Science and Technology for Energy Transition **79**, 82 (2024)
© The Author(s), published by EDP Sciences, 2024
https://doi.org/10.2516/stet/2024076

Available online at:
stet-review.org

Emerging Advances in Hybrid Renewable Energy Systems and Integration
Muhammad Zakarya, Santosh Tirunagari, Jinguang Han and Peiying Zhang (Guest Editors)

REGULAR ARTICLE OPEN ACCESS

Energy and carbon-aware distributed machine learning tasks scheduling scheme for the multi-renewable energy-based edge-cloud continuum

Zicong Miao[1], Lei Liu[2,*], Haijing Nan[1], Weize Li[1], Xiaodong Pan[1], Xin Yang[1], Mi Yu[1,3], Hui Chen[1], and Yiming Zhao[4]

[1] Hybrid Cloud Intelligent Innovation Laboratory, China Telecom Cloud Computing Corporation, Beijing 100088, China
[2] School of Energy Science and Engineering, Central South University, Changsha 410083, China
[3] China Telecom Corporation Limited Chongqing Branch, Chongqing 401120, China
[4] School of Business, Renmin University of China, Beijing 100872, China

Received: 28 June 2024 / Accepted: 30 August 2024

Abstract. As an increasing number of Distributed Machine Learning (DML) tasks are hosted on cloud platforms in the edge-cloud continuum, Data Centers (DCs) with massive data and computational requirements have become one of the world's largest energy consumers, leading to significant carbon emissions. Reducing energy consumption and carbon emissions is an extremely crucial and challenging issue for the sustainable development of cloud service providers. While utilizing renewable energy can help reduce the carbon emissions of DCs, the intermittent and unstable nature still causes DCs to rely heavily on high-carbon brown energy. For the resource-intensive and delay-tolerant DML tasks, this paper introduces multi-renewable energy in the geo-distributed continuum to address this issue, the spatiotemporal complementarity maximizes the renewable energy utilization and compensates for time-dependent energy differences with geographic advantages. Additionally, considering the dynamic differences in carbon intensity and electricity prices across distributed DCs in the continuum, we propose an energy and carbon-aware algorithm called ECMR for scheduling heterogeneous virtual machine creation tasks of DML among multi-clouds in different time zones. It is demonstrated that compared with the baseline methods, the ECMR significantly reduces the total power consumption, energy cost, and carbon emission of data centers while maintaining an acceptable service quality. The utilization of renewable energy in data centers has been significantly improved to 90.8% by flexibly leveraging the spatiotemporal complementarity of multi-renewable energy. Compared with existing competing algorithms, the proposed method exhibits significant improvements with an achieved average response time of 12.6 ms, and a task failure rate of 1.25%.

Keywords: Edge-Cloud Continuum, DML task Schedulilng, Distributed Data Centers, Multi-Renewable Energy, Energy Consumption, Carbon Emission, Energy Complementarity.

Abbreviations

RES	Renewable Energy Source
DC	Data Center
DML	Distributed Machine Learning
CSP	Cloud Service Provider
CWS	Cloud Workload Scheduler
VM	Virtual Machine
QoS	Quality of Service
PUE	Power Use Efficiency
PV	Solar Photovoltaic

* Corresponding author: csu_liu@csu.edu.cn

Symbols

d_j, N	The jth DC and the total number of the DCs
s_i, M	The ith server and the total number of servers in d_j
v_k, L	The kth VM and the total number of VMs running on s_i
$P_{s_i}(t)$	The power of server s_i at time slot t
$P_{d_j}(t)$	The power of all servers in d_j at time slot t
$P_{d_j}^{\mathrm{DC}}(t)$	The power consumption of d_j at time slot t
$P_{s_i}^{\mathrm{ilde}}$	The idle power of the server s_i

$P_{s_i}^{\text{peak}}$	The maximum power of server s_i
$P_{s_i}^{\text{dynamic}}$	The maximum dynamic power of server s_i
$u_{s_i}(t)$	The utilization of s_i at time slot t
U_{ij}^{CPU}	The CPU capacity of s_i in d_j
$\text{AC}_i(t)$	The available CPU resources of s_i at time slot t
$u_{v_k}(t)$	The utilization of kth VM running on s_i
$v_{\text{in}},\ v_{\text{out}},\ v_r$	Start-up, survival, and rated wind speed
P_r	Rate power of the wind turbine
$v_j(t)$	Wind speed of the turbine in d_j at time slot t
$P_q^{\text{wind}}(t)$	Output wind power of the qth turbine at time slot t
ω	Total number of wind turbines in d_j
$Pw_j(t)$	Total output wind power of turbines in d_j
$G,\ F$	Irradiance and temperature
n_s	The number of the solar cells in the photovoltaic panel
$I_{\text{sc}}\ (G,\ F)$	Short-circuit current
$I_{\text{o}}(F)$	The dark saturation current
$r_{mp},\ i_{mp}$	The optimal load and the corresponding current
$v(G,\ F)$	The product of the output voltage
$i(G,\ F)$	The product of the output current
v_{oc}	Open-circuit voltage
V_{th}	The junction thermal voltage
R_s	The series resistance of the photovoltaic panel
$P_g^{\text{solar}}(t)$	Output solar power of the gth PV panel at time slot t
λ	The total number of PV panels in d_j
$Ps_j(t)$	The total output solar power of PV panels in d_j
$\text{Cer}_j(t)$	Carbon intensity of d_j at time slot t
$\text{CE}_j\ (\text{t})$	Carbon emission of d_j at time slot t
C_{carb}	Total carbon emission of all DCs
$E_j^{\text{brown}}(t)$	Brown energy consumption of d_j at time slot t
$E_j^G(t)$	Green energy generation of d_j at time slot t
$E_j(t)$	Energy consumption of d_j at time slot t
$E_j^{\text{wind}}(t)$	Supplied wind energy in d_j at time slot t
$E_j^{\text{solar}}(t)$	Supplied solar energy in d_j at time slot t
T	Total time slots
$\xi_j(t)$	The locational marginal price of d_j at time slot t
$Q_j(t)$	Energy cost of d_j at time slot t
C_{eng}	Total energy cost of all DCs
ψ_r	The rth VM request
R_r	the required computing resources of ψ_r
$\delta_j(t)$	The list of running VMs in d_j at time slot t
$R_{rj}^{\text{CPU}}(t)$	Requested CPU of ψ_r that dispatched to d_j at time t
$R_{rj}^{\text{MEM}}(t)$	Requested memory of ψ_r that dispatched to d_j at time t
$R_{rj}^{\text{MEM}}(t)$	Requested storage of ψ_r that dispatched to d_j at time t
$y_{rij}(t)$	The Boolean variable
$T_{r,\,d_j}$	Request time of ψ_r dispatched to dj
Tw_{rj}	Waiting time of ψ_r dispatched to dj
Tr_{rj}	Response time of ψ_r dispatched to dj
Te_{rj}	Execution time of ψ_r dispatched to dj
Ta_{rj}	Total processing time of ψ_r dispatched to dj
L_{rj}	Latency of ψ_r
L_{delay}	Total latency of all requests

1 Introduction

The proliferation of intelligent vehicles, mobile phones, and various Internet of Things (IoT) devices across society, coupled with the growing number of edge devices and applications, is driving a substantial increase in data volume [1]. The availability of massive amounts of data has played a crucial role in the advancement of Distributed Machine Learning (DML) solutions [2]. However, IoT devices located at the edge of the network face limitations in computing power, energy, storage capacity, and bandwidth, making it impossible to run resource-intensive machine learning tasks. The resource-constraint nature of edge devices needs to be integrated with cloud computing platforms, and tasks should be scheduled based on the resource requirements of DML tasks and the corresponding parameters of edge and cloud resources, which promotes the emergence of the edge-cloud continuum.

With the increase of DML tasks allocated to cloud Data Centers (DCs), the energy consumption of the edge-cloud continuum is becoming a critical concern economically, environmentally, and societally [3]. Cloud Service Providers (CSPs) typically deploy geo-distributed data centers with tens of thousands of servers to process requests from internet users worldwide [4]. Despite globally distributed cloud DCs providing users in different time zones with flexible access to computing resources and reducing user costs, if we consider the worldwide DCs as one country, it would be the fifth-largest electricity consumer in the world today [5, 6]. High energy consumption directly contributes to high carbon emissions, as of 2021, the carbon emissions contributed by DCs accounted for 1% of global greenhouse gas emissions [7], it is projected that by 2025, DCs will emit 1.9 billion tons of carbon dioxide [8]. Consequently, how to reduce energy consumption and carbon emissions is an extremely important and challenging issue for the sustainable development of CSPs.

An effective approach to reducing the high energy consumption and carbon emissions from DCs is to improve the operation efficiency of DCs. Typically, the average resource utilization of Physical Machines (PMs) ranges from 10% to 50%, and most DCs' PMs are underutilized, yet they consume nearly 70% of the peak energy [9]. Therefore, actively managing and consolidating idle PMs to reduce the number of active servers could lead to a reduction in total energy consumption while maintaining the same number of workloads. Resource and data-intensive and delay-tolerant

DML tasks can be viewed as requests to Virtual Machines (VMs), currently, most researchers employ heuristic and metaheuristic algorithms in their VM scheduling systems [10, 11], which can reallocate VMs located in high energy consumption and high carbon emissions in DCs within a limited decision time, but these algorithms are less generalized, and the scheduling effect becomes worse when facing heterogeneous tasks on different edge computing nodes. Furthermore, the delay-tolerant tasks have the operation characteristics of temporal delay and spatial transfer, so that the CSPs can choose the execution time of tasks and transfer the power load through flexible VM scheduling within or even across data centers, thus reducing energy costs and carbon emissions. Although VM scheduling can effectively reduce energy consumption, the reduction of carbon emissions is constrained by the energy supply sources. Specifically, Renewable Energy Source (RES), known as green energy, has significantly lower carbon intensity compared to fossil fuels such as coal, natural gas, and oil, which are considered brown energy sources, yet 80% of the electricity used by DCs worldwide still comes from carbon-intensive energy sources [8].

To mitigate the rapid increase in carbon emissions caused by energy consumption, CSPs are actively operating their cloud DC using RES [12]. For instance, approximately 50% of the DCs power for *Amazon* and *Microsoft* comes from RES [13], and *Apple's* DCs are even 100% powered by RES [14]. Therefore, powering DCs partially or entirely by RES has gained popularity in modern DC practices as it can significantly reduce reliance on brown energy and decrease carbon footprint. However, the fact is that RES exhibits intermittency and instability due to climate and geographical location variations [15]. Firstly, DCs distributed in geographical locations will exhibit different availability and carbon emission intensity due to climate differences. The annual solar irradiance in the western United States surpasses that in the Northeast, enabling solar farms with the same scale and energy conversion efficiency to produce more electricity in the western region. Secondly, within the same region, different types of renewable energy show different availability even under the same weather conditions. For example, solar energy can only be available during the day, but wind turbines driven by wind can even achieve 24-hour operation. Renewable energy has significant temporal variability and location dependence, moreover, RES exhibits a clear "peak-valley" pattern at different periods [15]. Therefore, it is a remaining challenge for CSPs to develop VM scheduling strategies based on load requests, carbon emission intensity, electricity prices, and the temporal-spatial variability of renewable energy to maximize its utilization.

This research aims to introduce multi-RES into geo-distributed DCs in edge-cloud continuums and flexibly leverage their complementary characteristics in time and space through scheduling algorithms to minimize DC energy consumption and carbon emissions. We formulate the task scheduling scheme as a constrained optimization problem, where the constraints consider the task heterogeneity, carbon intensity, electricity price, and availability of different types of RES and QoS. We propose a Mixed Integer Linear Programming (MILP) algorithm called ECMR, which solves the problem of peak time and regional spatial mismatch between computing resources and power loads as well as multiple types of renewable energy sources. Compared to existing jobs, the innovation of ECMR is reducing DC energy consumption and carbon emissions from the perspective of complementarity of multi-RES in geo-distributed environments. Robustness testing shows that the ECMR algorithm can still maintain good performance under pressure conditions such as resource constraints and reduced renewable energy supply. The main contributions and highlights of this paper are as follows: (i) We propose a new heuristic heterogeneous VM task scheduling algorithm from the perspective of temporal and spatial complementarity of multi-RES to regulate energy utilization issues between data centers and power grid systems. (ii) The wind and solar energy are modeled from both temporal and spatial dimensions based on the difference in geographical locations and weather of DCs, the intermittency and instability of renewable energy are effectively solved by leveraging the spatiotemporal complementarity of multi-RES in geo-distributed edge-cloud continuum. (iii) The proposed ECMR algorithm is validated extensively based on real-world VM request information, electricity prices, carbon intensity, and availability of renewable energy.

2 Related work

To reduce energy consumption in DCs and achieve carbon neutrality goals, extensive research has been conducted by the industry and academia. To improve the energy efficiency of data centers first requires enhancing their operational efficiency. Dynamic Voltage and Frequency Scaling (DVFS) manages server power consumption by adjusting the frequency and voltage scaling of components such as CPUs and memory [16]. For instance, [9] utilized the optimal power allocation of large server farms by analyzing the impact of DVFS on server frequency. A DVFS-based scheduling algorithm that can ensure system performance and improve resource utilization was proposed in [17]. Some researchers have also proposed energy-saving technologies based on IT device hibernation mechanisms [18] and dynamic speed scaling [19]. Although the above methods can reduce energy consumption, reducing processor frequency will cause a decrease in processor performance, making the DCs take longer to respond to external events or tasks.

Besides optimizing energy consumption at the IT hardware and DC power supply equipment level, scheduling and consolidation of VM is also an important method to reduce DC power consumption, which allows CSPs to optimize the spatial and temporal distribution of power loads based on power supply conditions. Large CSPs such as *Google* [20] and *Microsoft* [21] have proposed task scheduling algorithms to improve server utilization by consolidating fragmented available resources, thereby releasing more occupied servers into low-power mode. A multi-objective integer linear programming model was established to maximize the number of hosted VMs [22]. The researchers proposed an online VM consolidation mechanism that improves computing resource utilization and reduces power consumption under the constraints of CPU, memory, bandwidth, and Service Level Agreement (SLA) [23]. For the scheduling of heterogeneous tasks, the job scheduling problem was formulated as a Markov decision process under the premise of comprehensively considering job correlation, heterogeneity, and Quality of Service (QoS) [24]. A dynamic VM consolidation method for heterogeneous cloud DCs was proposed in [25], incorporating greedy heuristic algorithms and interactive operations to minimize migration and energy costs. Geo- distributed clouds have received considerable attention in recent years. A multi-objective optimization algorithm that considers initial VM placement and VM consolidation to minimize DC energy cost in a federated cloud environment was proposed in [26]. An integer linear programming for geo-distributed task scheduling problems while balancing workload heterogeneity was proposed in [27], which reduced carbon emission through spatiotemporal scheduling.

Although these studies can improve the energy efficiency of DCs, they overlook the utilization of renewable energy and limit the reduction of carbon emissions. To further reduce the operating costs of DCs, the application of RES in data centers is also attracting increasing attention. A method for managing carbon footprint and solar energy in distributed DCs located in different time zones was proposed in [16]. By evaluating the impact of renewable energy utilization and distribution on energy consumption in different DCs, the task scheduling strategies were formulated to reduce carbon emissions while ensuring QoS. Wind energy was introduced into the power supply system for DCs in [28], and a combined forecasting method of RES power generation based on integrated Empirical Mode Decomposition (EMD) and Temporal Convolutional Network (TCN) was used to improve the utilization efficiency of wind energy. Furthermore, researchers in [4] introduced distributed DCs with wind power and energy storage to adjust the time dimension availability of renewable energy. However, the above studies are all focused on a single type of RES, the hybrid energy (brown and green energy) supply is still needed to support the full availability of DCs, resulting in RES utilization being limited by the maximum output power of the adopted green energy.

The implementation of multi-renewable energy into the energy supply of DCs is gradually receiving attention. For instance, solar and wind energy were introduced to power DCs [29, 30], however, they did not distinguish the distributed RES by region but considered it as a whole. Similarly, the VM dispatch algorithm proposed in [31] only considered the migration of VMs between DCs in the same region, which means all DCs under the same environmental condition. The above methods ignore the spatial variations in RES, resulting in the inability to utilize RES for global DCs when weather conditions are unfavorable. Although the application of multi-RES articles has been discussed [32, 33], seldom focused on the complementary characteristics displayed by various types of energy sources (hydropower, wind energy, solar energy) throughout the year and day. Therefore, the remaining challenge for CSPs is to develop VM scheduling strategies based on load requests, carbon emission intensity, electricity prices, and the spatiotemporal variability of renewable energy to maximize its utilization.

3 Proposed system model

This article only considers DML tasks that require a lot of computing resources to be processed in the cloud, the model of an energy and carbon-aware VM scheduling system for geo-distributed DCs in the edge-cloud continuum is built. The proposed system relies on a hybrid energy source including green energy and brown energy to power the DC. DML requests from geographic regions are allocated to the appropriate DC through the Cloud Workload Scheduler (CWS) to reduce energy costs and carbon emissions. We analyze and establish models for DC's energy consumption and renewable energy generation, and the objective function is determined by comprehensively considering energy cost, carbon emission, and QoS. When analyzing the complementary characteristics of RES, this study mainly focuses on wind and solar energy, as detailed meteorological data for simulation are available in references [34, 35].

3.1 System architecture

The overall system architecture of the proposed method is presented in Figure 1. Similar to the architecture mentioned in [3], the edge-cloud continuum comprises three distinct regions and energy providers that are geographically distributed in different areas. In each region, both edge and cloud DC are included. The CSP operates distributed DCs across different time zones, and all DCs are interconnected via high-speed backbone networks. Other essential elements in the system include the CWS, and grid utility. Given that VM creation requests may originate from diverse regions, the requests they submit to the cloud for execution are also distributed. The CWS is used to connect all DCs and is the most critical component in the entire system architecture. On the one hand, the CWS collects VM creation requests, on the other hand, it gathers status information and characteristics from all DC feedback, including energy utilization, power consumption, renewable energy availability, energy price, carbon intensity, etc. The CWS then allocates received VM requests to appropriate DCs for execution based on the designed scheduling algorithm. The distributed DCs are powered by local grid utility, and we assume that the power grid in each DC is integrated with wind, solar, and brown energy. We do not consider cross-domain scheduling and long-term storage of renewable energy.

Based on the described architecture, we establish a basic mathematical model for a system with N distributed DCs, denoted as $d_j \in \{d_1, d_2, \ldots d_N\}$. Each DC consists of M servers of various models and configurations, denoted as $s_\mathrm{i} \in \{s_1, s_2, \ldots s_M\}$. We assume that the proposed scheme operates over discrete time $t \in \{1, 2 \ldots T\}$, with the maximum duration of each time slot being 1 h. The CWS dispatches the requests within each time slot, which are executed by VMs that are reported as $v_k \in \{v_1, v_2, \ldots v_\mathrm{L}\}$.

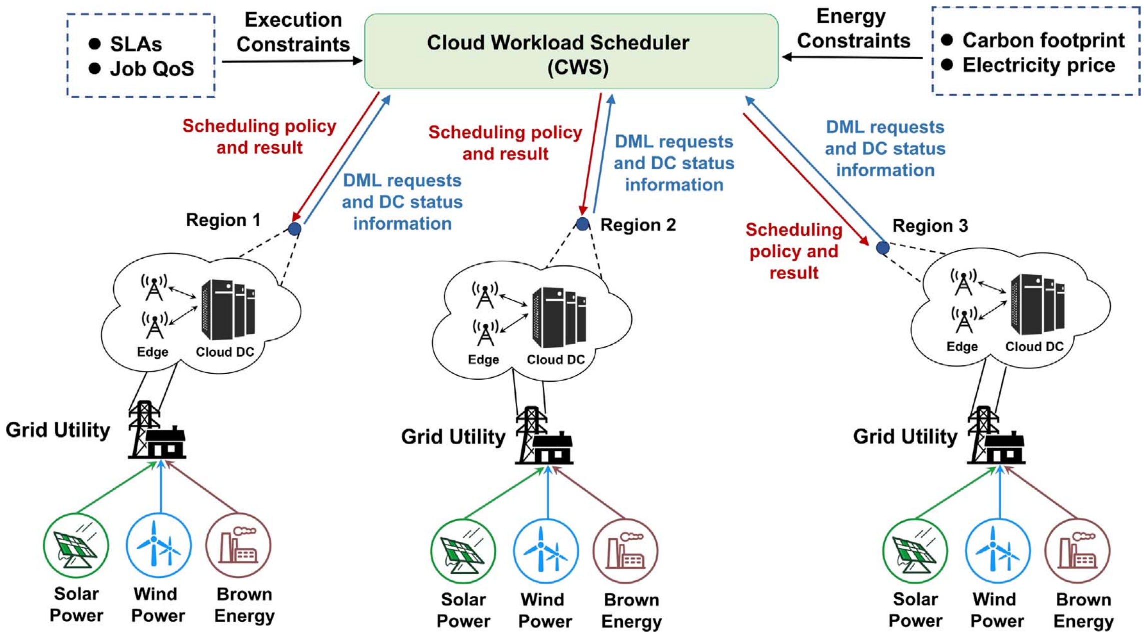

Fig. 1. The architecture of the proposed DML tasks scheduling scheme for geo-distributed edge-cloud continuums. The CWS optimizes task scheduling based on DML requests, SLA/QoS constraints, and RES availability for efficient energy usage and reduced carbon emissions.

3.2 Heterogeneous workload model

The DML workloads in DCs are generally divided into two categories: latency-sensitive workloads and latency-tolerant workloads. Latency-sensitive workloads (*e.g.* online forecasting system, financial trading system) require processing within a few milliseconds to a dozen milliseconds after they are issued, which greatly constrains the time cost of task scheduling and is not suitable for scheduling strategies that cause more time delays. In contrast, latency-tolerant workloads (*e.g.* climate model prediction, geological exploration data analysis) typically require longer runtime and a large amount of computing resources without immediate processing. This allows the CWS to optimize scheduling by fully utilizing the state information of the distributed system, such as real-time electricity prices and RES availability. Therefore, this paper focuses solely on latency-tolerant workloads in the simulation. It is worth noting that adopting strict strategies to reduce energy consumption and carbon emission while ignoring the resources required for DML tasks may lead to a decrease in QoS, resulting in SLA violations. Furthermore, due to the high heterogeneity of latency-tolerant workloads in production environments, they vary in terms of CPU, memory, and storage requirements, etc. [29, 36]. When the CWS receives a VM creation request and allocates it to a server in one of the available DCs based on the required resource, there are the following constraints for the VM task ψ_r to be assigned:

$$\sum_{i=1}^{M}\sum_{j=1}^{N} y_{rij}(t) = 1, \tag{1}$$

where $y_{rij}(t)$ is a Boolean variable used to determine whether the required resources of ψ_r are allocated to a server s_i in DC d_j. If $y_{rij}(t) = 1$, it means that ψ_r is allocated to s_i in d_j, and 0 otherwise not. The scheduling algorithm also needs to ensure that each VM is allocated to one and only one server. In addition, when scheduling VMs, the available resource constraints of servers need to be satisfied, meaning that the available resource capacity of the allocated server should be greater than the resource requirements of CPU, memory, and storage for VM requests. If the total number of VMs running in DC d_j at time slot t is $\delta_j(t)$, then we can get the following constraints:

$$\sum_{r=1}^{\delta_j(t)} R_{rj}^{\mathrm{CPU}}(t) \times y_{rij}(t) \le U_{ij}^{\mathrm{CPU}}, \tag{2}$$

$$\sum_{r=1}^{\delta_j(t)} R_{rj}^{\mathrm{MEM}}(t) \times y_{rij}(t) \le U_{ij}^{\mathrm{MEM}}, \tag{3}$$

$$\sum_{r=1}^{\delta_j(t)} R_{rj}^{\mathrm{STO}}(t) \times y_{rij}(t) \le U_{ij}^{\mathrm{STO}}, \tag{4}$$

where the $R_{rj}^{\mathrm{CPU}}(t)$, $R_{rj}^{\mathrm{MEM}}(t)$, $R_{rj}^{\mathrm{STO}}(t)$ is the number of the CPU cores, memory, and storage required by VM requests ψ_r allocated to ψ_r at time slot t, U_{ij}^{CPU}, U_{ij}^{MEM}, and U_{ij}^{STO} denote the total capacity of corresponding resources of server s_i in d_j, respectively.

3.3 Data center energy consumption model

The energy consumption model of the DC is an important basis for scheduling strategies. Power Use Efficiency (PUE) is one of the most effective parameters for measuring the energy efficiency of DCs. It is the ratio of the total energy consumption of the DC to the energy consumption of IT equipment, a higher ratio indicates a smaller proportion of overhead power consumed by cooling, lighting, and other systems. Since PUE is generally calculated using energy consumption over a certain period (such as a month or a year), this paper sets the experimental period to one week. Therefore, like most other research the PUE is set as a constant [31]. The power of a server shows a significant positive correlation with its CPU utilization, so we consider the CPU utilization as the server utilization. We incorporate the power model proposed in [37] to describe the linear relationship between server power and utilization, and the power model of the server s_i at time slot t can be expressed as:

$$P_{s_i}(t) = P_{s_i}^{\text{ilde}} + u_{si}(t) \times P_{s_i}^{\text{dynamic}}, \tag{5}$$

where the maximum dynamic server power $P_{s_i}^{\text{dynamic}}$ is the difference between the peak server power $P_{s_i}^{\text{peak}}$ and the idle server power $P_{s_i}^{\text{idle}}$, which is denoted as:

$$P_{s_i}^{\text{dynamic}} = P_{s_i}^{\text{peak}} - P_{s_i}^{\text{idle}}, \tag{6}$$

$P_{s_i}(t)$ is calculated by the idle power $P_{s_i}^{\text{idle}}$ and dynamic power $P_{s_i}^{\text{dynamic}}$ of the server, and $P_{s_i}^{\text{dynamic}}$ is linearly related to the utilization of server s_i at time slot t. Assuming that the maximum available CPU resource provided by s_i in the d_j is U_{ij}^{CPU}, if the available CPU resource margin of the server s_i at time slot t is AC_i (t), then the utilization of server s_i at time slot t can be calculated as:

$$u_{s_i}(t) = \frac{U_{ij}^{\text{CPU}} - \text{AC}_i(t)}{U_{ij}^{CPU}}. \tag{7}$$

Although the consumed energy of a server includes the consumption of virtualization programs and management components, this portion is all included in $P_{s_i}^{\text{idle}}$. Therefore, the utilization of the server s_i in d_j is nearly equivalent to the sum of the utilization of allocated VMs running on s_i at time slot t, thus:

$$u_{s_i}(t) = \sum_{k=1}^{L} u_{v_k}(t), \tag{8}$$

where $u_{v_k}(t)$ is the utilization of the kth VM running on the server s_i and the L is the total number of allocated VMs on it. Then, the total power of all servers $s_i \in \{s_1, s_{2,} \ldots s_\text{M}\}$ in DC d_j at time slot t is defined as:

$$P_{d_j}(t) = \sum_{i=1}^{M} P_{s_i}(t). \tag{9}$$

According to the definition of the PUE formula, we obtain the total power of the data center as:

$$P_{d_j}^{\text{DC}}(t) = \text{PUE} \times P_{d_j}(t). \tag{10}$$

Then the energy consumption of DC d_j at time slot t can be calculated as:

$$E_j(t) = \int_t^{t+\Delta t} P_{d_j}^{\text{DC}}(t) \tag{11}$$

3.4 Multi-renewable energy source generation model

The availability of RES fluctuates significantly across geographies and times, and the availability of different types of RES can vary significantly even under the same weather and environmental conditions. Our goal is to coordinate multi-RES and flexibly utilize their complementarity to schedule VM requests to reduce carbon emissions. Wind energy and solar energy, as the most prominent RES, currently provide 62% and 13% of non-hydropower clean energy globally, respectively [32]. We introduce wind energy, solar energy, and brown energy together to power geo-distributed DCs and prioritize assigning tasks to DCs with abundant and available renewable energy in our scheduling algorithm.

3.4.1 Wind power model

Existing research shows that the electricity output from wind turbines can be modeled by actual wind speed [30]. Assume that the wind speed at time slot t near the DC d_j is $v_j(t)$, then the output power $P_q^{\text{wind}}(t)$ of the qth wind turbine in the wind farm of d_j can be approximated as follows:

$$P_q^{\text{wind}}(t) = \begin{cases} 0 & v_j(t) < v_{\text{in}}, v_j(t) > v_{\text{out}} \\ P_r \times \dfrac{v_j(t) - v_{\text{in}}}{v_r - v_{\text{in}}} & v_{\text{in}} < v_j(t) < v_r \\ P_r & v_r < v_j(t) < v_{\text{out}} \end{cases} \tag{12}$$

where v_{in}, v_{out} are the start-up and survival wind speed, v_r, P_r are the rated wind speed and the rated power of the wind turbine. When the wind speed exceeds v_{in}, the wind turbine starts to rotate and output power. The output power increases linearly between v_{in} and v_r. When the wind speed exceeds the rated wind speed and is less than the survival wind speed v_{out}, the output power $P_q^{\text{wind}}(t)$ is equal to the rated power P_r. The braking system of the wind turbine adopts the v_{out} to bring the wind turbine rotor to a standstill to eliminate the risk of damage to the wind turbine rotor due to excessive wind speed. Assuming that the wind speed in the same wind farm is the same and the total number of turbines ω is constant during VM scheduling, then the total output power in the wind farm of DC d_j at time slot t is defined as:

$$Pw_j(t) = \sum_{q=1}^{\omega} P_q^{\text{wind}}(t). \tag{13}$$

Correspondingly, the wind energy generated by the wind turbines in the wind farm of DC d_j at time slot t is:

$$E_j^{\text{wind}}(t) = \int_t^{t+\Delta t} Pw_j(t) \tag{14}$$

3.4.2 Solar power model

Solar Photovoltaic (PV) power generation is mainly related to the solar cell parameters and weather conditions, such as irradiance and temperature. Referring to the existing model [32], we derive the current-voltage characteristic equation for solar power generation, taking into account irradiance G and temperature F, as follows:

$$i(G,F) = I_{\text{sc}}(G,F) - I_o(F)\cdot e^{\frac{v(G,F)+i(G,F)\cdot R_s}{n_s\cdot V_{\text{th}}}}, \tag{15}$$

where V_{th} is the junction thermal voltage, I_{sc} is the short circuit current under Standard Test Condition (STC), *i.e.* $G = 1000\ \text{W/m}^2$, $T = 25\ °\text{C}$. I_o is the saturation current, R_s is the series resistor, n_s is the number of solar cells in the series connected PV panels, *e.g.,* in the 66HS605-625W model panels [38], $n_s = 132$. This research simplifies the derivation process of the solar energy generation model, the output power of the PV panels is estimated as the maximum power that can be extracted from the PV panels, and the output power of the gth PV panel is denoted as:

$$P_g^{solar}(t) = i_{mp}^2 \cdot r_{mp}, \tag{16}$$

where r_{mp} is the optimal load and the i_{mp} is the corresponding current. If there are λ independent PV panels in the solar plant in DC d_j, the total output solar power at time slot t is calculated as:

$$Ps_j(t) = \sum_{g=1}^{\lambda} P_g^{\text{solar}}(t). \tag{17}$$

Correspondingly, the solar energy generated by the solar plant in d_j at time period from t to $t + \Delta t$ is:

$$E_j^{\text{solar}}(t) = \int_t^{t+\Delta t} Ps_j(t)\cdot \Delta t. \tag{18}$$

Specifically, in the simulation process of Section 5.1, we estimate the output power of PV panels under different conditions based on the current-voltage curve provided in the data manual [38], combined with irradiance G, temperature T, and the Temperature coefficient of PV, and the duration of the time slot t is set as one hour.

3.5 Carbon emission and energy cost model

The carbon intensity is applied to evaluate the grams of CO_2 emitted per kilowatt-hour of electricity, which is mainly generated by fossil fuels and other high-carbon energy sources. In this research, the carbon intensity of RES is estimated to be zero, even though there may be a small amount of carbon emission during the storage and transportation processes. Due to variations in energy structure (*e.g.*, different proportions of green and brown energy) and economic structure (*e.g.*, varying proportions of heavy industry and services) across different regions, carbon intensity exhibits significant spatial and temporal dimensions. If the carbon intensity of the DC d_j is $\text{Cer}_j(t)$ at time slot t, the carbon emission of d_j is formalized as:

$$\text{CE}_j(t) = \text{Cer}_j(t) \times E_j^{\text{brown}}(t), \tag{19}$$

where $E_j^{\text{brown}}(t)$ is the brown energy consumption of d_j at time slot t, if the total energy consumption during time slot t is $E_j(t)$ and the available RES is $E_j^G(t)$, then it can be calculated that:

$$E_j^{\text{brown}}(t) = \max\left[E_j(t) - E_j^G(t),\, 0\right], \tag{20}$$

where $E_j^G(t)$ is the sum of wind energy and solar energy provided at time slot t:

$$E_j^G(t) = E_j^{\text{wind}}(t) + E_j^{\text{solar}}(t). \tag{21}$$

Then the total carbon emission of N data centers in the total time slot T is:

$$C_{\text{carb}} = \sum_{t=1}^{T}\sum_{j=1}^{N} \text{CE}_j(t). \tag{22}$$

Since the electricity generated by wind and solar energy is both operated and delivered through grid utilities, we do not distinguish the price differences of each type of energy source, and only consider the hourly fluctuation in electricity prices across different geographic locations. We adopt the Locational Marginal Price (LMP) to represent the energy price, if the LMP of DC d_j at time slot t is $\xi_j(t)$, then the energy cost at time slot t is:

$$Q_j(t) = \xi_j(t) \times E_j(t). \tag{23}$$

Then the total energy cost of N data centers in the total time slot T is:

$$C_{\text{eng}} = \sum_{t=1}^{T}\sum_{j=1}^{N} Q_j(t). \tag{24}$$

3.6 Objective function

The objective of the VM scheduling algorithm in geo-distributed DCs powered by multi-RES is to reduce the energy cost, and carbon emission while ensuring the QoS. In the distributed multi-cloud environment, the VM requests of users are undirected. While assigning tasks to the nearest DC would reduce the processing time, our scheduling algorithm prioritizes dispatching tasks to DCs with more sufficient RES, which may result in an increase in processing time. To address this, we impose a limitation on the total latency of scheduling all tasks.

We define the time T_{r,d_j} that it takes for the user's request to be allocated from the source to DC d_j as the request time. The waiting time and execution time of the task in the cloud are Tw_{rj} and Te_{rj}, respectively. Then the response time of task ψ_r is $Tr_{rj} = T_{r,d_j} + Tw_{rj}$, and the total time to process task ψ_r is:

$$Ta_{rj} = T_{r,d_j} + Tw_{rj} + Te_{rj}. \tag{25}$$

Due to different tasks requiring different execution time the response time Tr_{rj} does not contain information about task execution time. Therefore, we introduce the parameter of task latency L_{rj} to measure the efficiency of task allocation throughout the entire scheduling process, and the L_{rj} is defined as:

$$L_{rj} = \frac{Te_{rj}}{Ta_{rj}}. \tag{26}$$

Then we can get the total task latency of N data centers:

$$L_{\text{delay}} = \sum_{j=1}^{N} \sum_{r=1}^{\delta_j(t)} L_{rj}. \tag{27}$$

When constructing the VM Scheduling strategies (VS) with the proposed algorithm, it is necessary to select the optimal placement solution by combining the optimization goals mentioned above. Therefore, we are committed to minimizing the three objective functions of energy cost (f_1), carbon emission (f_2), and total task latency (f_3). At the same time, the scheduling strategy needs to satisfy the resource capacity constraints of PMs and VMs:

$$\begin{cases} \min f_1(\text{VS}) = \sum_{t=1}^{T} \sum_{j=1}^{N} Q_j(t) \\ \min f_2(\text{VS}) = \sum_{t=1}^{T} \sum_{j=1}^{N} \text{CE}_j(t) \\ \min f_3(\text{VS}) = \sum_{j=1}^{N} \sum_{r=1}^{\delta_j(t)} L_{rj} \end{cases} \tag{28}$$

$$s.t. constraints\,(2) - (4).$$

4 Virtual machine dispatching algorithm

We propose a VM task scheduling algorithm ECMR with multi-RES for geo-distributed DCs to achieve the optimization goals of minimizing energy cost, carbon emission, and total task latency as noted in equation (28). Besides, we design another five variants based on ECMR as the baseline algorithms by changing the priorities of different optimization indicators and the generation conditions of RES, to demonstrate the effectiveness of ECMR in reducing DCs energy consumption and improving the availability and complementarity of RES.

4.1 Energy and Carbon-aware dispatching with Multi-RES (ECMR)

Algorithm 1 displays the pseudocode of the VM scheduling algorithm ECMR proposed in this article, which comprehensively considers the availability of RES, DC computing resources, electricity price, carbon intensity, and other factors. The specific execution steps of the algorithm are as follows:

Step 1: The required distributed DC status information for scheduling is input into CWS, and then calculate the operating energy consumption of each DC and the generation energy of RES (lines 2–3). If the available RES is greater than the consumed energy of the DC, it is classified as D_G; otherwise, it is classified as D_B, indicating the need to compensate for the shortage of renewable energy with brown energy (lines 4–9).

Step 2: Sort the data center $d_j \in D_G$ in ascending order based on the distance from the task source of the cloud user to the DC, and then assign the task ψ_r to the DC closest in distance (lines 12–13) to minimize the response time of tasks while matching the RES availability.

Step 3: Determine whether the servers in the DC to be allocated can provide the computing resources required by the VM. If the available resources meet the task requirements (lines 16–17), based on the Most Effective Server First (MESF) scheduling scheme proposed in [39] assigning the tasks to the most power-efficient server, *i.e.* the server with the least increase in energy consumption (line 18). From this minimize the energy consumption of servers as much as possible.

Step 4: Determine whether the available RES can meet the energy consumption required for task processing if the VM is allocated to the current DC, and whether the task latency is below the set threshold (lines 19–23). This ensures the prioritized use of DCs with sufficient RES for processing tasks.

4.1 If the RES energy is sufficient and the task latency is below the threshold, allocate the VM to the target server and complete the task allocation, proceed to step 6.

4.2 If the requirements cannot be met, return to step 2 to forward the VM to another DC with sufficient renewable energy. If the allocation is completed, proceed to step 6. If the allocation still cannot be completed, it means that a DC with sufficient RES and satisfactory latency cannot be found, proceed to step 5.

Step 5: When balancing the RES availability and task latency is not feasible, the DCs are sorted in descending order based on their available computing resources. Then, tasks are assigned to the DC with the most sufficient available computing resources (lines 26–31). In this scenario, the utilization of RES is disregarded to ensure that the tasks can be ultimately processed.

5.1 If the computing resources of the target DC's server can meet the requirements of the task, allocate the VM to the target server in that DC, the task allocation is completed, and proceed to step 6.

5.2 If the resource requirements cannot be met, it indicates that all DCs are unable to provide the required resources in the current state. In this case, the task cannot be allocated and a failure value is returned.

Step 6: Output the destination DC of the VM request that has been allocated, and update DC status information such as power, available computing resources, energy cost, carbon emission, and task latency in the DC. After the allocated tasks are processed, the occupied VM resources must be released.

Algorithm complexity analysis: Assuming there are N data centers, each with M servers. Firstly, the DCs are categorized based on the availability of RES. After the submission of VM requests, prioritize assigning tasks to the DC closest to the request source and with sufficient green energy. Subsequently, tasks are allocated to the server with the least increase in energy consumption based on the utilization of computing resources of servers. The time complexity of this algorithm is: $\Theta(N \log N) \times \Theta(M \log M)$. When all DCs are short of RES, tasks are sorted in ascending order based on the distance between the request source and the DCs. In this case, tasks are allocated priority to the nearest DC, and then to the server with the least energy consumption growth, at this point, the time complexity of the algorithm is $(\Theta(N) + \Theta(N \log N)) \times \Theta(M \log M)$.

Algorithm 1. Energy and Carbon-aware VM dispatching algorithm

```
Input:  the list of distributed data center $D\_list$, all servers in each data center $S\_list$, request $\psi_r$ in list $\delta_i(t)$
        arriving at time slot $t$, multi-RES generation power at time slot t, task latency threshold $T_{thre}$
Output: Requests dispatched destination
1   for $d_j$ in $D\_list$ do
2       $E^{DC}_{d_j} \leftarrow$ Calculate the energy consumption of $d_j$ at time slot $t$ using Eq. (11)
3       $E^{G}_{d_j} \leftarrow$ Calculate RES generation energy of $d_j$ at time slot $t$ using Eq. (21)
4       $RES\_magin \leftarrow E^{G}_{d_j} - P^{DC}_{d_j}$
5       if $RES\_magin > 0$ then
6           add $d_j$ into $D_G$
7       else
8           add $d_j$ into $D_B$
9       end
10  end
11  for $\psi_r$ in $\delta_i(t)$ do
12      Sort all the DCs in $D_G$ based on the distance between the source of $\psi_r$ and the target DC in ascending order
13      Dispatch the request $\psi_r$ to the nearest data center $d_j$
14      $R_r \leftarrow$ Get the required computing resources of $\psi_r$
15      for $d_j \in D_G$ do
16          $AC_i \leftarrow$ Get the remaining available computing resources of servers in the $S\_list$ of $d_j$
17          if $AC_i > R_r$ then
18              Assign $\psi_r$ to the server $s_i$ in the $S\_list$ that has the least increased energy consumption $\Delta Er$
19              if $RES\_magin > \Delta Er \&\& L_{rj} < T_{thre}$ then
20                  Allocate task $\psi_r$ to server $s_i$ belongs to $d_j$
21                  Update $E^{DC}_{d_j} = E^{DC}_{d_j} + \Delta Er$
22                  Update $RES\_magin$, $C_{carb}$, $C_{eng}$, $L_{delay}$ by using Eq. (22), (24), and (27)
23              end
24          end
25      end
26      Sort the available computing resources $AC_j$ of all DCs in descending order
27      Dispatch the request $\psi_r$ to the data center $d_j$ with the most available resources
28      if $AC_i > R_r$ then
29          Allocate task $\psi_r$ to server $s_i$ belongs to $d_j$
30          Update $E^{DC}_{d_j} = E^{DC}_{d_j} + \Delta Er$
31          Update $RES\_magin$, $C_{carb}$, $C_{eng}$, $L_{delay}$ by using (22), (24), and (27)
32      else
33          The request is paused for dispatching and waiting for available data center
34      end
35  end
```

4.2 Energy and Carbon-aware dispatching with QoS (ECQ)

Different from the ECMR algorithm, the ECQ algorithm pays more attention to the QoS, which is mainly reflected in the response time of VM dispatching. Compared with Algorithm 1, the ECQ reduces the priority of RES and therefore omits lines 5–9 regarding the division of data center RES abundance, and line 15 will no longer restrict the target data center $d_j \in D_G$. In this case, the task will first be allocated to the DC closest to the request source. If the nearest DC cannot provide sufficient computing resources, the request will be prioritized for allocation to a DC with more abundant resources.

4.3 Energy-aware dispatching with non-RES (ECNR)

The ECNR algorithm does not consider the use of renewable energy when formulating VM scheduling strategies, which also makes it difficult to achieve the goal of reducing carbon emissions. It is worth noting that all the algorithms proposed in this article involve the use of renewable energy except ECNR. Specifically, starting from line 3 in Algorithm 1, the code related to RES usage is ignored, only the nearest DC and servers with lower energy consumption growth are considered when scheduling VM. This baseline algorithm serves to underscore the importance of RES in reducing DC carbon emissions, and it is foreseeable that ECNR will result in the highest carbon emissions among all the algorithms.

4.4 Energy and Carbon-aware dispatching with Only Wind energy (ECOW)

As mentioned earlier, renewable energy sources exhibit significant temporal variability and spatial location dependence, we introduce wind and solar energy to flexibly leverage their complementary characteristics in time and space. To demonstrate the temporal complementary characteristic of different types of renewable energy within the same region, we propose the ECOW and ECOS algorithms, which only consider wind or solar energy respectively as the sole renewable energy source. Compared to Algorithm 1, ECOW has been modified in line 3 to calculate only the output power of wind turbines, which may lead to a lack of complementary use of renewable energy when environmental wind speed decreases, directly affecting the RES utilization of the DC.

4.5 Energy and Carbon-aware dispatching with Only Solar energy (ECOS)

The ECOS algorithm is similar to the ECOW algorithm, where DCs in different regions only consider the use of solar energy. Therefore, in Algorithm 1, line 3 is modified to only calculate the output power of solar panels. Since solar energy is mainly affected by irradiance and temperature, solar panels only have significant output power during fixed periods (approximately from 14:00 to 19:00) every day, and in rainy and low-temperature weather, the available time segments for solar energy will be shorter. Compared with the ECOW algorithm, the ECOS algorithm has lower renewable energy availability, which may lead to higher carbon emissions.

4.6 Geo-distributed VM dispatching with Same Weather (GVSW)

To further prove the spatial complementarity of multi-RES, we propose the GVSW algorithm, which assumes that the weather and environmental conditions of geo-distributed DCs are consistent, thus eliminating spatial differences. Compared with Algorithm 1, this baseline algorithm only selects the weather condition of one region as input when calculating the output power of RES in line 3. It can be foreseen that this benchmark algorithm cannot take advantage of the complementary energy advantages brought about by changes in weather conditions in different regions, resulting in a decrease in the utilization of RES and an increase in carbon emissions.

5 Performance validation

In this section, we simulate the operation performance of the cloud Infrastructure as a Service (IaaS) layer using

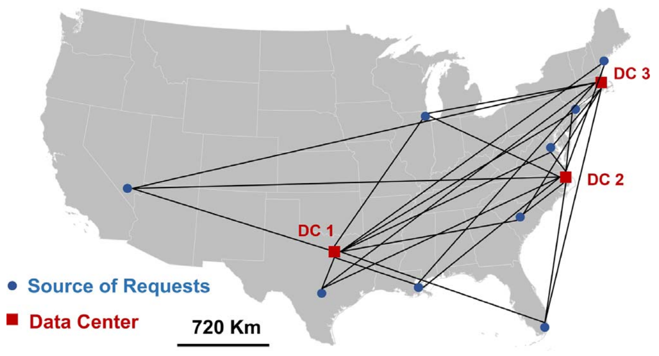

Fig. 2. Locations of the geo-distributed Data Centers (DCs) and requests source. DC1, DC2, and DC3 represent Dallas, Elizabeth City, and Boston respectively.

real-world datasets to evaluate the effectiveness of the proposed algorithm ECMR. The algorithms are executed by using Python IDE with NumPy library as a simulator, which is deployed on a Personal Computer (PC) with an AMD Ryzen 9 7945HX CPU clocked at 2.5 GHz, an NVIDIA GeForce RTX 4060 GPU, and 2× 8 GB DDR5 5200 MHz dual channel RAM. Then the configuration of experimental parameters is introduced in detail and the experimental results are analyzed.

5.1 Experimental design

A detailed description of the real-world data configuration and sources required for the simulation are provided, such as server configurations and their power consumption model, workloads, wind speed, irradiance, temperature, carbon intensity, LMP, PUE, and so on, ensuring the smooth progress of subsequent simulation.

5.1.1 Data centers configuration and sources of requests

To simulate a multi-cloud environment of geo-distributed DCs, as shown in Figure 2, we select three DCs located in Dallas, Elizabeth City, and Boston, as well as nine request sources distributed around them, these locations are located in different time zones of the United States and have significant latitude differences. The main reasons for choosing these three locations as DCs are: (i) We refer to Google's DCs deployment in the United States [40], which has DCs in Texas and Elizabeth City; (ii) These regions exhibit distinct climate variations and have different power suppliers, which can reduce the same environmental conditions caused by geographic proximity, which is more conducive to studying the complementarity of multi-RES.

The number of servers, wind turbines, and PV panels of each DC, along with DC characteristics are presented in Table 1. The parameters are set the same for each DC to avoid the impact of the size of DCs on simulation results. Each selected request source is connected to three DCs. We assume that VM requests are evenly distributed among each source, and CWS allocates VM requests to the optimal DC according to the scheduling strategies. Table 2 lists the distance between all DCs and the request source in

Table 1. Data centers and renewable energy resources distribution characteristics.

DC Number	Location	Time zone	latitude	Number of servers	Number of wind turbines	Number of PV panels
DC1	Dallas, Texas	UTC-6	32°47′ N	120	2	30
DC2	Elizabeth City, North Carolina	UTC-5	36°17′ N			
DC3	Boston, Massachusetts	UTC-5	42°21′N			

Table 2. The straight-line distance from the source of the request to the DCs in kilometers (km).

Source of requests	Dallas	Elizabeth City	Boston	Times zone
Las Vegas, NV	1740	3490	3830	UTC-8
Austin, TX	290	2100	2725	UTC-6
Chicago, IL	1295	1165	1365	UTC-6
New Orleans, LA	720	1450	2170	UTC-6
Miami, FL	1780	1228	2018	UTC-5
Colombia, SC	1500	443	1190	UTC-5
Washington, D.C.	1905	300	630	UTC-5
New York, NY	2215	530	295	UTC-5
Portland, ME	2600	965	160	UTC-5

Table 3. Server types.

Server types	CPU types	CPU cores	RAM (GB)	Clock speed (GHz)	Bandwidth (Gbps)
Huawei RH2285 V2	Intel Xeon E5-2470	2*8	24	2.3	25
Huawei RH2288H V3	Intel Xeon E5-2698 V4	2*20	64	3.6	25
Lenovo SR655 V3	AMD EPYC 9654	1*96	192	2.4	25

kilometers and the time zone (observed during standard time) of each request source, which is obtained from Google Maps [41]. We use these data to find the nearest DC and calculate the additional time incurred when VM requests are scheduled to other DCs.

5.1.2 Servers configuration and VM instances

Referring to related studies using VM instances [16, 26], we configure three types of heterogeneous servers and four types of VM instances for each DC. The server configuration and VM instance of each DC are consistent. Table 3 shows three types of servers (Huawei RH2285 V2, Huawei RH2288H V3, and Lenovo SR655 V3) along with their configuration parameters, with 120 servers in each DC and one-third of each type of server. This configuration is mainly due to the gradual increase in CPU cores and RAM capacity of these three servers, and the heterogeneous architecture of AMD and Intel chips. At the same time, the power consumption of these three servers is lower among servers with the same performance, which also helps to reduce the overall power consumption and carbon emissions of the DC. The difference between each type of server is mainly in CPU cores and RAM capacity, which reflects the difference in the processing capabilities of the server. We assume that all servers use Network Attached Storage (NAS) systems regardless of local hard drives, which facilitates data sharing, collaboration, and access in multi-cloud environments.

According to the results of the SpecPower benchmark [42], Table 4 shows the corresponding relationship between power consumption in watts and utilization of the three types of servers, the current power consumption of the server can be calculated by combining equation (5). It can be found that as the CPU cores and memory capacity of the server increases, its power consumption will also increase. To cope with VM tasks with different resource requirements, Table 5 shows the four types of VM instances we set up. Each instance is created based on the requested number of CPU cores, memory, and storage capacity, and once a task is completed, the occupied resource will be released.

5.1.3 Workload data

To simulate the requests submitted by VMs, we estimated the daily dynamic VM creation submissions of different request sources based on the workload of MetaCentrum 2 [43]. The reason for using this dataset is that it provides a wide range of applications and has been adopted by many researchers in simulation environments. Considering the availability of data required for simulation, the total simulation period is set to 7 days.

Table 4. Power consumption models of servers versus loads in Watts.

Servers	Load										
	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Huawei RH2285 V2	51	72.5	81.9	92	101	114	133	154	178	218	241
Huawei RH2288H V3	43.5	83.7	101	117	131	145	164	187	228	277	329
Lenovo SR655 V3	63.2	124	145	166	186	206	227	244	280	308	351

Table 5. Virtual machine instances.

Instance type	CPU cores	RAM (GB)	Storage (GB)	Clock speed (GHz)	Bandwidth (Gbps)
small	1	2	250	0.5	0.8
medium	2	4	500	1.0	1.5
large	4	8	750	1.5	2
xlarge	8	16	1000	2.0	3

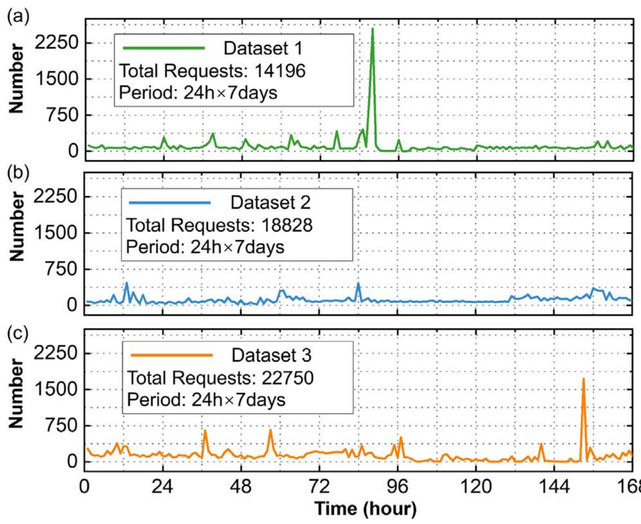

Fig. 3. (a) Dataset 1, (b) Dataset 2, and (c) Dataset 3 hourly workloads of three datasets within 7 days. Datasets 1–3 submitted 14196, 18828, and 22750 VM creation requests respectively.

We select three parts from the workload dataset every week, which are respectively represented as dataset 1, dataset 2, and dataset 3. We check the DC operation logs [43] for abnormally high request load conditions to confirm whether there are abnormal job submissions or system failures. The workloads datasets after outlier processing as shown in Figure 3, during the seven-day simulation period, datasets 1–3 submitted 14196, 18828, and 22750 VM creation requests respectively. The workloads in the real world have obvious peaks and valleys during the day and night, we divide each dataset into 168 time slots with hourly steps. We assume that workloads come from an average of nine request sources, and the time zones of the workloads are adjusted to match the request sources. If the requested resource exceeds what can be provided by all VM instances, the dedicated VM instance will be flexibly created based on demand. This study only focuses on the scheduling of VM tasks and the allocation of server resources, without considering the applications running on the VMs.

5.1.4 Multi-renewable energy sources

To simulate the intermittency and complementarity of wind and solar energy in geo-distributed DCs, we collect wind speed, Global Horizontal Irradiance (GHI), and temperature data at hourly intervals from January 11, 2024, to January 17, 2024, by Windfinder [34] and Solcast [35]. We confirm whether there are abnormal weather conditions based on weather records and remove or correct unexplained anomalies. Figure 4 shows the information on RES for Dallas, Elizabeth City, and Boston throughout the entire simulation period of 168 hours.

It can be observed that wind energy fluctuates irregularly but can achieve power output throughout the day as it is only influenced by wind speed, while solar energy shows obvious peak valley fluctuations as GHI intensity changes every day. The wind energy and solar energy in the same region exhibit obvious complementary characteristics, such as the 84 h–96 h period in Figure 4a, the 48 h–60 h period in Figure 4b, and the 108 h–120 h period in Figure 4c. At the same time, there is a clear spatial complementarity in the availability of renewable energy among the three regions. For example, during the 84 h–96 h period, solar energy is relatively lacking in Dallas, but more abundant in Elizabeth City and Boston. During the 108 h–132 h period, wind energy is relatively abundant in Dallas, but relatively lacking in Elizabeth City and Boston. The temperature curves of the three DCs are shown in Figs. 5a–5c, respectively.

Unlike [29, 30], we do not set up hundreds or even thousands of RES power generation equipment. Firstly, we estimate the required power for 360 servers in three DCs. Secondly, we choose a single power generation equipment with higher output power, finally, we consider the actual construction and operation cost of RES power stations. We assume that each DC is equipped with 2 units of

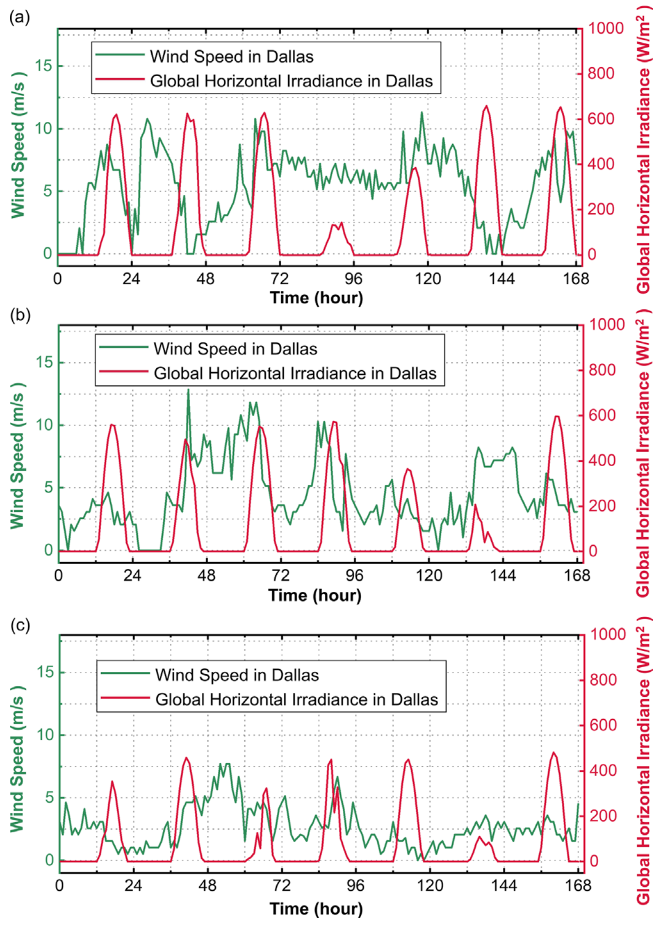

Fig. 4. Wind speed and global horizontal irradiance of (a) Dallas, (b) Elizabeth City, and (c) Boston respectively. Wind energy and solar energy in the same region exhibit obvious complementary characteristics, and there is a clear spatial complementarity in the availability of renewable energy among the three regions.

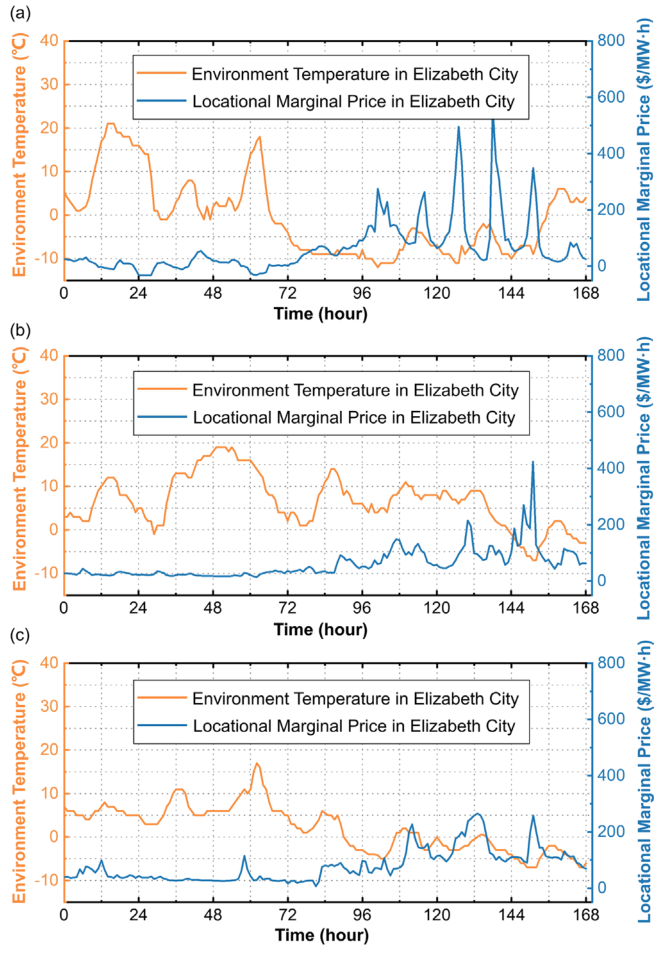

Fig. 5. Environment temperature and locational marginal price of (a) Dallas, (b) Elizabeth City, and (c) Boston respectively. The energy prices will increase significantly as the temperature decreases in winter.

WD-DLK-10 KW wind turbines [44] and 30 units of 66HS605-625 W PV panels [38], the parameters of wind turbines and solar panels are shown in Table 6 and Table 7, respectively.

The available energy for a single wind turbine and a single PV panel in each DC over the entire simulation period are shown in Figures 6 and 7, respectively, which have been adjusted according to the time zone. Compared to the other two regions, Dallas has more powerful wind and solar energy. By deploying geo-distributed multi-renewable energy, the DCs can harness RES at every time interval.

5.1.5 Locational marginal price

To accurately obtain the price differences of geo-distributed power nodes and grasp real-time power price fluctuations, we use locational marginal price from three independent electricity operators ERCOT, PJM, and ISO-NE [45] to calculate the energy cost, the three DCs are powered by their respective local grid utility. Figures 5a–5c show the curves of the LMP changes of the three DCs at hourly intervals respectively. We combine the weather data from [34, 35] to analyze the rationality of electricity price data and remove or revise the unexplained outliers, it can be intuitively found from Figure 5 that energy prices will increase significantly as the temperature decreases in winter.

5.1.6 Carbon intensity and PUE

Due to the integration of various energy carbon emission factors from the local power grid, the hourly carbon intensity data is unstable. Therefore, as shown in Table 8, we choose daily carbon intensity data of three DCs from electricity maps [46] to ensure the accuracy of the data. Compared to the other two locations, the carbon intensity of DC 2 is significantly higher, with an average value of 455 g/KW h. DC 1 and DC 2 have an average carbon intensity of 408 g/KW h and 334 g/KW h, respectively. Referring to Alibaba Cloud [47] for the current state-of-the-art DC, the PUE can reach 1.17. Although the actual PUE of DCs can vary, we assume a fixed PUE of 1.2 in this simulation.

Table 6. Parameters of a wind turbine.

Model	P_r (KW)	v_{in} (m/s)	v_{out} (m/s)	v_r (m/s)	Net weight (kg)	Wind wheel diameter (m)
WD-DLK-10KW	10	2.5	45	10	365	7.2

Table 7. Parameters of a solar Photovoltaic panel in STC.

Model	Max output power (W)	Temperature coefficient (Power)	n_s	Net weight (kg)	Open circuit voltage, v_{oc} (V)	I_{sc} (A)
66HS605-625 W	625	−0.3%/°C	132	29	48.4	16

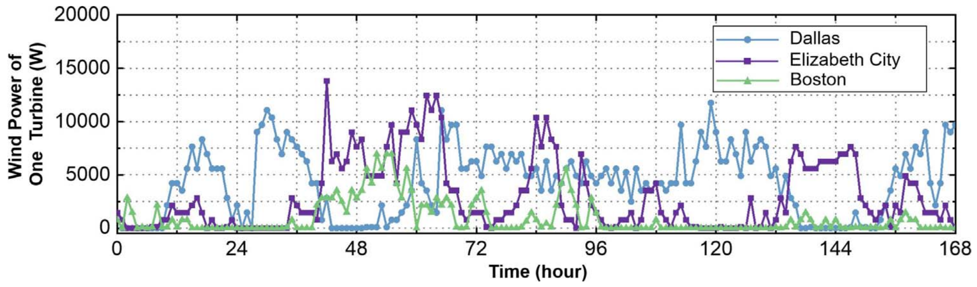

Fig. 6. Output wind power of Dallas in the blue line, Elizabeth City in the purple line, and Boston in the green line.

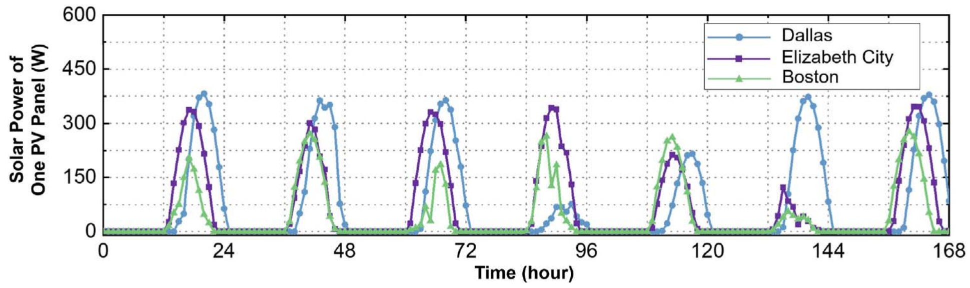

Fig. 7. Output solar power of Dallas in the blue line, Elizabeth City in the purple line, and Boston in the green line.

Table 8. Three DCs carbon intensity (g/KW h).

Data center	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Average
DC1	326	357	373	435	497	450	418	408
DC2	421	424	397	444	510	500	488	455
DC3	316	339	306	286	342	384	362	334

5.2 Experimental results study

In this section, we evaluate the performance of the ECMR algorithm according to the indicators included in the objective function equation (28) and compare the performance with the five baseline algorithms proposed in Section 4. Before the formal test, we select 1 day of data to conduct a pilot test on the ECMR algorithm to ensure that the algorithm can be deployed correctly. In the pilot test, we evaluate the performance of ECMR to ensure that it can work according to the designed program and can process data effectively. Based on the results of the preliminary test, we optimize the ECMR algorithm to ensure that it can achieve the ideal effect in the formal simulation. At the same time, to reduce potential errors and deviations, we select three sets of workload datasets to simulate the operation of the geo-distributed DCs for 7 days. Each dataset is repeated 5 times and the average simulation results are

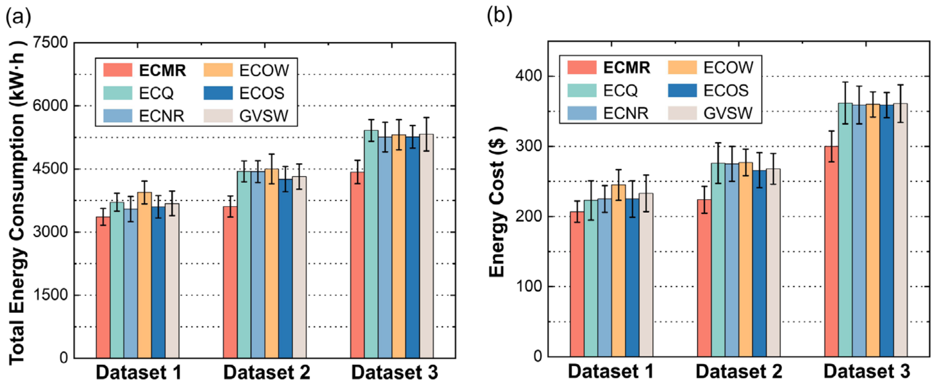

Fig. 8. (a) Total energy consumption, (b) energy cost. Error bars represent the SD of the results from five repeated experiments. ECMR has the lowest energy consumption and energy cost among all algorithms.

reported. By comparing the performance of ECMR with the other five baseline algorithms, ECMR demonstrates its superiority in reducing energy consumption and carbon emissions, improving RES utilization, and ensuring SLA compliance in geo-distributed DCs.

5.2.1 Energy consumption and cost

Figure 8a shows the comparison of total energy consumption between the ECMR algorithm and five other baseline algorithms, and the result shows that ECMR has the lowest energy consumption among all algorithms. Due to ECQ being more focused on dispatching tasks to the nearest DC, optimization of energy consumption and the use of green energy have been neglected, resulting in a significant increase in energy consumption. ECNR and ECOS have relatively good energy consumption performance in the three datasets. This is because compared with ECOW and GVSW, ECNR can focus more on the optimization of energy consumption and the improvement of computing resource utilization, thus avoiding the negative impact of the mismatch between RES availability and DC's energy demand.

The ECOS has lower energy consumption compared with ECOW and GVSW. Different from the method that also considers energy consumption and carbon emissions [28], ECMR does not increase energy consumption due to prioritizing the use of green energy. This is because this method can flexibly utilize the spatiotemporal complementarity between distributed multi-RES to ensure DC's availability of RES during peak load demands. Due to the workloads of Dataset 3 being 60% higher than that of Dataset 1 and 21% higher than that of Dataset 2, the energy consumption will significantly increase as the workloads increase. We average the results of three datasets and find that the energy consumption of ECMR is 15.9%, 14.2%, 17.3%, 12.8%, and 14.4% lower than that of ECQ, ECNR, ECOW, ECOS, and GVSW, respectively.

Figure 8b shows a comparison of energy costs between different algorithms. Although energy cost is a direct result of energy consumption, the increase in energy consumption does not necessarily lead to an increase in energy cost due to the real-time pricing of LMP and significant price differences across distributed regions. Therefore, we find that the energy consumption of the ECQ based on the workloads of Dataset 1 is higher than that of ECNR, ECOS, and GVSW, but its energy cost is lower than these three algorithms. However, when the difference in energy consumption is large, lower energy consumption will inevitably lead to lower energy cost, such as ECMR's simulation results based on three datasets all have the lowest energy cost. We average the results of three datasets, ECMR reduces energy cost by an average of 15.1%, 14.9%, 17.1%, 14.2%, and 15.3% compared to ECQ, ECNR, ECOW, ECOS, and GVSW, respectively.

5.2.2 Multi-RES usage

Figure 9a presents the utilization of renewable energy, which represents the ratio of wind and solar energy to the total energy consumed by the DC. The RES utilization of the ECMR algorithm is the highest in all three datasets, reaching a maximum of 90.8% and an average of 87.4%, which is significantly higher than the renewable energy utilization of the competitive algorithm EFP [28], which is 73.1%. Although both the ECMR and GVSW consider the availability of wind and solar energy, GVSW assumes that the weather and environmental conditions are consistent across different DCs. This results in an average RES utilization of 76.8% for GVSW, which is notably lower than ECMR, this result further demonstrates the advantage of spatial complementarity of RES. Although the ECQ reduces the priority of renewable energy during scheduling, the DCs are still powered by a combination of green energy and brown energy, so its average RES utilization can still reach 69.2%. Since ECOW and ECOS algorithms only consider a single RES, their average RES utilization is only 54.4% and 18.8%, respectively. On the one hand, this is due to the mismatch between the availability of a single RES and the peak workload requests, and on the other hand, there may be a shortage of renewable energy due to environmental impact. Therefore, using multi-RES can effectively compensate for the gaps caused by a single source, and the temporal complementary characteristic ensures the 24/7 availability of green energy for DCs. Because the ECNR algorithm does not consider the use of green energy, its RES utilization is 0.

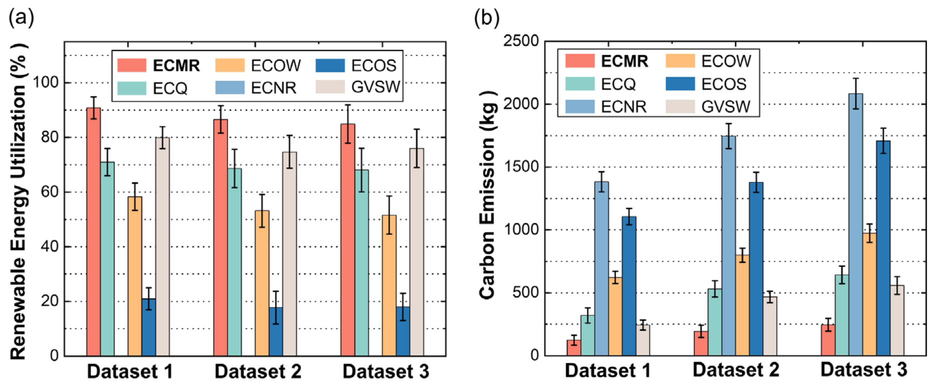

Fig. 9. (a) The renewable energy utilization, (b) carbon emission. Error bars represent the SD of the results from five repeated experiments. ECMR has the highest renewable energy utilization and the lowest carbon emissions among all algorithms.

5.2.3 Carbon emission

Since this study assumes that only brown energy sources result in carbon emission, the result shown in Figure 9b is negatively correlated with the RES utilization rate in Figure 9a, a higher RES utilization leads to lower carbon emission. Therefore, ECMR maintains the lowest carbon emission in simulations based on three datasets, averaging only 210.2 kg. Conversely, ECNR exhibits the highest carbon emission with an average of 1737.6 kg, which is nearly 8 times higher than ECMR. The average carbon emissions of ECQ, ECOW, ECOS, and GVSW are 497.3 kg, 793.2 kg, 1397.5 kg, and 422.4 kg, respectively. These simulation results demonstrate the strong energy and carbon awareness of ECMR, by leveraging the temporal and spatial complementarity of multi-RES, the ECMR not only reduces energy consumption but also maintains a high utilization of renewable energy.

5.2.4 SLA

We evaluate the level of QoS based on two different SLA metrics: (i) the response time of VM tasks, which is the weighted average of task response time in each dataset, and (ii) the VM task failure rate. Figure 10a respectively shows the task response time of all algorithms of Dataset 1–3, the average response time is the average of the results of three datasets. Due to the ECQ algorithm's focus on minimizing task scheduling latency by assigning tasks to the nearest DC, it achieves the shortest average response time of 12.2 ms. On the other hand, ECOS has the longest average response time of 16.6 ms, possibly due to the algorithm spending more time searching for the availability of solar energy. Compared with the competitive WSG algorithm [16] with an average response time of 158.3 ms, the proposed ECMR achieves only 12.6 ms. The average response time of ECNR, ECOW, and GVSW algorithms is 14.7 ms, 15.1 ms, and 13.8 ms, respectively. Comparing the simulation results among the three datasets, we find that due to the randomness in task arrival time, quantity, and resource demands, different algorithms do not exhibit a clear pattern in terms of average response time. The significant error bars in each experiment also indicate poor experimental reproducibility.

Task failure occurs when VM requests cannot be scheduled and processed. Figure 10b compares the task failure rates for each algorithm, ECMR, ECQ, and ECNR have the lowest average task failure rates, which are 1.25%, 1.27%, and 1.3%, respectively. In contrast, the existing VM scheduling algorithm OUR-ACS [26] has an average failure rate of 7.2%. Algorithms that consider only a single RES (ECOW and ECOS) or only a single environmental condition (GVSE) without considering the complementarity show higher task failure rates, therefore, ECOW and GVSE have average task failure rates of 1.36% and 1.37% respectively. And ECOS has the highest average task failure rate among all algorithms, reaching 1.47%.

Since the ECMR algorithm takes into account the spatiotemporal complementarity of RES, it helps balance energy loads and ensure that DCs in different geographical locations have a stable supply of renewable energy, which reduces the possibility of task failure due to local RES shortages. Furthermore, task scheduling across geographical regions can avoid excessive load on a single DC. These factors make ECMR have the lowest failure rate. Although ECQ only considers scheduling tasks to the nearest DC to reduce the response time of task scheduling, ignoring the upper limit of resources that the server can carry and fluctuations in RES supply will lead to a higher failure rate. ECNR only considers cross-regional task scheduling and ignores the supply of RES, so it can only rely on brown energy, which increases the uncertainty of scheduling strategies caused by energy price fluctuations, thus affecting the success of task scheduling. GVSE assumes that the weather and environmental conditions of different DCs are consistent, which means that when faced with adverse environmental conditions such as insufficient wind speed or sunlight, sudden fluctuations in energy supply will reduce the scheduling efficiency of the GVSE, which may result in tasks not being processed manner promptly. Since ECOW and ECOS only consider a single RES, these two algorithms cannot take advantage of the spatiotemporal complementary characteristics between wind and solar energy. For example, solar energy is usually only significant during a specific period of the day (about 2 to 7 p.m.). The output power is constant, and in rainy or low-temperature weather, the effective utilization rate of solar energy will be greatly

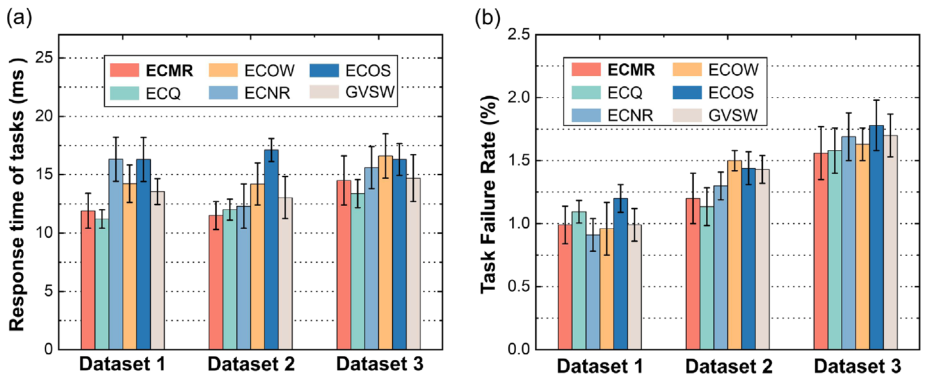

Fig. 10. (a) Response time of tasks. (b) Task failure rate. Error bars represent the SD of the results from five repeated experiments. The average response time of ECMR is 12.6 ms and it achieves an average task failure rate of only 1.25%.

shortened, so the ECOS algorithm will take longer to traverse the solar energy availability of each data center. Longer response time will lead to more tasks exceed the latency threshold T_{thre}, thereby increasing the possibility of task failure.

5.2.5 Robustness test

We conduct robustness tests on the ECMR to analyze its stability and effectiveness under various stress conditions, we regard the datasets and parameters configured in Section 5.1 as the original configuration. First, we add two scenarios: medium workloads and heavy workloads, which correspond to 3 times and 5 times the original workloads, respectively. The rest of the datasets and parameter settings remain the same as the original conditions. Afterwards, we simulate the case of DC resource constraints, including CPU and memory limits to 1/2 and 1/5 of the original conditions. Finally, we simulate the case of renewable energy availability constraints, reducing the availability of RES to 1/2 and 1/5 of the original conditions. Like the original configuration, we select three sets of workload datasets to simulate the operation of the geo-distributed DCs for 7 days. Each dataset is repeated 5 times and the average simulation results are reported. The results of the robustness test are shown in Table 9. As the workload increases, the energy consumption and electricity price of the DC increase significantly, and the utilization of RES also shows a clear downward trend, which in turn leads to an increase in carbon emissions. Since a large amount of workload will lead to a shortage of DC resources and increase the decision-making time of CWS, the response time and task failure rate will be increased. However, experiments have shown that even with 5 times the normal requests, the ECMR algorithm can still effectively balance resource usage and ensure stable operation of the system by adjusting task scheduling. However, if the available resources of the DC, including CPU and memory are limited, it will have a significant impact on the response time and task failure rate of ECMR. In particular, when the available resources of the DC are reduced to 1/5 of the original configuration, the task failure rate increases by more than 5 times. This is mainly because the shortage of resources causes tasks to queue longer and are difficult to schedule. However, in this case, the ECMR algorithm can still ensure the completion of key tasks maximize the utilization of RES, and reduce carbon emissions. Limiting the availability of RES will have a significant impact on energy consumption, electricity prices, RES utilization, and carbon emissions. This is mainly because ECMR cannot make use of the spatiotemporal complementarity of RES, resulting in increased energy consumption. At the same time, due to the reduction in the total amount of RES in the edge-cloud continuum environment, RES utilization and carbon emissions have increased significantly. However, the ECMR algorithm can still ensure the response time of the task and keep the task failure rate at a low level. Robustness tests have proved that the ECMR algorithm can maintain good performance in complex, changing, and stressful environments.

5.3 Discussions

The VM scheduling algorithm for sustainable DCs is subject to various constraints, which are summarized in Table 10 along with metrics for algorithm optimization. Compared with existing works, the ECMR proposed in this paper considers distributed DCs in different time zones and various types of green energy sources *e.g.* wind energy and solar energy, which can dynamically allocate heterogeneous VM tasks between geo-distributed DCs by utilizing the complementarity of multi-RES. Furthermore, the proposed method takes into account the variations in electricity prices and carbon intensity across different regions and minimizes the energy consumption and carbon emissions of DCs while ensuring QoS.

The ECMR algorithm significantly reduces the energy consumption and cost of DCs and reduces dependence on fossil fuels by optimizing VM scheduling and utilizing RES. This algorithm improves computing resource utilization, reduces resource waste, and reduces unnecessary expenses. At the same time, it helps reduce the carbon emissions of DCs, avoids potential additional costs, and enhances the market competitiveness and service quality of CSPs. Although some investment may be required in the early stages of implementation, it can be rewarded in the long run by saving energy costs and improving efficiency and may receive financial incentives from the government for

Table 9. Robustness test results (each of the three sets of workloads is replicated five times and calculate average values).

Parameter	Original results	3 times workload	5 times workload	1/3 of DC resources	1/5 of DC resources	1/3 of RES availability	1/5 of RES availability
Energy consumption/kW h	3800	12,160	20,900	4015	4569	3987	4235
Energy cost/$	243	802	1361	257	278	285	302
The RES utilization	87.4%	82.2%	79.3%	85.2%	83.6%	28.6%	15.8%
Carbon emission/kg	210.2	735.7	1196.2	218.9	225.6	689	1287
Response time/ms	12.6	13.5	14.6	16.2	20.5	13.6	14.2
Task failure rate	1.25%	2.14%	3.21%	4.8%	6.9%	1.27%	1.29%

Table 10. Comparison of related work.

Reference	Distributed multi-clouds	Task scheduling	Heterogeneous tasks	Multi-RES	RES complementarities	Metrics			
						Energy cost	QoS	Carbon emission	RES utilization
[4]	✓	✓	✗	✗	✗	✗	✓	✓	✗
[15]	✗	✗	✗	✓	✓	✓	✗	✓	✓
[16]	✓	✓	✗	✗	✗	✗	✓	✓	✓
[24]	✗	✓	✓	✗	✗	✓	✓	✓	✗
[27]	✓	✓	✓	✗	✗	✓	✗	✓	✗
[28]	✓	✓	✗	✗	✗	✗	✓	✓	✓
[29]	✓	✓	✓	✓	✗	✓	✗	✓	✗
[30]	✓	✗	✗	✓	✗	✗	✓	✗	✓
[32]	✓	✓	✗	✓	✗	✓	✗	✓	✗
[48]	✓	✓	✗	✓	✗	✓	✗	✗	✗
[49]	✗	✓	✗	✗	✗	✓	✗	✗	✓
This work	✓	✓	✓	✓	✓	✓	✓	✓	✓

energy conservation and emission reduction, thereby further reducing operating costs.

In the future, we will study the application of the proposed algorithm in real production environments. ECMR can be deployed in a variety of environments, such as large CSPs, research institutions, enterprise DCs, and edge computing scenarios to optimize resource allocation and reduce operating costs while meeting sustainable development goals. Although the ECMR performs well in simulation environments, there are still some potential defects and challenges, such as: (i) The ECMR algorithm may have delays when processing large-scale data sets, which will affect the real-time performance of task scheduling. (ii) The ECMR algorithm needs to be seamlessly integrated with the existing DC management system and ensure compatibility with various hardware devices and software platforms. (iii) When scheduling tasks across DCs, the security of data transmission and the protection of user privacy needs to be given special consideration. (iv) As the scale of DCs grows, the ECMR algorithm needs to have good scalability and easy maintenance. Therefore, in order to be applied in practical scenarios, ECMR needs to take the following measures to meet these challenges: (i) Adopt streaming data processing technology and distributed computing frameworks, such as Apache Kafka and Apache Spark, to achieve rapid processing of real-time data. (ii) Standardized interfaces and protocols should be adopted to ensure that ECMR can interact smoothly with existing systems, and algorithm adaptation should be done based on different hardware and software platforms to ensure the compatibility of ECMR. (iii) Encryption technology and strict access control policies can be used to ensure data security and privacy and comply with relevant data protection regulations. (iv) The ECMR algorithm should be developed using the principle of modular design, and containerization and microservice architecture should be used to improve the flexibility and scalability of the system.

6 Conclusions and future work

In this paper, we proposed the scheduling method called ECMR of DML task (VM request) from the perspective of spatiotemporal complementarity of multi-RES in the geo-distributed data centers of the edge-cloud continuum environment, which can reduce energy consumption and carbon emissions of DCs while ensuring user QoS.

We addressed the problem of mismatched peak times and spatial distribution between computational resources, power loads, and the multi-RES. To effectively utilize the complementarity of RES, we explicitly modeled the wind and solar resources from both temporal and spatial dimensions based on the differences in geographical location and weather of DCs located in Dallas, Elizabeth City, and Boston. We conducted extensive simulations using real-world datasets in the Python IDE and compared the ECMR algorithm with the five proposed baseline algorithms. The results indicated that the ECMR can significantly reduce the total power consumption, energy cost, and carbon emissions of DCs while maintaining an acceptable QoS. Compared with the five baseline algorithms, the proposed ECMR can achieve a maximum renewable energy utilization of 90.8% in simulation, and reduce the total energy consumption, carbon emissions, and energy cost on average by 14.9%, 61.7%, and 15.3%, meanwhile, the average task latency with 12.6 ms and task failure rate with 1.25% was guaranteed. The significance of the complementary nature of multi-RES in improving the green energy utilization of DCs was demonstrated. The robustness test shows that the ECMR algorithm can still maintain good performance under stress conditions such as resource constraints and reduced RES supply.

In future work, we plan to employ the dual-Multilayer Perception (MLP) frequency domain learning prediction algorithm to predict renewable energy availability and VM requests. This helps CWS plan resource allocation in advance to avoid insufficient or excessive resources, and high-energy-consuming VM requests can be processed in DCs with sufficient RES and lower costs, thereby reducing the overall operating costs and carbon emissions of the edge-cloud continuum. Additionally, we also consider evaluating the proposed approach as middleware programs in a real virtualized cloud environment.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant 52306178, the China Telecom Cloud Computing Hybrid Cloud Research Fund, the High-Performance Computing Center of Central South University, and the Open Foundation of Key Laboratory of Energy Thermal Conversion and Control of Ministry of Education.

Conflicts of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Authorship contribution statement

Zicong Miao: Methodology, Investigation, Data curation, Writing-Original draft. **Lei Liu:** Supervision, Conceptualization, Writing-Review and Editing, Funding acquisition. **Haijing Nan**: Software, Visualization, Writing-Original draft. **Weize Li**: Investigation, Formal analysis, Writing-Review and Editing. **Xiaodong Pan**: Investigation, Visualization, Resources. **Xin Yang**: Validation, Conceptualization, Writing-Review and Editing. **Mi Yu**: Investigation, Funding acquisition. **Hui Chen**: Formal analysis, Investigation. **Yiming Zhao**: Data curation, Writing-Review and Editing.

References

1 Sartzetakis I., Soumplis P., Pantazopoulos P., Katsaros K.V., Sourlas V., Varvarigos E.M. (2022) Resource allocation for distributed machine learning at the edge-cloud continuum, in: *ICC 2022-IEEE International Conference on Communications, Seoul, South Korea, May 2022*, pp. 5017–5022.

2 Marozzo F., Orsino A., Talia D., Trunfio P. (2023) Scaling machine learning at the edge-cloud: a distributed computing perspective, in: *19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT), June 2023, Coral Bay, Pafos, Cyprus*, IEEE, pp. 761–767.

3 Patel Y.S., Townend P., Singh A., Östberg P.O. (2024) Modeling the Green Cloud Continuum: integrating energy considerations into Cloud–Edge models, *Cluster Comput.* **27**, 4095–4125.

4 Zhang G., Zhang S., Zhang W., Shen Z., Wang L. (2020) Distributed energy management for multiple data centers with renewable resources and energy storages, *IEEE Trans. Cloud Comput.* **10**, 4, 2469–2480.

5 Mahbod M.H.B., Chng C.B., Lee P.S., Chui C.K. (2022) Energy saving evaluation of an energy efficient data center using a model-free reinforcement learning approach, *Appl. Energy* **322**, 119392.

6 Ren C., Wang D., Urgaonkar B., Sivasubramaniam A. (2012) Carbon-aware energy capacity planning for datacenters, in: *2012 IEEE 20th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, August 2012, Arlington, Virginia*, pp. 391–400.

7 Busmachiu S. (2023) *The energy consumption of data centers environmental concerns*, Accessed January 16, 2024, available at: https://utilitiesone.com/the-energy-consumption-of-data-centers-environmental-concerns.

8 Vidal J. (2017) *Tsunami of data could consume one fifth of global electricity by 2025*. Available at: http://t.cn/EMNGBNm. [Accessed January 17, 2024].

9 Gandhi A., Harchol-Balter M., Das R., Lefurgy C. (2009) Optimal power allocation in server farms, *ACM SIGMETRICS Perform. Eval. Rev.* **37**, 1, 157–168.

10 Mao H., Schwarzkopf M., Venkatakrishnan S.B., Meng Z., Alizadeh M. (2019) Learning scheduling algorithms for data processing clusters, in: *Proceedings of the ACM special interest group on data communication, August 2019, Beijing, China*, pp. 391–400.

11 Elmougy S., Sarhan S., Joundy M. (2017) A novel hybrid of Shortest job first and round Robin with dynamic variable quantum time task scheduling technique, *J. Cloud Comput.* **6**, 1–12.

12 Bosker B. (2010) *Google buys 20 years' worth of wind energy to power data centers*. Accessed January 17, 2024, available at: https://www.huffpost.com/entry/google-wind-farm-purchase_n_653146.

13 Kamiya G. (2019) *Data centres and energy – from global headlines to local headaches*. Available at: Accessed January 18, 2024. https://www.iea.org/commentaries/data-centres-and-energy-from-global-headlines-to-local-headaches.

14 Apple (2018) *Apple now globally powered by 100 percent renewable energy*, Accessed January 17, 2024, available at:

https://www.apple.com/hk/en/newsroom/2018/04/apple-now-globally-powered-by-100-percent-renewable-energy/.

15 Xu D., Xiang S., Bai Z., Wei J., Gao M. (2023) Optimal multi-energy portfolio towards zero carbon data center buildings in the presence of proactive demand response programs, *Appl. Energy* **350**, 121806.

16 Xu M., Buyya R. (2020) Managing renewable energy and carbon footprint in multi-cloud computing environments, *J. Parallel Distrib. Comput.* **135**, 191–202.

17 Wu C.M., Chang R.S., Chan H.Y. (2014) A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters, *Future Gener. Comput. Syst.* **37**, 141–147.

18 Chen H., Caramanis M.C., Coskun A.K. (2014) Reducing the data center electricity costs through participation in smart grid programs, in: *International green computing conference, November 2014, Dallas, Texas*, pp. 1–10.

19 Wierman A., Andrew L.L., Tang A. (2009) Power-aware speed scaling in processor sharing systems, in: *IEEE INFOCOM 2009, April 2009, Rio de Janeiro, Brazil*, pp. 2007–2015.

20 Verma A., Pedrosa L., Korupolu M., Oppenheimer D., Tune E., Wilkes J. (2015) Large-scale cluster management at Google with Borg, in: *Proceedings of the 10th European Conference on Computer Systems, April 2015, Bordeaux, France*, pp. 1–17.

21 Vavilapalli V.K., Murthy A.C., Douglas C., Agarwal S., Konar M., Evans R., Graves T., Lowe J., Shah H., Seth S., Saha B., Curino C., O'Malley O., Radia S., Reed B., Baldeschwieler E. (2013) Apache hadoop yarn: Yet another resource negotiator, in: *Proceedings of the 4th annual Symposium on Cloud Computing, October 2013, Santa Clara, California*, pp. 1–16.

22 Regaieg R., Koubàa M., Ales Z., Aguili T. (2021) Multi-objective optimization for VM placement in homogeneous and heterogeneous cloud service provider data centers, *Computing* **103**, 1255–1279.

23 Khoshkholghi M.A., Derahman M.N., Abdullah A., Subramaniam S., Othman M. (2017) Energy-efficient algorithms for dynamic virtual machine consolidation in cloud data centers, *IEEE Access* **5**, 10709–10722.

24 Liu W., Yan Y., Sun Y., Mao H., Cheng M., Wang P., Ding Z. (2023) Online job scheduling scheme for low-carbon data center operation: An information and energy nexus perspective, *Appl. Energy* **338**, 120918.

25 Wu Q., Ishikawa F., Zhu Q., Xia Y. (2016) Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters, *IEEE Trans. Serv. Comput.* **12**, 4, 550–563.

26 Khodayarseresht E., Shameli-Sendi A. (2023) A multi-objective cloud energy optimizer algorithm for federated environments, *J. Parallel Distrib. Comput.* **174**, 81–99.

27 Wang P., Liu W., Cheng M., Ding Z., Wang Y. (2022) Electricity and carbon-aware task scheduling in geo-distributed internet data centers, in: *2022 IEEE/IAS Industrial and Commercial Power System Asia (ICPS Asia), July 2022, Shanghai, China*, pp. 1416–1421.

28 Zhao D., Zhou J. (2022) An energy and carbon-aware algorithm for renewable energy usage maximization in distributed cloud data centers, *J. Parallel Distrib. Comput.* **165**, 156–166.

29 Gu C., Fan L., Wu W., Huang H., Jia X. (2018) Greening cloud data centers in an economical way by energy trading with power grid, *Future Gener. Comput. Syst.* **78**, 89–101.

30 Zhang Y., Wang Y., Wang X. (2011) Greenware: Greening cloud-scale data centers to maximize the use of renewable energy, in: *ACM/IFIP/USENIX 12th International Middleware Conference, December 12–16, 2011, Lisbon, Portugal*, pp. 143–164.

31 Khosravi A., Nadjaran Toosi A., Buyya R. (2017) Online virtual machine migration for renewable energy usage maximization in geographically distributed cloud data centers, *Concurr. Comput. Pract. Exp.* **29**, 18, e4125.

32 Deng W., Liu F., Jin H., Li B., Li D. (2014) Harnessing renewable energy in cloud datacenters: opportunities and challenges, *IEEE Netw.* **28**1, 48–55.

33 Aslam S., Aslam S., Herodotou H., Mohsin S.M., Aurangzeb K. (2020) Towards energy efficiency and power trading exploiting renewable energy in cloud data centers, in: *2019 International Conference on Advances in the Emerging Computing Technologies (AECT), February 2020*, pp. 1–6.

34 Windfinder (2024) *Wind and temperature data*. Accessed January 8, 2024, available at: https://www.windfinder.com/.

35 Solcast (2024) *Irradiance data*. Accessed January 8, 2024, available at: https://toolkit.solcast.com.au/world-api.

36 Li Y., Wang Y., Yin B., Guan L. (2012) An online power metering model for cloud environment, in: *2012 IEEE 11th International Symposium on Network Computing and Applications, August 2012, Massachusetts, Cambridge*, pp. 175–180.

37 Fan X., Weber W.D., Barroso L.A. (2007) Power provisioning for a warehouse-sized computer, *ACM SIGARCH Comput. Architect. News* **35**, 2, 13–23.

38 TW Solar (2024) *66HS605-625 W Model PV Panels*. Accessed January 2, 2024, available at: https://www.tw-solar.com/module.html.

39 Dong Z., Zhuang W., Rojas-Cessa R. (2014) Energy-aware scheduling schemes for cloud data centers on Google trace data, in: *2014 IEEE Online Conference on Green Communications (OnlineGreenComm), November 2014*, pp. 1–6.

40 Google (2024) *Google data center locations*. Accessed January 4, 2024, available at: https://www.google.com/intl/zh-CN/about/datacenters/locations/.

41 Google (2024) *Google maps*. Accessed January 4, 2024, available at: https://www.google.com/maps.

42 Spec. (2024) *SpecPower*. Accessed January 10, 2024, available at: https://www.spec.org/benchmarks.html#power.

43 Parallel Workloads Archive (2013) *The metacentrum 2 log*. Accessed January 5, 2024, available at: https://www.cs.huji.ac.il/labs/parallel/workload/l_metacentrum2/index.html.

44 Weide (2024) *DLK-10KW Wind Turbine*, Accessed January 2, 2024, available at: http://weidewind.com/list_24/31.html.

45 Engie Resources (2024) *Market-data*. Accessed January 10, 2024, available at https://www.engieresources.com/market-data.

46 Electricity Maps (2024) *Carbon intensity*. Accessed January 10, 2024, available at: https://app.electricitymaps.com/zone/US-NW-PACE.

47 Alibaba (2019) *PUE of Alibaba cloud green data center*. Accessed January 12, 2024, available at: https://developer.aliyun.com/article/691750.

48 Han O., Ding T., Yang M., Jia W., He X., Ma Z. (2024) A novel 4-level joint optimal dispatch for demand response of data centers with district autonomy realization, *Appl. Energy* **358**, 122590.

49 Goiri Í., Le K., Haque M.E., Beauchea R., Nguyen T.D., Guitart J., Bianchini R. (2011) Greenslot: scheduling energy consumption in green datacenters, in: *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, November 2011, Seattle, Washington*, pp. 1–11.