy	Green energy	Migration	Overbooking	On-site renewable
Celesti et al. [26]	✓	✓	✗	✗	✗
Le et al. [27]	✓	✓	✗	✗	✗
Farahnakian et al. [28]	✓	✗	✓	✗	✗
Li et al. [29]	✓	✓	✗	✗	✓
Toosi et al. [12]	✓	✓	✗	✗	✓
Son et al. [9]	✓	✗	✓	✓	✗
Hasan et al. [30]	✓	✓	✗	✗	✓
Goiri et al. [16]	✓	✓	✗	✗	✓
Khosravi et al. [13]	✓	✓	✓	✗	✓
our work	✓	✓	✓	✓	✓

same problem, Toosi and Buyya [34] proposed a load balancing algorithm based on fuzzy logic that does not require *a priori* knowledge. These works are based on routing incoming loads as per their power state when receiving job requests. However, none of these works exploited renewable-energy-aware VM management within CDCs based on overbooking techniques.
Khosravi et al. [13] considered migration of VMs between CDC sites based on the limited and intermittent nature of renewable energy aiming for better utilization. However, their work does not consider opportunities to apply VM or workload management inside individual CDCs. Some earlier work [10], [31], and more recent work [9], [28] mainly focused on reducing the number of active physical machines to improve system power efficiency. Farahnakian et al. [28] considered future resource demands and applied VM consolidation based on a utilization prediction model. Though these works can save energy by better managing active physical machines, aside from Son et al. [9], none have considered network energy and overbooking resources as a management technique.
Son et al. [9] did not consider cooling power as a part of their management technique. Energy supply and source awareness were not a part of their resource management framework, so release overbooking was not considered when it does not affect carbon emission. Notable all omitted awareness of green energy sources as a part of energy efficiency.
Another approach is to defer power-hungry workloads to align them with renewable energy availability. Goiri et al. [16] presented a prototype of a small-scale green CDC and proposed *GreenSwitch* to explore the ability to delay MapReduce jobs when the workload can be deferred. Hasan et al. [30] proposed green energy awareness for interactive cloud applications by compromising non-core and independent features of the services that can be isolated to be activated or deactivated. These works primarily focus on tuning the application level of the given services but do not consider green awareness as a core component of the system framework. On the other hand, our *EPU* method performs resource management in accordance with energy supply from green sources. Table 1 gives a brief comparison of our proposed work with existing schemes in the literature.

A common limitation of the schemes reported above includes not explicitly managing access to renewable or non-renewable power sources to manage CDCs. None of these works exploited VM management using awareness of energy sources within CDCs based on overbooking techniques. We overcome this limitation by allowing more efficient use of renewable power if available, focusing on μCDC with on-site renewable generation. This paper presents a sustainable system architecture that focuses on dynamic overbooking techniques and consolidation of VMs within the μCDC to provide *Elastic Power Utilization*. Hence, we present a system architecture able to elastically shape workloads to best match available power from green sources.

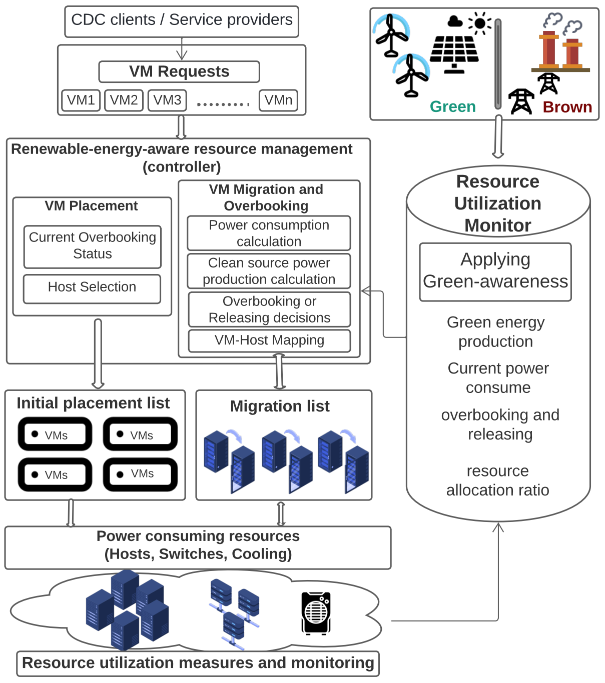

CDC clients / Service providers
VM Requests
VM1
VM2
VM3
VMn
Green
Brown
Renewable-energy-aware resource management (controller)
VM Placement
Current Overbooking Status
Host Selection
VM Migration and Overbooking
Power consumption calculation
Clean source power production calculation
Overbooking or Releasing decisions
VM-Host Mapping
Resource Utilization Monitor
Applying Green-awareness
Green energy production
Current power consume
overbooking and releasing
resource allocation ratio
Initial placement list
VMs
Migration list
Power consuming resources (Hosts, Switches, Cooling)
Resource utilization measures and monitoring

Fig. 1: Renewable-energy-aware Resource Allocation

2.1 System Overview

In our proposed *Elastic Power Utilization* shown in Fig. 1, we present a management system to align μCDC power consumption to match renewable energy availability. We employ dynamic VM consolidation and computing resource overbooking that accounts for green power sources. The primary idea of this architecture is to utilize the information provided by resource utilization monitor to manage resources in μCDC. Based on this information, the required energy consumption of the μCDC to serve the workload is calculated and accordingly a resource management scheme is applied to match power consumption with available green power generation. As stated earlier, favoring green sources, we assume CDC clients will accept resources to be overbooked to some level based on the Green SLA when the

energy supply from green sources is inadequate.
CDC Clients/Service providers request VMs to be executed in the CDC as the host for their applications. *Controller* receives these requests, and with the help of monitored data, initially places VMs, applies overbooking, releases overbooking, and performs VM migration dynamically if required. This way it matches the power consumption of the μCDC with power availability from green sources, which are variable and intermittent. The *VM placement* component checks the current status of the hosts and decides where to launch a new VM when a request initially arrives into the system. The *VM migration component and overbooking* component decides upon the migration and overbooking of VMs when required, the selection of source and destination hosts, and the maintenance of the *VM-to-host mapping* lists for the framework. It also refers to a migrations list that is created based on the monitored data.
The *Resource Utilization Monitor* is in charge of monitoring the resource utilization and the associated energy consumption levels. It receives all the relevant data for monitoring purposes and runs all the resource utilization and power consumption models to keep track of the resources. The resource utilization data are analyzed along with the energy available from the clean sources to decide when to perform overbooking, release overbooking, and migration methods for matching the energy requirement with the supply of clean energy sources.
In a typical CDC scenario, CDC clients are service providers who deliver applications to end-users, and they subscribe resources from CDC operators to run and manage end-users applications. We propose an agreement between CDC clients and CDC operators; we call it *Green SLA*, as noted earlier. This agreement can influence both parties to adapt greener services knowingly and mutually. On one hand, accepting this agreement, the service providers agree to endure some levels of overbooking from the CDC operator. In return, CDC clients can receive incentives (e.g., discount, or carbon tax benefits) when they are exposed to overbooking. For many services commonly remains underutilized, or applications that can handle occasional overbooking, *Green SLA* provides a suitable scope. It is worth mentioning that overbooking may result in QoS degradation (e.g., delayed response time) for end users of CDC service providers. However, it is up to CDC clients to make sure if *Green SLA* is right for them and manage their resources accordingly. On the other hand, the CDC operators exploit these opportunities for their overbooking technique to save energy, when grid power is required to run their workload. Note that, the common practice is that the CDC operators apply overbooking without informing clients and allow considerable levels of CPU over-commitment without significantly impacting VM performance [17]. For example, VMWare permits up-to three-fold overbooking as a general guide for performance sizing as per best practice recommendations [18]. So, the proposed *Green SLA* motivates both parties participating in a transparent process to accelerate decarbonization and greener services.

2.2 Motivational Example

This subsection presents our objective by providing a motivational example. On the left part of Fig. 2, we have a group of VMs hosted by the corresponding physical servers (hosts). We assume CDC clients are willing to accept some level of overbooking when possible, as shown by the blue area in the figure. As stated earlier, the client's motivations for accepting overbooking can be various incentives such as discounts from cloud operators, being environmentally friendly, or carbon tax benefits. Accordingly, clients provide the level of overbooking to which they are willing to accept a compromised ratio of resource demands when green energy is inadequate, i.e., in the form of an upper bound for overbooking level. The way CDC clients set acceptable overbooking level, and design of incentive mechanisms fall out of the scope of this work.

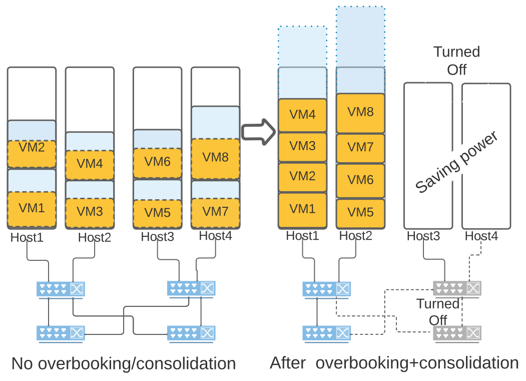


Fig. 2: Example of consolidation with overbooking. The blue part areas in VMs denote how much overbooking is allowed compared to the base requested amount of resource for the VMs.

Energy components like hosts and switches consume considerable power even when serving a minimum workload as well. This happens because of the base idle power consumption, which is a constant factor consumed by the host no matter how much load it serves [9]. This work aims to save this energy as much as possible. The idea is to minimize the number of active components as much as possible by exploiting overbooking (allowing more VMs to be packed in a host) and switching off unused components to save power when green energy is insufficient. On the right part of Fig. 2, we show that the joint application of overbooking and consolidation can lead to lower resource usage and save power consumption. Section 3 formulates this objective.

3 Problem Formulation

This section presents energy models for the different components of a CDC, followed by the formulation of the primary objective addressed in this work.

3.1 Energy Model

3.1.1 Compute Energy Model

The amount of power consumed by a server is often varied linearly with the CPU utilization level, in addition to the base idle power consumption. To model CPU power usage, we adopt the popular power model from [19] [35]. This power model is also being adopted by other recent works such as [9] [36] [37] [28]. Thus, the power consumption of host i is modeled as:

$$P\left(h_i\right)=\begin{cases} P_{idle}+\left(P_{\text{peak}}-P_{\text{idle}}\right)\cdot u_i & \text{if } \alpha_i>0 \\ 0 & \text{if } \alpha_i=0 \end{cases} \quad (1)$$

TABLE 2: Notations used for problem formulation

Symbols	Description
h_i	The i^{th} host in the CDC
H	The set of all hosts in the CDC, $\forall i, h_i \in H$
$\lvert H \rvert$	The total number of hosts in the data center
s_i	The i^{th} switch in the CDC
S	The set of all network switches in the CDC, $\forall i, s_i \in S$
$\lvert S \rvert$	The total number of network switches in the data center
$P(h_i)$	Power consumption of host i
P_{idle}	Idle power consumption of host
P_{peak}	Peak power consumption of host
u_i	CPU utilization percentage of host i
α_i	The number of VMs placed in host i
$P(s_i)$	Power consumption of switch i
$\lvert VM \rvert$	The total number of VMs in the CDC at any given point of time

where P_{idle} and P_{peak} are the power consumption of a host in its idle and peak states, which are constant factors, u_i is the dynamic power consumption of host, which is linearly related to the CPU utilization percentage, and α_i is the number of VMs placed in host i.

The *compute* subsystem of a CDC comprises a set of hosts. Its energy consumption during time slots 0 to T is determined by the equation 2:

$$P_C = \sum_{t=0}^{T}\left(\delta_t \cdot \sum_{i=1}^{|H|} P(h_i)\right) \tag{2}$$

where $|H|$ is the total number of hosts and δ_t is the duration of the respective time slot.

3.1.2 Network Energy Model

Network power consumption is mainly determined by its switching components when they are active [38] [32] [39]; going from zero to full traffic increases power consumption by only less than 8% [32]. So, turning off a switch can give the most benefits, while an unused port only can save 1-2 Watts. Also, *Network* subsystem power consumption in a CDC is relatively low compared to the *compute* subsystem and typically comprises $\approx$ 5% of the total power consumption [19] of a data center. Hence, we ignore the power variation due to traffic as it can contribute only up to a negligible fraction. Thus, we adopt the power consumed by the network is predominantly determined by the number of active (On) switches. The power consumption of any switch s_i, $P(s_i)$, is modeled as:

$$P(s_i) = \begin{cases} P_{sw} & \text{if } s_i \text{ is } ON \\ 0 & \text{if } s_i \text{ is } OFF \end{cases} \tag{3}$$

where P_{sw} is the power consumption of the switch.

Hence *network*, using Equation 3, we compute the network subsystem energy consumption over time period 0 to T by the given Eq. 4:

$$P_S = \sum_{t=0}^{T}\left(\delta_t \cdot \sum_{i=1}^{|S|} P(s_i)\right) \tag{4}$$

3.1.3 CRAC Model

The thermal management of a CDC is performed by the Computer Room Air Conditioning (CRAC) subsystem. We assume the only cooling system facility available in the data center is the CRAC unit. We adopt the widely used model from [40] to estimate CRAC subsystem power consumption, also employed in other recent work [41]. According to this model, the efficiency of a CRAC subsystem is measured by the Coefficient of Performance (CoP) metric, that is a function of cold air supply temperature T_s. CoP is defined as the ratio of the total power dissipated by the compute subsystem to the total power consumed by the CRAC system to extract the dissipated heat.

$$\mathrm{P_{CRAC}} = P_C / CoP(T_\mathrm{s}) \tag{5}$$

where P_C and P_{CRAC} represent compute and cooling system power. Cooling system power can be reduced by decreasing the compute load or increasing the cold air supply temperature. We adopt the following regression model from [40] to estimate CoP:

$$\mathrm{CoP}(T_\mathrm{s}) = \alpha T_\mathrm{s}^2 + \beta T_\mathrm{s} + \gamma. \tag{6}$$

where α = 0.0068, β = 0.0008, and γ = 0.458. Equations 5 and 6 indicate that the increasing value of T_s leads to reduce the cooling power when compute system power remains the same. Thus, the energy consumed by the cooling system over the time slots 0 to T is determined as a function of cold air supply temperature T_s and the amount of power consumption P_C by the compute system and can be written as equation:

$$P_{CRAC}(T_s, P_C) = \frac{\sum_{t=0}^{T}\left(\delta_t \cdot \sum_{i=1}^{|H|} P(h_i)\right)}{\mathrm{CoP}(T_\mathrm{s})} \tag{7}$$

Thus, the total energy consumption by the compute, network and CRAC system of data center over the time slots 0 to T can be formulated as:

$$P_T = P_S + P_C + P_{CRAC}(T_s, P_C) \tag{8}$$

That is:

$$P_\mathrm{T} = \sum_{t=0}^{T}\left(\delta_t \cdot \sum_{i=1}^{|S|} P(s_i)\right) + \left(\frac{\alpha T_\mathrm{s}^2 + \beta T_\mathrm{s} + \gamma + 1}{\alpha T_\mathrm{s}^2 + \beta T_\mathrm{s} + \gamma}\right) \cdot \sum_{t=0}^{T}\left(\delta_t \cdot \sum_{i=1}^{|H|} P(h_i)\right) \tag{9}$$

3.2 Problem Formulation

We employ a resource management scheme that aims to minimize the utilization of grid power from brown sources when the supply from green sources is inadequate over the time slots 0 to T. This is done by exploiting overbooking and releasing overbooking according to the power availability from green sources. The primary objective can be formulated as:

$$\text{minimize: } \sum_{t\in T}\left(\delta_t \cdot \sum_{si\in S} P(s_i)\right) + \left(1 + \frac{1}{CoP(T_\mathrm{s})}\right) \cdot \sum_{t\in T}\left(\delta_t \cdot \sum_{h_i\in H} P(h_i)\right) \tag{10}$$

subject to constraints:

$$\sum_{i=1}^{|H|} \alpha_i = |VM| \tag{11}$$

where T can persist for as long as we employ the scheme.

Our proposed solution relies on joint VM consolidation and overbooking in accordance with *Green SLA* requirements. We solve (10) as a resource management problem, with dynamic overbooking of resources to maintain energy efficiency throughout the process. In the next, we propose our algorithm along with a few baselines experimental comprehensiveness.

4 Overbooking and VM Consolidation Exploiting Renewables

Our approach focuses on green-aware shaping of workload and managing the resources dynamically to match the renewable energy supply. This work jointly exploits the overbooking and consolidation of VMs together to shape the workload. Our workload shaping can make the system more power elastic. As we exploit overbooking, this mechanism allows CDC operators to accommodate more virtual machines on a physical server (host) than in actual or non-overbooked states. When we implement overbooking, a host's capacity is increased as per the overbooking ratio described in the following equation.

$$R^{A}_{GOBR}(h_i) = (1 + GOBR) \times R(h_i) \tag{12}$$

where $R(h_i)$ is the available resource of host h_i in its non-overbooked state, $R^{A}_{GOBR}(h_i)$ is the capacity of host h_i when overbooking is applied, and $GOBR$ is the green energy aware overbooking ratio. We named it Green Overbooking Ratio (GOBR) as we implement overbooking to reduce the demand for brown energy when the clean energy supply is inadequate. Now, we explain our algorithm as follows.

4.1 Renewable Energy-Aware Overbooking & VM consolidation

This renewable-energy-aware resource management algorithm implements our proposed concept of *EPU*. It exploits overbooking (OB), VM migration (Mig), along with the launching of VMs on the most loaded host with enough remaining capacity following a best fit (BF) policy. We named this algorithm as $BFOBMig$. In Algorithm 1, we first gather the required initial information and initialize state variables (Inputs & Steps 1-2). These are monitoring information used throughout the algorithm, such as the number of hosts in the μCDC, a list that tracks the number of VMs in all hosts, etc. There are K number of pods in an order K DCN topology; each pod consists of $K/2$ edge switches, each connected with $K/2$ hosts. So, the total number of hosts is $K \times K/2 \times K/2 = K^3/4$. We initiate the required counters to keep track of event times, like launching and termination of VMs, placements, and power status (Steps 3-6). Our algorithm continuously checks if any new request for launching or termination of a VM has arrived or if the duration for the current time slot has expired. This algorithm manages two main parts while continuously checking system state. One part (Steps8-18) is responsible for managing launch and termination requests of VM instances in the μCDC. When receiving a $NewRequestReceived$ event, it detects the request type, and whether a launch or termination is requested. New VM launch requests are executed as per the current status of the hosts (Steps9-12). The VM termination requests are managed as per Steps 13-16. $NewRequestReceived$ event iterates over all hosts to manage resources and has time complexity of $O(|H|)$. The second part (Steps19-21) calls Algorithm 2.

Algorithm 2 manages the implementation of overbooking and VM consolidation. This algorithm initiates with updated global variables, lists and counters from Algorithm 1. At the end of each time slot, it recalculates the total energy consumption (Steps 1-2) as per the models explained in Section 3.1. Notably, VM migrations constitute additional energy use, which can be estimated by considering the VM to be active on two hosts during the migration process. This amount of energy is also fed into the model. After calculating the total energy consumption of the slot, it checks the renewable energy supply; if supply is insufficient, it executes migration with overbooking (if that has not already been done) and performs the required update for the resource capacity to the set of hosts (Steps 3-11). If supply is sufficient to serve the workload, the system releases the overbooking and performs the required resource capacity update to the set of hosts (Steps 12-21). All the updated global variables, lists, and counters are returned to Algorithm 1. To manage resources, this algorithm iterates over all hosts, and has time complexity of $O(|H|)$. As stated earlier, Algorithm 1 and 2 perform resource allocation by jointly exploiting overbooking and VM consolidation as per renewable energy availability.

4.2 Baseline Algorithms with Modifications

In the following, we introduce a few baseline and state-of-the-art algorithms, later used for comparison with BFOBMig and show the impact of renewable-energy-aware joint overbooking and VM consolidation.

LB Algorithm: This naïve algorithm tries to distribute VMs evenly among the available hosts—keep the Load Balanced (LB)—and selects the least loaded host to accommodate a newly requested VM. It does not consider VM migration after the initial placement of VMs, and we consider this widely used policy as a baseline algorithm.

BF Algorithm: This algorithm places VMs into the most loaded host, which still has the capacity to accommodate more VMs following the best fit (BF) policy. This is a well-known energy-efficient technique that, similar to LB, does not consider VM migration and is widely used to compare with any modified strategies with add-on constraints. We also consider this policy as a baseline algorithm in this paper.

BFMig Algorithm: Similar to BF, this algorithm initially launches VMs on the most loaded host. Then, it performs VM migration to reduce brown energy use when the available renewable energy is less than required to serve the workload. However, it does not perform overbooking. It calculates the total power consumption at the end of each prefix time slot and implements the VM migration if required as per available supply of renewable energy. A modified version of this policy is also considered for comparison by a recent work [28]. We adopt this policy and modified it as a renewable-energy-aware and brown energy-saving technique by exploiting VM consolidation.

BFOBAlMig Algorithm: This algorithm also launches VMs on the most loaded host following the best fit (BF) policy. It overbooks ($OBAl$) resources all the time and considers VM migration (Mig) at the end of each prefixed time slot, regardless of renewable-energy generation status. We name it $BFOBAlMig$, and it is identical to $BFMig$ algorithm, other than incorporating overbooking.

Algorithm 1 Renewable-energy-aware resource allocation

```
Input: $H$: The set of all hosts in the DC; $K$: The order of the DCN topology; $I_{VM}$: Temp. queue for new VM requests (Launch or Termination); $RE_t$: Renewable energy availability in time-slot t
Output: Distribution of the workload with the timeline
1: $h \leftarrow (K^3)/4$    ▷ The total number of hosts in the CDC
2: $L_h, T_{TS}, P_{TS} \leftarrow \theta$    ▷ List of the VMs under each host, timestamps when event (IN or OUT of VMs) occurs, energy consumption rates by hosts at all time stamps, all initiated to $NULL$
3: $currState \leftarrow NonOB, C \leftarrow C_{NonOB}$    ▷ Initial state is set to non-overbooked state, host capacity C
4: $Count\_In \leftarrow numOf(I_{VM}.LaunchReqs), Count\_Out \leftarrow numOf(I_{VM}.TermReqs)$    ▷ Keeping track of newly coming requests, Count_Out=0 initially
5: $T_{TS}.Append(Start\_Time)$
6: $P_{TS}.Append(Pow(L_h))$    ▷ Energy consumption for initial workload
7: while completing a slot OR NewRequestReceived do
8:     if NewRequestReceived then
9:         if IN request then
10:            $H_{in} \leftarrow Select\ most\ filled\ and\ still\ has\ capacity\ host$
11:            $L_h[H_{in}].push(newLaunchReq)\ in\ currState$
12:        end if
13:        if OUT request then
14:            fetch $VMID$, $HostID$ of $OUT$ request
15:            $L_h[HostID].pop(VMID)$
16:        end if
17:        Update all counters
18:    end if
19:    if end of a slot then
20:        $do\ follow\ Algorithm$ 2
21:    end if
22: end while
```

Algorithm 2 Dynamic overbooking of hosts and VM Consolidation

```
Input: Last updated Lists and Variables from Algorithm 1
Output: Distribution of the workload with the timeline
1: while completing of each time-slot t do
2:     $TotalPow_{TS_t} = Pow^H_{TS_t} + Pow^N_{TS_t} + Pow^C_{TS_t} + Pow^M$    ▷ over uneven time partitioning TS of slot t total Power consumption by Hosts, Network, Cooling, Migrations are $Pow^H_{TS_t}, Pow^N_{TS_t}, Pow^C_{TS_t}, Pow^M$ at the currState (overbooked or non-overbooked)
3:     if $RE_t \leq TotalPow_{TS_t}$ then
4:         if currState is NonOB then
5:             $currState \leftarrow OB$
6:             $C = C_{OB}$
7:             $L^{OB}_h \leftarrow Update\ containing\ capacity\ of\ L_h$
8:         end if
9:         $P = \lceil sum(allLen(L^{OB}_h(i))/C)) \rceil$
10:        $OBMig\_List(Source\#\ L^{OB}_h[0..P] \leftarrow Dest\#\ L^{OB}_h[P+2..len(L^{OB}_h)]$
11:        $L_h \leftarrow L^{OB}_h$
12:    else
13:        if currState is OB then
14:            $currState \leftarrow NonOB$
15:            $C = C_{NonOB}$
16:            $L^{NonOB}_h \leftarrow Update\ containing\ capacity\ of\ L_h$
17:        end if
18:        $P = \lceil sum(allLen(L^{NonOB}_h(i))/C)) \rceil$
19:        $OBMig\_List(Source\#\ L^{OB}_h[0..P] \leftarrow Dest\#\ L^{OB}_h[P+2..len(L^{OB}_h)]$
20:        $L_h \leftarrow L^{NonOB}_h$
21:    end if
22: end while
23: return All updated global variables, counters, lists to Algo1.
```

BFOBAlMig overbooks resources all the time, even when sufficient renewable energy is available. Therefore it violates Green SLA though it minimizes the brown-energy use. We proposed this algorithm as a lower bound baseline.

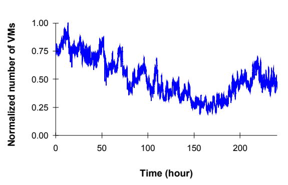
5 Performance Evaluation

We implemented all the methods as described in the previous section, to evaluate and compare their relative performance in minimizing brown energy consumption and Green SLA violations. We discuss the details of the experiments and analysis of the results.

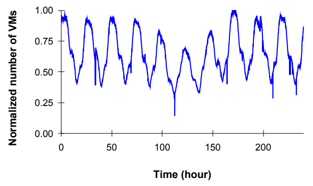
5.1 Experimental Setup

In order to evaluate our approach, we implement the algorithms in a simulation environment described in [37]. We added monitoring components to handle variable renewable energy availability, energy consumption for cooling, and dynamic overbooking methods. The data center simulated for our experiment comprises 128 hosts each using 260W on-peak, connected by a fat-tree network where the switch uses 60W of power, topology of order 8 with 16 hosts in each pod and cold air supply temperature is set to 25°C for the cooling system. The total energy consumption by the hosts, CRAC component, and switches are computed based on the power models explained in Section 3.1. Different compute sets can comprise non-uniform server types in large CDCs, but these physical servers are usually of the same type in a single rack or cluster. In the current work, we assume physical servers and VMs are homogeneous. Each physical server can host up to 34 VMs without overbooking at the peak load. We exploit 30% overbooking level for the experiments and then we vary overbooking level to analyze its impact. We ran all simulations on Intel Core i7-8850H 2.6GHz × 12 CPU, 64-bit computer with 32 GB RAM, running Ubuntu 18.04.3. For experiments with migration policy, the time window to run migration is set to 30 minutes (identical to [9]), so migration is attempted if required as per the availability of renewable energy every 30 minutes.

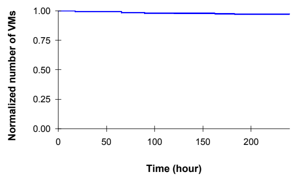
Our experimental setup variables are considered to build a realistic data center environment. Additionally, all experiments are performed on the same setup for equitable evaluation. We employed common and widely used energy



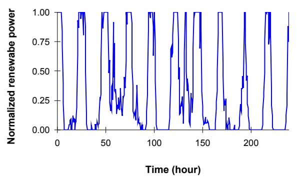
(a) Google workload trace.



(b) Wikipedia workload trace.



(c) Nectar workload trace.



(d) Normalized total power

Fig. 3: Workload and renewable energy traces for 10 days

models to measure the estimated energy consumption. Recent works, such as [36] [37] [9] [41], also adopted these models for similar purposes. The validation of these energy models can be found in the original papers. Please note that it is not our goal to propose accurate energy consumption models of a data center in this paper. In addition, our work focuses more on the components that can directly vary with the computational load, so we discarded other load-invariant components, such as lighting and power control systems, contributing to less than ≈9% of total power in a typical data center [19]. Therefore, the results indicate the estimated values based on the adopted models, and the reported study eliminates all possible explanations other than the relative performance of the compared schemes.

5.2 Workload

We employed *Google Cluster*, *Wikipedia*, and *Nectar Cloud*, traces as representative workloads. All of these are real-world workload traces. *Google Cluster* and *Wikipedia* widely used in the literature and are publicly available [42], [43]. The *Google Cluster* trace includes job requests submitted to a cluster over a period of one month. Google workload represents activities in a cluster of approximately 12,500 compute cells managed by the cluster management software of Google, known as Borg. The trace describes every job submission, scheduling decision, and resource usage data for the jobs that ran in that cluster. We mapped these *job* request traces to generate VM requests similar to [34]. The trace contains nearly 480,000 requests over about one month, and we use the first ten days of data for the experiment. Fig. 3a shows the normalized workload of VM requests generated based on the scheduling algorithm in [12] [34]. We use 4231 number of VMs for the peak and scale the workload accordingly, following works in [12] [34].

Similarly, we depict the shape of the *Wikipedia* trace in Fig. 3b. This trace contains 10% of all *http* requests issued to *Wikipedia* (in all languages) available at [43] [44]. The workload follows a diurnal pattern with clear periods of different workload intensity. We used the requests over ten days (dated Oct 02-11, 2007; about 2119.6M requests) and mapped these request traces to generate VM requests similar to [34]. We normalized the *Wikipedia* workload shape to the peak and then matched its peak to the peak of the *Google* workload to standardise the experimental environment.

Nectar [45] is an Australia's national research cloud, providing cloud computing services and tools to Australian researchers. The traces for the *Nectar Cloud* are extracted from the real usage of resources For the *Nectar* workload, we used the trace over ten days (dated Dec 22-31, 2017), normalized the workload shape to the peak, and then matched its peak to the peak of the *Google* workload to standardize the experimental environment. The Nectar cloud is often used for research projects running scientific batch jobs over long-running VMs. So its workload pattern is consistent throughout the year as shown in Fig. 3c.

5.3 Renewable Energy Traces

To capture renewable energy availability, we adopt the solar irradiation and wind energy near to the University Campus; we use meteorological data traces by *Weather Underground* [46] with half-hour granularity between April 08-17, 2021.

We presume the data center uses a small scale in-house wind turbine set to generate wind power. To estimate the average wind power production per half-hour, we adopt the model in [47] [48] where the wind speed, air pressure, cut-in, and cut-out speed are fed into the model. Similarly, the Global Horizontal Irradiance (GHI) in the same location is used to calculate solar photovoltaic (PV) output power. To estimate the average solar power production per half-hour, we adopt the model in [49] [12] where GHI data, tilt angle of 45°for the panels, calendar dates for the angle of the solar radiation, PV cell efficiency of 30%, and location latitude 38°South are fed into the model. We depict the shape of the renewable power trace in Fig. 3d

5.4 Results and Analysis

This section presents our experimental results. We experimented with the simulation setup described in the previous section (Section 5.1) with ten-day VM request traces of three workloads (Section 5.2) with renewable energy traces described in Section 5.3.

5.4.1 Impact on Brown Energy Consumption

Fig.4, Fig.6, and Fig.8 show the total power consumption throughout ten days for different algorithms compared to renewable power generation. $BFMig$, $BFOBMig$, and $BFOBAlMig$ which are renewable-energy aware, consumed less power to serve workloads than the baseline algorithms LB and BF. We assumed that the operator buys the grid electricity, presumably brown energy, when insufficient renewable energy is available. In all three figures (Fig.4, Fig.6, and Fig.8), *RE* represents the supply from renewable sources, and the power consumption amounts that are non-overlapped with *RE* are supplied from the grid (brown source). We set 30% overbooking to present the results of *BFOBMig* in Figures 4-9. The impact of varying overbooking level is presented in section 5.4.4. The energy demands to serve the workload vary over time due to the policies employed in different algorithms. Unlike the naïve *LB* and the other baseline *BF* algorithms, the *BFMig* and *BFOBMig* algorithms vary to match the renewable energy supply. The energy consumption in *EPU* based *BFOBMig* algorithm varies more, over the other algorithms. *BFOBMig* matched power demand and renewable power supply more than the other algorithms and reduced power requirement to the lowest when power was taken from brown sources. It happened as it applied both VM migration

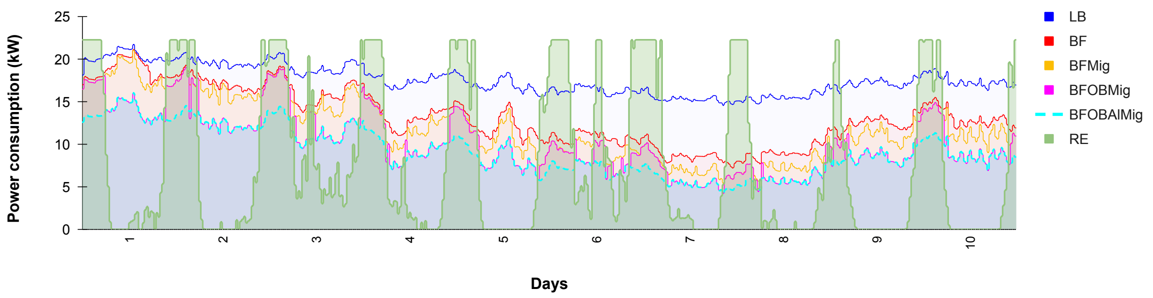
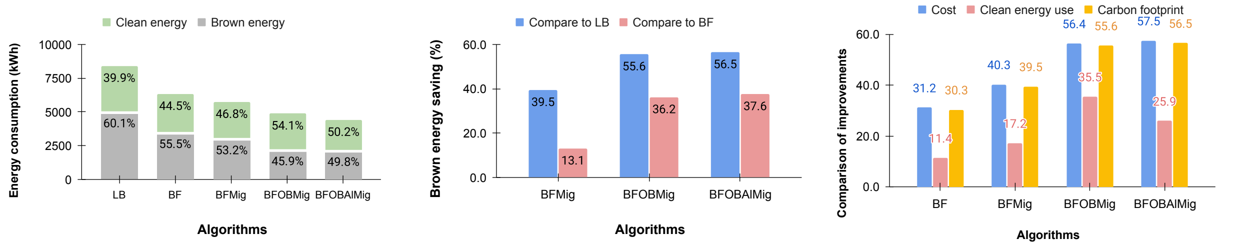


Fig. 4: Power consumption in different algorithms (*Google* workload) and supply from renewable (RE).



(a) The total amount of energy consumption (kWh) of different algorithms with proportion of brown and clean power usage.

(b) Brown energy savings of different algorithms compared to baseline algorithms (LB and BF).

(c) Improvements in carbon footprint, clean energy use, and costs over the naïve LB algorithm.

Fig. 5: Results of algorithms for *Google* workload.

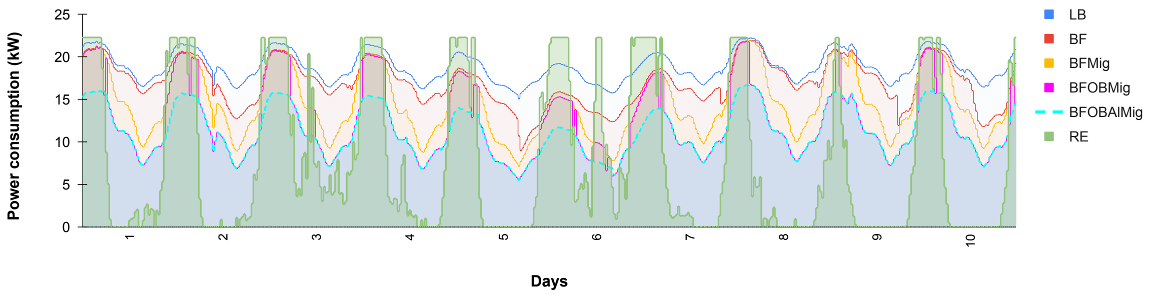
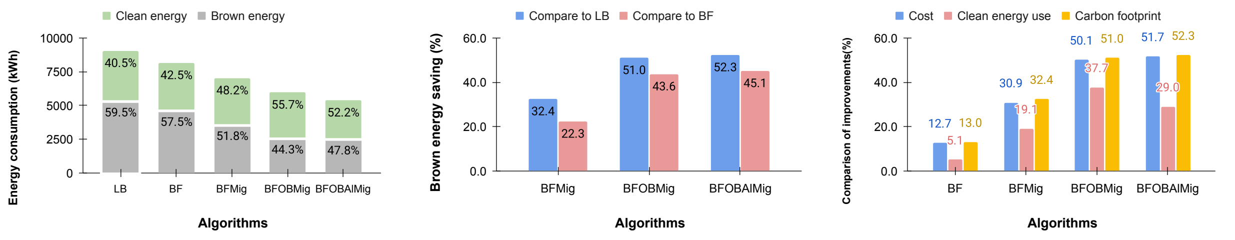


Fig. 6: Power consumption of different algorithms (*Wikipedia* workload) and supply from renewables (RE).



(a) The total amount of energy consumption (kWh) of different algorithms with proportion of brown and clean power usage.

(b) Brown energy savings of different algorithms compared to baseline algorithms (LB and BF).

(c) Improvements in carbon footprint, clean energy use, and costs over the naïve LB algorithm.

Fig. 7: Results of algorithms for *Wikipedia* workload.

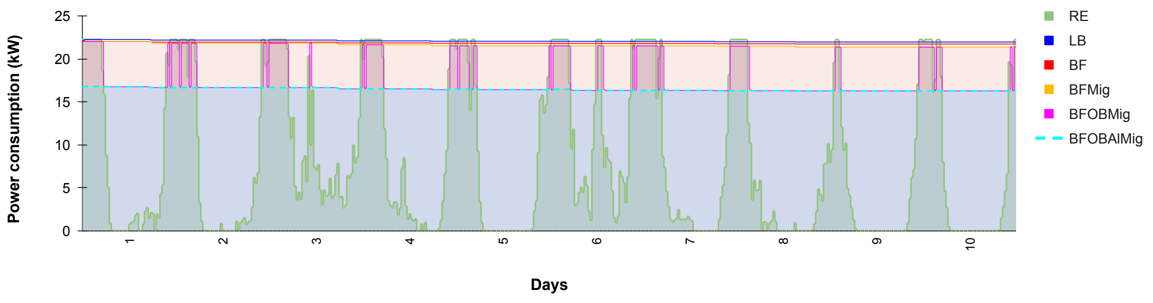
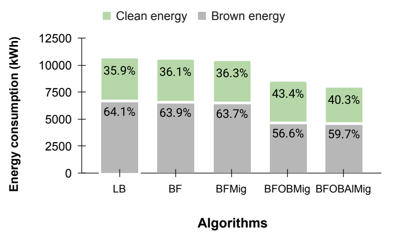
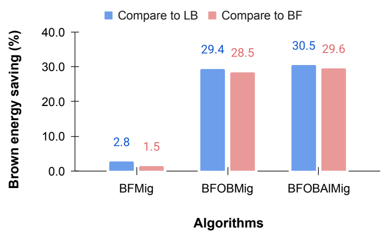


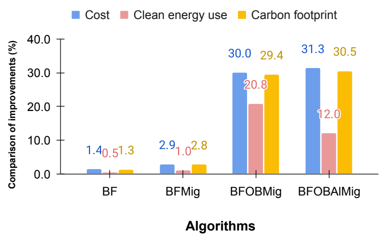
Fig. 8: Power consumption in different algorithms (*Nectar* workload) and supply from renewables (RE).



(a) The total amount of energy consumption (kWh) of different algorithms with proportion of brown and clean power usage.

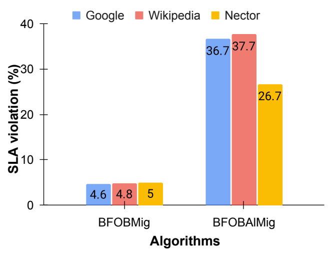


(b) Brown energy savings of diffrent algorithms compared to baseline algorithms (LB and BF).

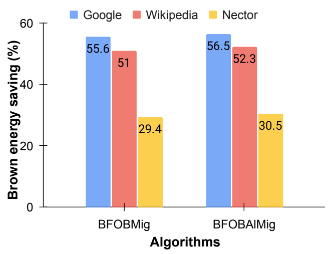


(c) Improvements in carbon footprint, clean energy use, and costs over the naïve LB algorithm.

Fig. 9: Results of algorithms for *Nectar* workload.

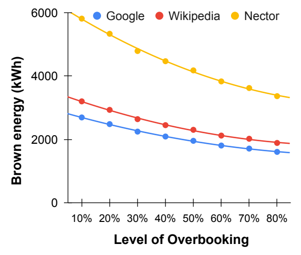


(a) Violations in *Green SLA* in *Google*, *Wikipedia*, and *Nectar* workloads

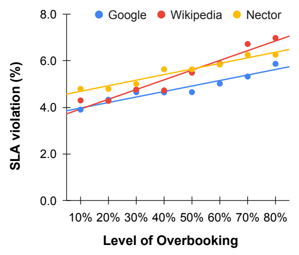


(b) Saving power from brown sources in *Google*, *Wikipedia*, and *Nectar* workloads

Fig. 10: Comparison of violating *Green SLA* and its impact on saving power from brown sources when serving *Google*, *Wikipedia*, and *Nectar* workloads with *BFOBMig* and *BFOBAlMig* algorithms



(a) Consumption from brown-source in Google, Wikipedia, and Nectar workloads with varying level of overbooking.



(b) Violations in *Green SLA* in *Google*, *Wikipedia*, and *Nectar* workloads with varying levels of overbooking.

Fig. 11: Comparison of power consumption from brown energy sources and violating *Green SLA* when serving *Google*, *Wikipedia*, and *Nectar* workloads with varying level of overbooking applied through *BFOBMig* algorithm.

and dynamic overbooking, whereas *BFMig* only implements VM migration, and *BFOBAlMig* implements overbooking always. Power consumption is always correlated with the workload, and our reactive resource management algorithm tries to reduce the consumption when not enough renewable energy is available, no matter the workload's pattern. We deliberately selected various workloads with different patterns to show how they affect the performance of our proposed algorithm.

Fig. 5b, Fig. 7b, and Fig. 9b show a comparative measure of energy consumption in all three workloads. The green-energy-aware $BFMig$, $BFOBMig$, and $BFOBAlMig$ algorithms consumed about 39.5%, 55.6%, 56.5% less brown energy compared to the naïve LB algorithm, and 13.1%, 36.2%, and 37.6% less brown energy compare to the baseline BF, respectively, to serve the *Google Cluster* workload (see Fig. 5b). In the *Wikipedia* workload, the green-energy-aware $BFMig$, $BFOBMig$, and $BFOBAlMig$ algorithms consumed 32.4%, 51.0%, 52.3% and 22.3% 43.6%, 45.1% less energy from brown sources than the LB, and BF, respectively (see Fig. 7b). Similarly, in the *Nectar* workload, the green-energy-aware $BFMig$, $BFOBMig$, and $BFOBAlMig$ algorithms consumed 2.8%, 29.4%, 30.5% and 1.5% 28.5%, 29.6% less energy from brown sources than the LB, and BF, respectively (see Fig. 9b). So our proposed *BFOBMig* algorithm can save a similar amount of brown energy for all three workload types without implementing the overbooking continuously (*BFOBAlMig*).

The naïve LB algorithm draws the highest amount of brown energy across all time slots compared to all other algorithms. At the same time, the green-energy-aware and overbooking capable *BFOBMig* and *BFOBAlMig* algorithms, consume less brown energy. The other green-energy-aware *BFMig* algorithm consumes less brown energy than the baselines but consumes more than *BFOBMig* and *BFOBAlMig* because it only applies VM migration, whereas *BFOBMig* and *BFOBAlMig* apply both VM migration and overbooking jointly and can consume less brown energy. The gain is significantly lower when only VM migration (without overbooking) based, green-energy-aware $BFMig$ algorithm is applied to the nearly flat Nectar workload. Our proposed green-energy-aware *EPU* based *BFOBMig* algorithm consumed about 36.2% (Google workload), 43.6% (Wikipedia workload), and 28.5% (Nectar workload) less brown energy than the baseline *BF* algorithm. These improvements are 23.1% (Google workload), 21.3% (Wikipedia workload), and 27.0% (Nectar workload) better (see the differences in brown energy savings of *BFMig* and *BFOBMig* when both measured compare to *BF* in Fig.5b, Fig.7b, and Fig.9b) than in *BFMig*, the renewable-energy-aware state-of-the-art algorithm that applied VM migration. Improvement reflected for all ten days for all the workloads. *BFOBMig* saved nearly similar amount of brown energy like *BFOBAlMig* (compare to naïve *LB*, just 0.9%, 1.3%, and 1.1 difference for *Google*, *Wikipedia*, and *Nectar* workloads, respectively), but it violates Green SLA around 32.1%, 32.9%, and 21.7% less than *BFOBAlMig* for *Google*, *Wikipedia*, and *Nectar* workloads, respectively (see the difference in Fig.10a and Fig.10b). Hence, without continuously overbooking the system, our proposed *EPU* based *BFOBMig* algorithm provided a similar gain as it adapted power saving strategies dynamically with the availability of renewable energy.

5.4.2 Impact on SLA Violations

As we exploit overbooking in favor of clean energy, the SLA violation is quantified as the ratio of total instance-time that system remains overbooked while there is enough renewable energy available to the total instance-time. Instance-time measures the duration of time that an instance remains in a particular state. For Green SLA violations, *BFOBMig* is measured to have around 4.6%, 4.8%, and 5.0% violation in comparison to around 36.7%, 37.7%, and 26.7% of always overbooked system *BFOBAlMig* for *Google*, *Wikipedia*, and *Nectar* workloads, respectively (Fig.10a). This results reflect that *BFOBMig* algorithm saves relatively similar brown energy to the always overbooked algorithm *BFOBAlMig*, while keeps the Green SLA violation significantly lower.

It is worth mentioning that overbooking may result in QoS degradation (e.g., delayed response time) for end users. In general, QoS degradation increases by the increase of overbooking levels, but depending on the application type or utilization levels, the impact might be different. Experiments carried out in [50] to measure the effect of overbooking level on average response time, in which one of the services has shown nearly 2.5 times (from $\approx$31ms. to $\approx$79ms.) increment from no overbooking state to 75% overbooking state. The other service has shown nearly two times (from $\approx$22ms. to $\approx$47ms.) increment from no overbooking state to 75% overbooking state. In general, CDC operators are unaware of type and current state applications hosted inside VMs; thus they cannot measure the impact of overbooking on QoS for end-users. As explained earlier, this is up to cloud clients (service providers) to manage QoS for their end users, determine the accepting level of overbooking, and if the Green SLA is right to them at all. Therefore, in this work, we do not report the QoS degradation in essence.

5.4.3 Carbon Footprint, Cost and Green Energy Usage

The overall performance of the proposed *BFOBMig* algorithm that implements the *EPU* concept through dynamic overbooking and VM migration-based resource management, compared to all other algorithms, can be determined by carbon emissions, cost of energy from the grid, and the fraction of clean energy usage.

We derive the carbon footprint as 1.2134 tCO_2e/MWh (tonnes of carbon dioxide equivalent per megawatt-hour) for the off-site grid electrical power, which is the weighted average of the carbon emission from the fossil fuel power stations [51] [52] in Victoria, Australia. We derive the cost of the electricity as per the rate in the area we gathered the energy source data. The electricity price of 26.95 c/kWh, 16.50 c/kWh for the peak and off-peak (11pm-7am) hours have been chosen to calculate a grid electricity price similar to the pricing for small industries in the Melbourne area [53].

We compared all the algorithms and measured their relative performance. To depict the relative improvement, we measured the performance of all other algorithms with respect to the naïve LB algorithm for carbon emission, cost of energy required to buy from the grid, and the proportion of clean energy usage. The results in Fig. 5c show that *BFOBMig* and *BFOBAlMig* reduce the carbon footprint to serve the Google workload by about 55.6% and 56.5%, than the naïve *LB*, which is around 25.3%, 26.2% and 16.1%, 17% better than the baselines *BF* and *BFMig* algorithms. This improvement for *BFOBMig* and *BFOBAlMig* are around 38.0%, 39.3% and 18.6%, 19.9% than the baseline *BF*, and *BFMig* algorithm for the *Wikipedia* workload (see the difference in Fig.7c). For the *Nectar* workload, this improvement for *BFOBMig* and *BFOBAlMig* are around 28.1%, 29.2% and 26.6%, 27.7% than the baseline *BF*, and *BFMig* algorithm, respectively(see the difference in Fig.9c). So, without overbooking the system continuously, our proposed *EPU* based *BFOBMig* algorithm provided a similar gain.

The cost of buying power from the grid to serve the *Google* workload is around 25.2%, 26.3% and 16.1%, 17.2% lower in *BFOBMig* and *BFOBAlMig*, respectively, than the baseline *BF*, and renewable-energy-aware *BFMig* algorithms (see the difference in *Cost* in Fig.5c). This improvement is around 37.4%, 39.0% and 19.2%, 20.8% for the *Wikipedia* workload and 28.6%, 29.9% and 27.1%, 28.4% for the Nectar, respectively (see the difference in *Cost* in Fig.9c and Fig.9c). These improvements reflect *BFOBMig* saved nearly similar cost to the always overbooked *BFOBAlMig* algorithm. Hence, *BFOBMig* provided a similar gain in cost savings without overbooking the system continuously, as it adapted power-saving strategies dynamically based on *EPU*.

5.4.4 Impact of Overbooking Level

The trend lines in Fig.11a show how power consumption from brown sources changes with varying overbooking levels with the *BFOBMig* algorithm. Results show that the more the overbooking level, the less the brown power consumption. Trend lines for all three workloads show that the performance gain becomes less significant with higher overbooking levels. This is because the rate of the number of hosts we can put in the idle state is reduced with the increase in overbooking level, while the workload remains the same. Therefore, increasing the overbooking level has a less significant impact on the brown energy consumption.

Fig.11b shows the *Green SLA* violation with varying levels of overbookings for all three types of workloads for *BFOBMig*. SLA violations occur due to the transition from the overbooked to non-overbooked states when renewable energy becomes sufficiently available. The amount of violation increases with the higher overbooking levels as it counts for more instance-time remained overbooked during the transition. Compared to the other two workloads, the violation rate increases a bit faster in *Wikipedia* workload since this workload follows a similar pattern as the renewable with a higher amount of workload at the transition.

5.5 Limitations

Our proposed solution is a reactive algorithm that works based on the source of energy information. A general limitation of the reactive approach is that decisions are made based on the system's current status. Forecasting variability is not explored here and prediction-based approaches are left as the future work.

Even though large-scale data centers may consist of heterogeneous devices with different batches of servers and switches, devices are mostly homogeneous within a single rack, cluster, and in small-scale data centers. Therefore, for the sake of simplicity, we focused on the homogeneous configuration of the problem and developed our strategies accordingly. However, without losing the general construction of the problem, the heterogeneous scenario can be modeled as a finite set of homogeneous subsets. Investigations on this variability are not explored here and are left as future work.

We carried out experiments with ten days of renewable energy data with real-world workloads. We varied the strategies keeping the environment same for experiments, achieved promising results in our strategy over the others for all days with all the workloads, strengthening validity [54]. The performance gain may vary if other days or price values are selected. However, the overall trends remains consistent over other various data. We considered a particular but realistic computing environment including devices, power consumption, and costs for our experimental setup as described, and hence, the results are specific to this setting. All setting values are adopted from relevant domain experimental setup of highly cited publications and common practice as described. Our investigation intended to compare the behaviors of load components that directly vary with computational load, allowing us to neglect load-invariant components, such as lighting and power control systems. As per Pelley et al. [19], these load-invariant power consumption components can contribute less than 9% of total power in a typical data center. Our results were produced to determine the relative performance of the respective schemes under comparison, not to produce an exact model of a specific data center, noting that energy results and load-invariant components will vary with the design of a data center.

6 Conclusions and Future Directions

This paper presents the concept of *EPU* in sustainable CDCs, a resource management framework for resource allocation for systems using on-site renewable energy that accounts for the type of energy source, renewable or non-renewable. We proposed an algorithm based on the *EPU* concept to allocate resources in a way that dynamically adopts an overbooking policy to shape electrical energy demands for serving workload to match the available clean energy supply. Our proposed method can reduce brown energy usage by exploiting VM consolidation and overbooking conditions when the supply of renewable energy is inadequate and lifting the overbooking when enough renewable energy supply is available. We demonstrated that our proposed *EPU* approach could effectively improve green energy utilization, and reduce brown energy usage.

In the future, we will investigate forecasting variability of workload and renewable energy and related prediction based approaches. We will extend our methodology by using Markov Decision Processes and Reinforcement Learning to dynamically set optimal overbooking ratios, and to explore prediction-based probabilistic decision approaches.

Acknowledgment

We thank *Nectar Cloud* to share their workload, MGS-MITS scholarship authority of Monash University, Australia.

References

[1] A. S. Andrae and T. Edler, “On global electricity usage of communication technology: trends to 2030,” *Challenges*, vol. 6, pp. 117–157, 2015.

[2] C. Trueman, “Why data centres are the new frontier in the fight against climate change,” *Computerworld*, vol. 9, 2019.

[3] D. J. Schumacher and W. C. Beckman, “Data center cooling system,” Apr. 23 2002, US Patent 6,374,627.

[4] E. Pakbaznia and M. Pedram, “Minimizing data center cooling and server power costs,” in *Proceedings of the 2009 ACM/IEEE International Symposium on Low Power Electronics and Design*, 2009, pp. 145–150.

[5] H. Goudarzi, M. Ghasemazar, and M. Pedram, “Sla-based optimization of power and migration cost in cloud computing,” in *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 2012, pp. 172–179.

[6] Q. Zhang, M. F. Zhani, S. Zhang, Q. Zhu, R. Boutaba, and J. L. Hellerstein, “Dynamic energy-aware capacity provisioning for cloud computing environments,” in *Proceedings of the 9th International Conference on Autonomic Computing*, 2012, pp. 145–154.

[7] T. M. Lynar, Simon, R. D. Herbert, and W. J. Chivers, “Reducing energy consumption in distributed computing through economic resource allocation,” *International Journal of Grid and Utility Computing*, vol. 4, no. 4, pp. 231–241, 2013.

[8] K. Zheng, X. Wang, L. Li, and X. Wang, “Joint power optimization of data center network and servers with correlation analysis,” in *INFOCOM 2014-IEEE Conference on Computer Communications*, pp. 2598–2606.

[9] J. J. Son, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, “Sla-aware and energy-efficient dynamic overbooking in sdn-based cloud data centers,” *IEEE Transactions on Sustainable Computing*, vol. 2, no. 2, pp. 76–89, 2017.

[10] A. Beloglazov, “Energy-efficient management of virtual machines in data centers for cloud computing,” Ph.D. dissertation, 2013.

[11] M. H. Ferdaus, M. Murshed, R. N. Calheiros, and R. Buyya, “Network-aware virtual machine placement and migration in cloud data centers,” in *Emerging research in cloud distributed computing systems*. IGI Global, 2015, pp. 42–91.

[12] A. N. Toosi, C. Qu, M. D. de Assunção, and R. Buyya, “Renewable-aware geographical load balancing of web applications for sustainable data centers,” *Journal of Network and Computer Applications*, vol. 83, pp. 155–168, 2017.

[13] A. Khosravi, A. N. Toosi, and R. Buyya, “Online virtual machine migration for renewable energy usage maximization in geographically distributed cloud data centers,” *Concurrency and Computation: Practice and Experience*, vol. 29, no. 18, p. e4125, 2017.

[14] A. Karimiafshar et al., “A request dispatching method for efficient use of renewable energy in fog computing environments,” *Future Generation Computer Systems*, vol. 114, pp. 631–646, 2021.

[15] A. A. Chien, “Good, better, best: how sustainable should computing be?” *Communications of the ACM*, vol. 64, no. 12, pp. 6–7, 2021.

[16] Í. Goiri, K. Le, M. E. Haque, R. Beauchea, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, “Greenslot: scheduling energy consumption in green datacenters,” in *Int. Conf. for High Performance Comp., Networking, Storage and Analysis*, 2011, pp. 1–11.

[17] VMware, “Performance best practices for vmware vsphere 6.7,” *VMware Inc.*, pp. 1–88, 2018.

[18] K. Leoni, “Maximizing vmware performance and cpu utilization,” 06 2020, https://www.heroix.com/blog/vmware-vcpu-over-allocation/.

[19] S. Pelley, D. Meisner, T. F. Wenisch, and J. W. VanGilder, “Understanding and abstracting total data center power,” in *Workshop on Energy-Efficient Design*, vol. 11, 2009, pp. 1–6.

[20] AWS, “Amazon ec2 spot instances,” 2022, https://aws.amazon.com/ec2/spot/, Accessed: April’22.

[21] R. Bianchini, “Leveraging renewable energy in data centers: Present and future,” in *Proceedings of International Symp. on High-Performance Parallel and Distributed Comp.* ACM, 2012, p. 135–136.

[22] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, “Geographical load balancing with renewables,” *SIGMETRICS Performance Evaluation Review*, vol. 39, no. 3, pp. 62–66, 2011.

[23] R. Wang, Y. Lu, K. Zhu, J. Hao, P. Wang, and Y. Cao, “An optimal task placement strategy in geo-distributed data centers involving renewable energy,” *IEEE Access*, vol. 6, pp. 61 948–61 958, 2018.

[24] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, “A taxonomy and survey of energy-efficient data centers and cloud computing systems,” in *Advances in computers*. Elsevier, 2011, vol. 82, pp. 47–111.

[25] J. Gao, “Machine learning applications for data center optimization,” *Available at: https://ai. google/research/pubs/pub42542*, 2014.

[26] A. Celesti, A. Puliafito, F. Tusa, and M. Villari, “Energy sustainability in cooperating clouds.” in *CLOSER*, 2013, pp. 83–89.

[27] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi, “Capping the brown energy consumption of internet services at low cost,” in *Int. Conf. on Green Computing*. IEEE, 2010, pp. 3–14.

[28] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, N. T. Hieu, and H. Tenhunen, "Energy-aware vm consolidation in cloud data centers using utilization prediction model," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 524–536, 2016.
[29] W. Li, T. Yang, F. C. Delicato, P. F. Pires, Z. Tari, S. U. Khan, and A. Y. Zomaya, "On enabling sustainable edge computing with renewable energy resources," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 94–101, 2018.
[30] M. S. Hasan, F. A. de Oliveira, T. Ledoux, and J.-L. Pazat, "Enabling green energy awareness in interactive cloud application," in *Int. Conf. on Cloud Computing Technology and Science*. IEEE, 2016, pp. 414–422.
[31] 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, 2012.
[32] B. Heller et al., "Elastictree: Saving energy in data center networks." in *Nsdi*, vol. 10, 2010, pp. 249–264.
[33] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *International green computing conference*. IEEE, 2012, pp. 1–10.
[34] A. N. Toosi and R. Buyya, "A fuzzy logic-based controller for cost and energy efficient load balancing in geo-distributed data centers," in *8th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, 2015, pp. 186–194.
[35] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A comparison of high-level full-system power models." *HotPower*, vol. 8, no. 2, pp. 32–39, 2008.
[36] A. Jayanetti and R. Buyya, "J-opt: A joint host and network optimization algorithm for energy-efficient workflow scheduling in cloud data centers," in *12th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, 2019, pp. 199–208.
[37] T. Chakraborty, A. N. Toosi, C. Kopp, P. Stuckey, and J. Mahet, "Joint host-network power scaling with minimizing vm migration in sdn-enabled cloud data centers," in *13th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, 2020, pp. 1–12.
[38] X. Wang, Y. Yao, X. Wang, K. Lu, and Q. Cao, "Carpo: Correlation-aware power optimization in data center networks," in *Proceedings of IEEE INFOCOM*. IEEE, 2012, pp. 1125–1133.
[39] A. Shehabi, S. Smith, N. Horner, I. Azevedo, R. Brown, J. Koomey, E. Masanet, D. Sartor, M. Herrlin, and W. Lintner, "United states data center energy usage report," *Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775 Page*, vol. 4, 2016.
[40] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, "Making scheduling "cool": Temperature-aware workload placement in data centers." in *USENIX annual technical conference*, 2005, pp. 61–75.
[41] S. Ilager, K. Ramamohanarao, and R. Buyya, "Etas: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 17, p. e5221, 2019.
[42] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format+ schema," *Google Inc., White Paper*, pp. 1–14, 2011.
[43] WikiBench, "Wikipedia access traces," 2021, http://www.wikibench.eu/wiki/2007-09/, Accessed: June'21.
[44] G. Urdaneta, G. Pierre, and M. Van Steen, "Wikipedia workload analysis for decentralized hosting," *Computer Networks*, vol. 53, no. 11, pp. 1830–1845, 2009.
[45] Nectar, "Nectar cloud," 2021, https://ardc.edu.au/services/nectar-research-cloud/, Accessed: June'21.
[46] "Weather station data, Melbourne," www.wunderground.com/dashboard/pws/IMELBOUR543/table/, Accessed:June'21.
[47] H. Gitano-Briggs, "Low speed wind turbine design," *in Wind Power, Rupp Carriveau, IntechOpen*, pp. 267–283, 2012.
[48] M. Fripp and R. H. Wiser, "Effects of temporal wind patterns on the value of wind-generated electricity in california and the northwest," *IEEE Transactions on Power Systems*, vol. 23, no. 2, pp. 477–485, 2008.
[49] Solar Radiation www.pveducation.org/pvcdrom/properties-of-\sunlight/\solar-radiation-on-a-tilted-surface, Access:June'21.
[50] L. Tomàs and J. Tordsson, "An autonomic approach to risk-aware data center overbooking," *IEEE Transactions on Cloud Computing*, vol. 2, no. 3, pp. 292–305, 2014.
[51] Wiki, "Power stations in victoria," en.wikipedia.org/wiki/List_\of_power_stations_in_Victoria_(Australia), Accessed: June'21.
[52] Electricity sector emissions and generation data www.cleanenergyregulator.gov.au/NGER/, Access:June'21.
[53] "Electricity price," www.globirdenergy.com.au/shared-assets/Victorian_Energy_Fact_Sheets/Victorian_Energy_Fact_Sheet_GLO295566MS_Electricity.pdf, Access:June'21.
[54] D. T. Campbell and T. D. Cook, "Quasi-experimentation," *Chicago, IL: Rand Mc-Nally*, 1979.

Tuhin Chakraborty Tuhin Chakraborty is a PhD student in the Department of Software Systems and Cybersecurity, Faculty of Information Technology, Monash University, Australia. The main area of his doctoral research is Sustainable Cloud Computing. His research interests include Green computing, Accessible computing, Human-computer interaction. He has published in top-tier venues such as ACM SIGCHI, ACM Transaction on Accessible Computing. He served as a reviewer in top-tier venues like ACM SIGCHI, ACM Mobile HCI, ACM DIS.

Adel N. Toosi (Member, IEEE) received the PhD degree from the School of Computing and Information Systems, University of Melbourne, in 2015. He is currently a lecturer (assistant professor) with the Department of Software Systems and Cybersecurity, Faculty of Information Technology, Monash University, Australia. From 2015 to 2018, he was a postdoctoral research fellow with the University of Melbourne. Dr Toosi has published over 50 peer-reviewed publications in top-tier venues such as IEEE Transactions on Cloud Computing, IEEE Transactions on Services Computing, and IEEE Transactions on Sustainable Computing. His publications have received over 3,500 citations with a current h-index of 28 (Google Scholar). His research interests include cloud, fog, edge computing, software-defined networking, green computing, and energy efficiency. Currently, he is working on building sustainable Edge/Fog computing environments. For further information, please visit his homepage: http://adelnadjarantoosi.info.

Carlo Kopp (M'98—SM'10) received a BE with First Class Honours in Electrical Engineering from the University of Western Australia, in 1984, and MSc and PhD degrees in Computer Science from Monash University, in 1996 and 2000, respectively. He is with the Department of Software Systems and Cybersecurity at Monash University in Melbourne, Australia. His doctoral work was on the adaptation of active phased arrays for Gigabit/s data rate airborne networking applications, and related low elevation angle tropospheric microwave propagation problems. He has previously conducted research in operating systems, ad hoc networks and radio-frequency propagation, radar signature computational modelling, optical communications, and satellite navigation support protocols. Dr Kopp has a total of around 670 publications in multiple categories across the full breadth of his research interests, including 67 peer reviewed research papers, and he co-authored Chapter 5 in the third edition of Skolnik's *Radar Handbook*. Dr Kopp has extensive industry experience as a designer, developer and integrator of hard real-time embedded software, operating systems internals software, especially Unix device drivers, communications protocols, high speed digital hardware including Emitter Coupled Logic, SPARC motherboards, graphics adaptors, and analogue electronic hardware including wideband optical fibre receivers and transmitters. Dr Kopp is an Associate Fellow of the AIAA and a Senior Member of the IEEE.