Future Generation Computer Systems 161 (2024) 302–314

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Energy-aware virtual machine placement based on a holistic thermal model for cloud data centers

Jianpeng Lin [a], Weiwei Lin [a,b,*], Wentai Wu [c], Wenjun Lin [a], Keqin Li [d]

[a] Department of Computer Science and Engineering, South China University of Technology, GuangZhou, 510000, China
[b] Peng Cheng Laboratory, Shenzhen, 518066, China
[c] College of Information Science and Technology, Jinan University, GuangZhou, 510000, China
[d] Department of Computer Science, State University of New York, New York, 12561, New Paltz, United States of America

ARTICLE INFO

Keywords:
Data center
Thermal model
VM placement
Swarm intelligence algorithm
Energy saving

ABSTRACT

As energy-intensive infrastructures, data centers (DCs) have become a pressing challenge for managers due to their significant energy consumption and carbon emissions. Information technology (IT) and cooling systems contribute the most to energy consumption. Energy-aware virtual machine (VM) scheduling methods have been widely demonstrated to reduce energy consumption and operating costs in DCs. However, as realistic DCs exhibit complex power and thermodynamic behaviors, existing works cannot provide efficient measures to optimize computing and cooling power consumption simultaneously. To overcome this challenge, we construct a holistic thermal model (including CPU and server inlet thermal models) to accurately represent the non-uniform, dynamic thermal environment. Subsequently, this work proposes a thermal model-based energy-aware VM placement method (TEVP) to minimize the holistic energy consumption of the DCs, considering resource and thermal constraints. We develop a novel hybrid swarm intelligence algorithm (DE-ERPSO) combining differential evolution (DE) and particle swarm optimization with an elite re-selection mechanism (ERPSO) to explore more energy-efficient VM placement schemes. Extensive experiments are conducted on an extended CloudSim to validate the performance of the proposed TEVP using real-world workload traces (PlanetLab and Azure). Results show that TEVP saves over 5.6% of the total energy consumption over the advanced baselines while maintaining low thermal violations.

1. Introduction

DCs, as contemporary digital infrastructures, significantly promoted the transformation and advancement of the global digital industry. Recently, governments have paid great attention to the data center industry, increasing policy support and financial investment. According to Gartner,[1] the global public cloud market reached $491 billion in 2022, demonstrating a growth rate of 19%. Simultaneously, cloud DCs' huge energy consumption and carbon emissions have emerged as pressing challenges for governments and industries. The journal Science reported that global data center energy consumption was estimated at 1% of total global electricity usage in 2018, or about 205 TW·h [1]. Moreover, the data center industry contributes 0.3% of global carbon emissions and will continue to grow over the next decade [2]. Therefore, leading economies have begun formulating policies and taking measures to promote energy-efficient, low-carbon, and sustainable development of DCs. For example, the Chinese government has announced it will achieve peak carbon emission and neutrality by 2030 and 2060, respectively [3]. The US state of California introduced a carbon-neutral law to reduce greenhouse gas emissions mainly from electricity generation facilities [2]. Additionally, several European cloud operators have pledged their commitment to the European Green Deal, with the aim of achieving climate neutrality by 2030 [4]. A series of green targets and policies are compelling DC owners to take proactive measures to coordinate energy-intensive systems to enhance the holistic energy efficiency of DCs.

IT and cooling systems are the two primary energy-consuming facilities in DCs, accounting for about 80% of the total energy consumption [5]. Numerous servers in DCs are under load or idle for a long time, generating unwanted energy wastage [6]. Therefore, energy-aware VM placement approaches are widely adopted to consolidate VMs to fewer servers and shut down idle servers to save IT energy [7,8]. However,

* Corresponding author at: Department of Computer Science and Engineering, South China University of Technology, GuangZhou, 510000, China.
E-mail addresses: csjianpenglin@mail.scut.edu.cn (J. Lin), linww@scut.edu.cn (W. Lin), wentaiwu@jnu.edu.cn (W. Wu), 202221044267@mail.scut.edu.cn (W. Lin), lik@newpaltz.edu (K. Li).

[1] <Gartner Forecasts Worldwide Public Cloud End-User Spending to Reach Nearly $600 Billion in 2023>, Gartner, April, 2023.

https://doi.org/10.1016/j.future.2024.07.020
Received 24 January 2024; Received in revised form 1 April 2024; Accepted 13 July 2024
Available online 17 July 2024

while VM consolidation significantly improves server resource utilization and energy efficiency, it also leads to overloaded and overheated hosts, which forces the cooling system to provide additional cooling supplies to cool down. Consequently, it is imperative to address the elevated cooling loads and thermal risks stemming from VM consolidation to effectively reduce both computing and cooling energy consumption in DCs.

However, the complex coupling of airflow organization and heat transfer in realistic DCs poses great challenges for efficient thermal management. Multiple variables, such as IT equipment's heat generation characteristics, the cooling system's heating extraction efficiency, and the infrastructure layout in DCs combine to form a time-varying, non-uniform thermal field [9]. Conventional static thermal re-circulation models suffer from low predictive accuracy and fail to capture the evolution and characteristics of the thermal field precisely. Furthermore, thermal models based on computational fluid dynamics (CFD) provide a comprehensive and precise representation of thermal fields. Still, their suitability for real-time temperature assessment and simulation is hindered by substantial computational requirements and the need for extensive parameter inputs [10]. Hence, to tackle this challenge, this work adopts a data-driven thermal modeling approach to construct a server inlet temperature model to quickly and accurately evaluate the thermal distribution of the server room. Furthermore, a CPU thermal model based on the Resistor-Capacitance (RC) model is used to simulate the thermal behavior of the chip. Subsequently, the developed thermal models guide VM placement and migration to minimize the energy consumption of the DCs' IT and cooling systems while meeting service level agrees (SLAs) and thermal constraints. Besides, the existing heuristic-based methods [11,12] for solving large-scale scheduling problems are susceptible to falling into the local optimum trap. Some VM placement methods based on swarm intelligence algorithms [8,13] have limitations in global optimal search. Therefore, we design a hybrid swarm intelligence algorithm that incorporates population diversity and elite reselection mechanisms to enhance the optimal solution search capability of the particle swarm optimization (PSO) algorithm. Moreover, most existing energy-aware VM placement works [8,14,15] consolidate VMs to as few hosts as possible to reduce IT energy consumption. However, these works make high-load active hosts prone to localized hotspots, generating more cooling energy consumption. Therefore, this work constructs a thermal model to guide VM placement and cooling management while reducing the overall energy consumption of computing and cooling facilities. In summary, the noteworthy contributions of this work are highlighted below:

(1) This work designs a thermal model-based VM placement approach to reduce the holistic energy consumption of DCs by 5.6% to 45.5% while maintaining acceptable SLA violation (SLAV) and thermal critical violation (TCV).
(2) We develop a holistic thermal model (including CPU and server inlet temperature models) to accurately represent the dynamic thermodynamic environment of DCs for guiding VM placement and dynamic cooling control.
(3) We design a novel hybrid swarm intelligence algorithm (DE-ERPSO) combining DE and PSO with an elite re-selection mechanism to explore more energy-efficient VM placement solutions.

The remainder of this paper is organized as follows. Section 2 presents the research advances and challenges of the related work. The system model and problem statement are given in Section 3. Section 4 details the principle and implementation of the proposed DE-ERPSO. Subsequently, the energy-aware VM placement method is presented in Section 5. Section 6 shows the simulation experiments and results. Section 7 discusses the conclusion and outlook. All abbreviations and full names in this manuscript can be found in Appendix.

2. Related work

2.1. Heuristic algorithm

Traditional heuristic algorithms are widely used for thermal management in DCs. Tang et al. [16] first proposed a low-complexity thermal re-circulation model to characterize the cross-interference behavior between heat sources in a server room. Subsequently, a thermal-aware workload scheduling method based on sequential quadratic programming is designed to minimize the peak inlet temperature of the hosts, thus reducing the cooling power. In addition, the work [17] developed a thermal imbalance model to predict future thermal profiles. Subsequently, a proactive VM allocation and migration scheme was designed to relocate and migrate more VMs to cooler servers. Similarly, Ilager et al. [18] train an XGBoost-based CPU thermal model to evaluate the thermal behavior of hosts based on server runtime records in a private cloud. Subsequently, a heuristic algorithm based on the thermal model was designed to allocate or migrate VMs to the coldest hosts, thus reducing peak host temperatures and cooling demand.

Moreover, the work [11] selected the VM placement solution with a minimal increase in server and CRAC power consumption. Likewise, Ilager et al. [12] mitigated the thermal gradient by dynamically migrating the VMs of overloaded hosts while shutting down the underloaded hosts to reduce energy consumption. The proposed method uses two greedy searches to find a VM placement scheme with minimum power consumption and a global cooling set-point that avoids hotspots. However, the work assumes that the flow field in the server room is stable and the CRAC supply air temperature is fixed, ignoring the effect of cooling variations on the thermal field. Moreover, to address the resource and thermal heterogeneity of hosts in clusters, the work [19] proposed a dynamic thermal management (DTM) technique that employs VM migration and dynamic voltage frequency scaling (DVFS) methods to tune the power distribution of clusters to improve performance and reduce the overall energy consumption. Note that the work considers the performance impact of heat transfer from neighboring servers to avoid local thermal risks. Overall, heuristic algorithms usually find sub-optimal scheduling solutions quickly for online scenarios. However, as the problem size increases, the heuristic algorithm tends to fall into local optimum, which reduces the possibility of reaching an optimal solution.

2.2. Swarm intelligence algorithm

Compared to traditional heuristic algorithms, swarm intelligence optimization algorithms exhibit more robust global search capabilities by mimicking swarms' social behavior and collective intelligence in solving problems. Therefore, some works have applied swarm intelligence algorithms to thermal-aware VM placement optimization scenarios [13,20,21]. For example, Feng et al. [22] proposed a two-step algorithm to reduce data center overhead from three aspects: cooling, computing, and networking. First, a simulated annealing (SA) algorithm is taken to formulate a VM placement scheme that minimizes the total power consumption. Subsequently, the migration distance of VMs with high traffic overhead is shortened to reduce the network overhead. Also, AghasiA et al. [21] proposed a VM placement scheme based on a gravity search algorithm to minimize the computational and cooling power consumption, considering the heterogeneity of the requested resources. Recently, Liu et al. [23] proposed a multi-objective optimal VM placement scheme considering VM migration cost and energy consumption. A swarm intelligence algorithm mixing pathfinder and genetic algorithm (GA) is designed to explore a better VM placement scheme. However, the single-set-point cooling control method can easily lead to over-cooling. Moreover, the PSO algorithm has been widely adopted to solve the energy-aware VM placement optimization problem. The work [14] improves PSO by redefining the parameters and encoding methods to find the optimal VM placement

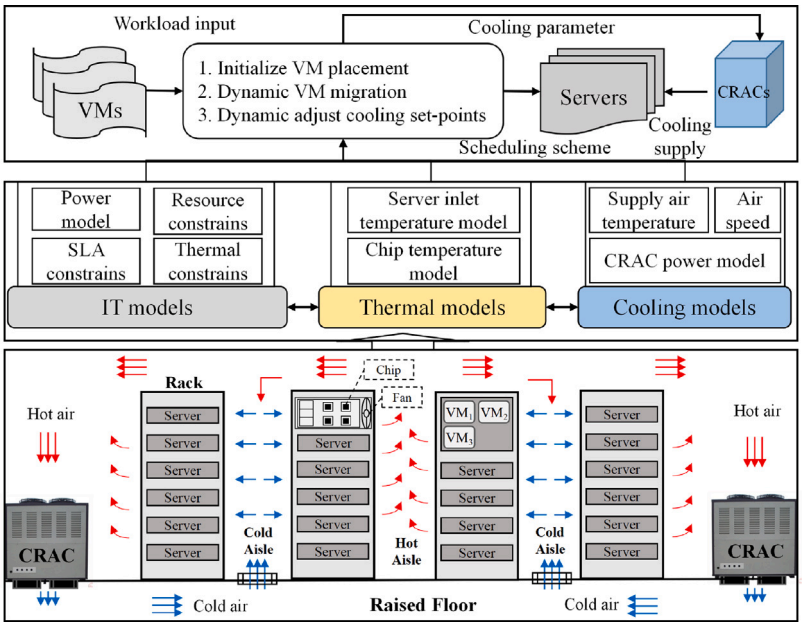

Fig. 1. System model.

scheme that minimizes the energy consumption of the server. Apart from the computing power, the work [15] also considers the link loss of VM placement and adopts the MOPSO algorithm to solve the multi-objective optimization VM placement problem. However, these works consolidate VMs to fewer hosts to reduce IT power, but high-loaded hosts are prone to localized hotspots that consume more cooling energy. Therefore, this work develops a holistic thermal model to guide VM placement and cooling control while reducing the overall energy consumption of computing and cooling facilities. Besides, a hybrid algorithm combining DE and PSO is designed to explore more energy-efficient VM placement schemes.

2.3. Machine learning algorithm

Some ML models are employed to predict system critical characteristics and make scheduling decisions. For example, Xiao et al. [24] adopted a Q-learning algorithm to determine the optimal host states based on available resources and cooling states. Subsequently, a load-balancing policy was executed for VM consolidation and migration. This control decoupling scheme performs well regarding energy consumption and latency metrics, but its host configuration only considers three power states, which is insufficient for fine-grained thermal management. Further, the work [25] employs a long short-term memory (LSTM) model to characterize server thermal behavior and power consumption. Subsequently, a deep Q network (DQN) model is trained to allocate workloads to minimize power consumption and temperature. Similarly, Zhou et al. [26] designed a parameterized action-space-based DQN algorithm that generates discrete and continuous actions to control the IT and cooling systems. These works construct a high-preserving digital twin system or environment to allow deep reinforcement learning (DRL) agents to explore and learn scheduling strategies. Reinforcement learning (RL) performs well in tackling online optimization problems with large-scale solution spaces [27]. However, the model's training overhead and trial-and-error mechanism may lead to unexpected SLAV and function failure.

3. System modeling and problem statement

As depicted in Fig. 1, the system model is divided into three layers. The infrastructure layer mainly consists of servers, racks, and computer room air conditioners (CRACs). Numerous heterogeneous servers form a cluster with powerful computing capabilities to provide cloud services to users via the Internet. CRACs are responsible for exhausting heat emitted from IT equipment and other heat sources to maintain a thermal environment with constant temperature and humidity. The model layer includes IT models, cooling models, and thermal models. Server power models characterize heterogeneous server states with different resource and power attributes. Secondly, SLAs, resource and thermal constraints are formulated to ensure the quality of service (QoS), performance, and security of the system for users. Chip and server inlet temperature models are constructed to represent the thermal profiles of the server room and hosts. The cooling power consumption depends on the supply temperature and air speed. The system management layer is responsible for scheduling submitted VM loads to the server and operating the cooling knobs to minimize the operational overhead of the system while ensuring efficiency, stability, and security. The notation involved in system modeling is shown in Table 1.

3.1. Thermal model

Thermal-aware IT scheduling focuses on chip temperature, which depends on the server's ambient temperature, chip power, and fan speed. Besides, the cooling system operates control knobs based on the temperature profile of the room. According to the thermal guidelines of ASHRAE [28], server inlet temperatures are often used to evaluate the data center's operating environment. The server inlet temperature is determined by the supply temperature and air velocity of the CRACs, and the power distribution of the heat source. The proposed CPU and server inlet temperature modeling is described in the following.

3.1.1. Server inlet temperature model

This work used the commercial simulation software 6SigmaRoom [29] to construct a CFD model of an air-cooled data center, as shown in Fig. 2. The parameter setting is shown in Table 2. Specifically, 20 racks are divided into four rows and placed back-to-back to form hot and cold aisles. Cold air enters from the front of the racks, absorbs the heat from the servers, and then generates hot air back to the CRACs. In addition, three thermal sensors at different heights are deployed on the front of each rack to collect server inlet temperature data.

A complex and dynamic thermal field is formed in the server room due to the heat sources heterogeneity, airflow organization,

Table 1
Definition of notations.

Notation	Definition	Notation	Definition
$\mathbb{C}, \mathbb{V}, \mathbb{S}$	Sets of CRACs, VMs and servers	t	Time interval
K, N, M	Numbers of CRACs, VMs, server	u	CPU utilization
$\mathbb{D}$	Temperature distribution	s_i, vm_i	*i-th server, vm*
$\mathbb{R}$	Resource type	$comp/fan$	CRAC compressor/fan
P	Power consumption (W)	sup	CRAC supply air
E	Energy consumption (kW·h)	$inlet$	Rack inlet
T	Temperature (°C)	w	Weight
Q_{CRAC}	Heat removed by CRACs (J)	$critical$	Thermal critical value
$S_{overload\_hot}$, S_{target}, S_{active}	List of overloaded, target and active servers	T_{sup}	Supply air temperature
$VM_{migrate}$	List of migrated VMs	UP_THR	Upper CPU usage threshold
BS	CRAC Blower speed	LW_THR	Lower CPU usage threshold

Table 2
Data center parameter settings.

Item		Value or Range
Rack	Number of racks	20
	Number of servers each rack	40
	Rack Power (P_{rack})	6 kW–12 kW
CRAC	Number of CRACs	4
	Blower speed (BS)	40%–100%
	Supply air temperature (T_{sup})	18 °C–27 °C
Server	Heat capacity of the CPU (C)	340 J/K
	Thermal resistance of the CPU (R)	0.34 K/W
	Initial temperature of the CPU	44.85 °C
	Temperature threshold of the CPU	75 °C
	Temperature threshold of the server inlet	30 °C

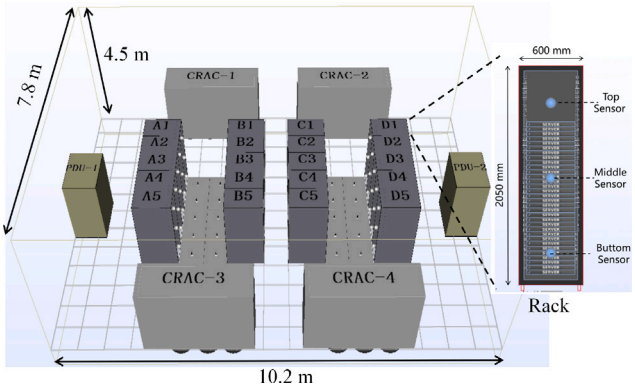

Fig. 2. Top view of the computer room CFD model and thermal sensor locations.

and facility layout. Traditional simplified physical models and CFD simulation-based modeling approaches suffer from the limitations of low prediction accuracy and high computational overhead, respectively. Therefore, to achieve an acceptable trade-off between accuracy and complexity, this work adopts a data-driven thermal modeling approach to predict the thermal distribution of a server room. The specific thermal modeling steps are as follows. A CFD model performs simulation calculations under various parameter configurations, and simulation records are collected to generate datasets. Subsequently, the dataset is used to train the data-driven thermal model and verify the prediction accuracy. Finally, the real-time temperature distribution is evaluated based on the trained thermal model. The artificial neural network (ANN)-based thermal modeling approach used in this work avoids the traditional sense of mechanical models. ANNs with good self-learning and fitting ability have apparent advantages in modeling nonlinear systems [30].

As shown in Fig. 3, the supply temperature vector $\boldsymbol{T_{sup}}$=[$T_{sup,1}$, …, $T_{sup,i}$] and the blower air speed vector $\boldsymbol{BS}$=[BS_1, …, BS_i] of the CRACs, and the rack power vector $\boldsymbol{P_{rack}}$=[$P_{rack,1}$, …, $P_{rack,i}$], are inputted into the ANN-based thermal prediction model. Subsequently, the thermal model outputs a temperature distribution that includes the

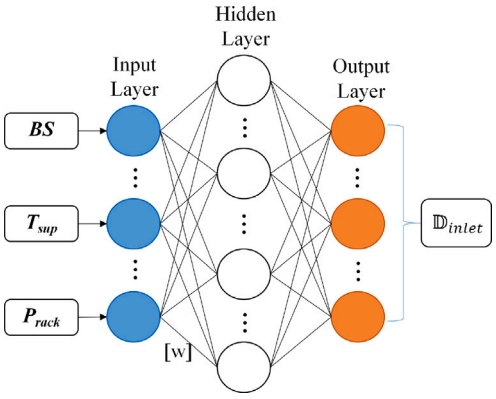

Fig. 3. Illustration of the ANN-based thermal model.

inlet temperatures at the top, middle, and bottom positions of each rack, expressed as,

$$\mathbb{D}_{\text{inlet}} = f\left(\boldsymbol{T}_{\text{sup}}, \boldsymbol{BS}, \boldsymbol{P}_{\text{rack}}\right). \tag{1}$$

The experiment uses the Latin hypercube sampling (LHS) method [31] to generate 1100 sets of random parameter combinations that uniformly cover the multi-dimensional parameter space. Subsequently, CFD numerical simulation calculations were carried out to generate 1100 sets of sample data, each containing 28 variables (blower speed $\boldsymbol{BS}$ and supply temperature $\boldsymbol{T_{sup}}$ for four CRACs and operating power $\boldsymbol{P_{rack}}$ for 20 racks) and 60 dependent variables (sensor temperature). The dataset is sliced into training and testing sets according to 10:1. Moreover, 10-fold cross-validation and Dropout methods are used to train the model to enhance the neural network's generalization ability. The adopted ANN model has three network layers with 28, 100, and 60 nodes, ReLU activation function, dropout rate of 0.5, batch size of 12, and learning rate of 0.01. The prediction results show that the root mean square error (RMSE) of the ANN-based thermal model is about 1.1 °C, the mean absolute percentage error (MAPE) is 3.1%, and the computation time is 3.5 ms, which can satisfy the prediction accuracy and efficiency demand.

3.1.2. CPU temperature model

The RC circuit model is commonly employed to construct a CPU thermal model owing to the inherent duality between the chip's thermal phenomena and the dynamic circuitry. Concretely, RC model establishes a quantitative relationship between the CPU temperature T_{cpu} and two key variables, power P_{cpu}, and server inlet temperature T_{inlet}, expressed as [32],

$$T_{cpu}(t+\Delta t) = T_{cpu}^{\infty}(t+\Delta t) + (T_{cpu}(t) - T_{cpu}^{\infty}(t+\Delta t)) \cdot e^{-\frac{\Delta t}{R\cdot(t+\Delta t)\cdot C}} \tag{2}$$

where Δt is a prediction time step. Besides, R, C denote the thermal resistance and specific heat capacity of the CPU, respectively. Therefore,

$T^{\infty}_{cpu}(t)$ is the steady state temperature denoted as,

$$T^{\infty}_{cpu}(t) = P_{cpu}(t) \cdot R(t) + T_{inlet}(t) \tag{3}$$

3.2. IT system

Assume that the data center has M servers forming a cluster denoted as $\sim = (s_1, s_2, \ldots, s_m)$. The cluster runs N VMs defined as $\mathbb{V} = (vm_1, vm_2, \ldots, vm_n)$. The resource type of each server is $\mathbb{R} = (CPU, RAM, BW, SR)$, denoting CPU, memory, bandwidth, and storage resources, respectively. Note that a VM is allowed to be placed on only one server. $x_{i,j}$ denote the mapping of VMs to servers, defined as,

$$x_{i,j} = \begin{cases} 1, & \text{if } vm_j \in VmList_i \\ 0, & \text{else} \end{cases}, \tag{4}$$

$$s.t. \sum_{i=1}^{M} x_{i,j} = 1 \tag{4a}$$

where $VmList_i$ is the VMs running on s_i. Besides, the resource utilization of the server depends on the amount of resource requests of the running VMs, expressed as,

$$u^{\mathbb{R}}_i = \frac{\sum \mathbb{R}^{req}_k}{\mathbb{R}^{cap}_i}, vm_k \in VmList_i, \tag{5}$$

where $\mathbb{R}^{req}_k$ and $\mathbb{R}^{cap}_i$ are the amount of resource requests for running VMs and the total resource capacity in s_i, respectively. Therefore, the VM placement must satisfy the following server resource constraints,

$$\sum \mathbb{R}^{req}_k \le \mathbb{R}^{cap}_i, \forall s_i \in \mathbb{S}, vm_k \in VmList_i, \tag{6}$$

This work uses a local linear-based server power model to evaluate server power consumption for different hardware configurations and operating states. The model assumes that server power correlates linearly with CPU utilization in each interval. Therefore, the CPU utilization is divided into p sub-intervals [0, 1/p), [1/p, 2/p), …, [(p-1)/p, 1], so express the power consumption P_j of server s_j by the CPU utilization as [33],

$$P_j\left(u^{cpu}_j\right) = \begin{cases} \alpha_1 \cdot u^{cpu}_j + \beta_1, & 0 \le u^{cpu}_j < 1/p \\ \alpha_2 \cdot u^{cpu}_j + \beta_2, 1/p \le u^{cpu}_j < 2/p \\ \alpha_p \cdot u^{cpu}_j + \beta_p, (p-1)/p \le u^{cpu}_j \le 1 \end{cases}, \tag{7}$$

where α_i, β_i (i = 1, 2, …, p) are constants that depend on the hardware configuration and power attributes of the server. Subsequently, we adopt the real power consumption data of the server provided by SPECpower [34] to fit the constant parameters. Therefore, from Eq. (7), the total energy consumption E_{IT} of the cluster can be estimated as,

$$E_{IT} = \sum_{j=1}^{N} \gamma_j \cdot P_j\left(u^{cpu}_j(t)\right), \tag{8}$$

where γ_j= 1, 0 denote the active and sleep modes of the server, respectively.

3.3. Cooling system

Assume that the cooling system consists of K CRACs, denoted as $\mathbb{C} = (CRAC_1, CRAC_2, \ldots, CRAC_k)$, where $CRAC_k$ is the kth CRAC. CRACs contribute 30%–40% of the energy consumption of the cooling system [5]. The power consumption of each CRAC mainly originates from the compressor and fan, so the power consumption of CRACk is defined as,

$$P^{CRAC}_k = P^{comp}_k + P^{fan}_k \tag{9}$$

where the fan power consumption P^{fan} shows a cubic exponential growth relationship with the speed fs [35], denoted as $P^{fan} \propto fs^3$. Besides, the power consumption of the compressor is defined as,

$$P^{comp} = \frac{Q_{CRAC}}{\text{COP}\left(T_{sup}\right)} \tag{10}$$

where Q_{CRAC} indicates the amount of heat removed from the server room by the cooling system, which is related to the total IT power P_{IT} [36]. A higher coefficient of performance (COP) indicates a more efficient cooling of the CRAC. Data measured by HP labs show that COP is positively correlated with the air supply temperature T_{sup} of the CRAC [11], expressed as,

$$\text{COP}\left(T_{sup}\right) = 0.0068 \cdot {T_{sup}}^2 + 0.0008 \cdot T_{sup} + 0.458. \tag{11}$$

Eq. (11) reveals that the higher the CRAC supply temperature T_{sup}, the higher the cooling efficiency. Moreover, the total energy consumption of the cooling system is estimated as,

$$E_{cooling} = \sum_{k=1}^{K} P^{CRAC}_k \tag{12}$$

3.4. Problem statement

The thermal-aware VM placement problem for DCs is formulated as a multi-constrained holistic energy minimization problem. Specifically, given cluster server states $\mathbb{S}$, air conditioning set-points $\mathbb{C}$, and VM workloads $\mathbb{V}$. The objective is to find an optimal VM placement scheme and cooling supply temperature to minimize the total energy consumption of IT and cooling facilities while satisfying resource and thermal constraints, denoted as,

Given : $\mathbb{S}, \mathbb{C}, \mathbb{V}$

Find : $\mathbb{V} \to \mathbb{S}, T_{sup}$ of $\mathbb{S}$

Minimizing : $E_{total} : E_{IT} + E_{cooling}$, (13)

$$\sum \mathbb{R}^{req}_{vms \text{ in } server_i} \le \mathbb{R}^{cap}_{server_i} i \in [1, \ldots, M], \tag{13a}$$

$$T_{inlet} \le T_{critical}. \tag{13b}$$

4. Swarm intelligence algorithm

4.1. Basic algorithm

Differential evolution is a swarm intelligence optimization algorithm that simulates the evolutionary pattern of heredity, mutation, and selection of biological populations in nature [37]. DE updates and evolves the population through mutation, crossover, and selection. Define the individuals of a population in the D-dimensional space as [38],

$$X^g_i = \left[x^g_{i,1}, x^g_{i,2}, \ldots, x^g_{i,D}\right], i = 1, 2, \ldots, NP, \tag{14}$$

where X^g_i = denotes the ith individual of the gth iteration. NP is the population size. The mutation operation randomly selects three different individuals X^g_{r1}, X^g_{r2}, and X^g_{r3} and generates the mutated individuals according to the DE/rand/bin strategy, denoted as,

$$M^g_i = X^g_{r1} + F\left(X^g_{r2} - X^g_{r3}\right), i \ne r1 \ne r2 \ne r3, \tag{15}$$

where F is the variation factor, which is used to control the vector difference. Besides, the crossover operation takes the parent individual X^g_i and the mutated individual M^g_i to produce a new individual according to Eq. (16).

$$C^g_{i,j} = \begin{cases} M^g_{i,j}, & \text{if } (rand \le CR) \quad \text{or } (rand(1, D) = j) \\ X^g_{i,j}, & \text{else} \end{cases}, \tag{16}$$

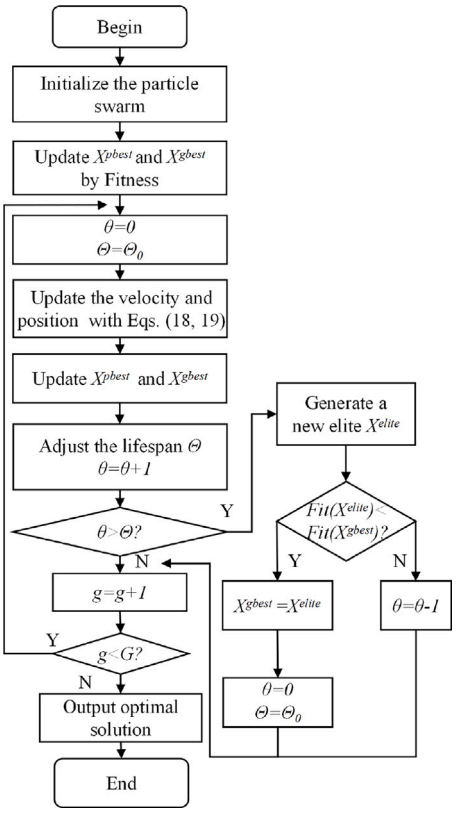

Fig. 4. Flowchart of proposed ERPSO algorithm.

where CR is the crossover factor and *rand* is a random number between 0 and 1. The selection operation uses a greedy strategy to decide whether the crossover individual replaces the parent as the following generation population individual based on fitness, denoted as,

$$X_i^{g+1} = \begin{cases} C_i^g, if\ \left(Fit\left(C_i^g\right) < Fit\left(X_i^g\right)\right) \\ X_i^g,\ \text{otherwise} \end{cases}, \tag{17}$$

where $Fit()$ is the fitness function related to the optimization objective.

Particle Swarm Optimization (PSO) is a swarm intelligence optimization algorithm that simulates the foraging behavior of the bird swarm [39]. PSO is widely adopted in combination optimization problems due to its simple principle, few dependent parameters, and fast convergence speed. The particles update their speed and position by individual and global optimal solutions during the evolutionary process, as shown in Eqs. (18), (19) [40].

$$V_i^{g+1} = wV_i^g + c_1r_1\left(X_i^{g,pbest} - X_i^g\right) + c_2r_2\left(X_i^{g,gbest} - X_i^g\right), \tag{18}$$

$$X_i^{g+1} = X_i^g + V_i^{g+1}, \tag{19}$$

where c_1, c_2 are learning factors and r_1, r_2 are random numbers uniformly distributed from 0 to 1. Also, w denotes the inertia weight that balances the exploitation and exploration capabilities of the algorithm. The encoding of particles uses a multi-dimensional vector to represent the mapping of VMs and servers. Specifically, the location number and value of the particle encoding are used as the VM and server number, respectively. For example, the particle X_i=[2, 5, 1, 4, 2] indicates that the vm_1 at location 1 is assigned to server s_2. Similarly, vm_2 is assigned to server s_5.

4.2. Proposed DE-ERPSO

Two modified methods are adopted to strengthen the global optimal search capability of PSO while maintaining the fast convergence property. More specifically, the mutation, crossover, and selection operations of the DE algorithm are employed to drastically perturb the worse-performing particles and increase the randomness to generate better solutions. Secondly, considering that populations are trapped in local optima at later stages of evolution, leading to evolutionary stagnation, a PSO with an elite re-selection mechanism is designed to provide new guidance for populations searching for optimal solutions [41]. As shown in Fig. 4, ERPSO introduces age θ and lifetime Θ variables for the global optimal particle X_i^{gbest} to simulate the life cycle. Specifically, the initial age and lifetime of the particle X_i^{gbest} are set to be θ= 0 and $\Theta = \Theta_0$, respectively. With each iteration, the age of X_i^{gbest} is added by 1, while the lifetime Θ needs to be dynamically adjusted according to the guidance capability. Eqs. (20) and (21) are designed to measure the guidance capability.

$$\delta_{X_i^{gbest}} = Fit\left(X_i^{g,gbest}\right) - Fit\left(X_i^{g-1,gbest}\right), \tag{20}$$

$$\sum_{i=1}^{M}\delta_{X_i^{pbest}} = \sum_{i=1}^{M}\left(Fit\left(X_i^{g,pbest}\right) - Fit\left(X_i^{g-1,pbest}\right)\right) \tag{21}$$

where $\delta_{X_i^{gbest}}$ and $\delta_{X_i^{pbest}}$ denote the global and local guidance capabilities of particle X_i, respectively. $Fit()$ is the fitness function that evaluates the solution, assuming the minimizing optimization objective. If $\delta_{X_i^{gbest}} < 0$ in Eq. (20) means that the current X_i^{gbest} has good leadership ability to guide the population to find a better solution. Therefore, the lifetime of X_i^{gbest} increases by 2. When $\delta_{X_i^{gbest}} = 0$ and $\sum_{i=1}^{M}\delta_{X_i^{pbest}} < 0$, it indicates that the optimal solution of the population is not improved, but the solution of at least one particle is enhanced, which means that the optimal particle still has the potential and possibility to enhance the population. Thus, the lifetime of X_i^{gbest} is only increased by 1. Once $\delta_{X_i^{gbest}} = 0$ and $\sum_{i=1}^{M}\delta_{X_i^{pbest}} = 0$, it implies that this optimal particle loses its leadership ability and cannot improve the quality of the population particles, so the lifetime remains unchanged. In summary, the lifetime Θ adjusting strategy of X_i^{gbest} is designed as,

$$\Theta = \begin{cases} \Theta + 2,\ \text{if}\ \delta_{X_i^{gbest}} < 0 \\ \Theta + 1,\ \text{if}\ \delta_{X_i^{gbest}} = 0\ \text{and}\ \sum_{i=1}^{M}\delta_{X_i^{pbest}} < 0 \\ \Theta,\ \text{if}\ \delta_{X_i^{gbest}} = 0\ \text{and}\ \sum_{i=1}^{M}\delta_{X_i^{pbest}} = 0 \end{cases}. \tag{22}$$

Moreover, once the age of X_i^{gbest} exceeds the lifespan, i.e., $\theta > \Theta$, implying that the evolution of the population has stagnated. In this case, a new elite particle X_i^{elite} needs to be generated to try to replace the old optimal particle, as shown in Eq. (23).

$$X_{i,j}^{elite} = \begin{cases} \text{random}\left(L_j, U_j\right), & \text{if (randj} < pro) \\ X_{i,j}^{gbest}, & \text{else} \end{cases} \tag{23}$$

A random number between 0 and 1 is generated to compare with *pro* to determine whether the value of the jth dimension of $X_{i,j}^{elite}$ is randomly generated within the constraint range $[L_j, U_j]$ or inherited from . Note that the parameter pro is usually set to a small value to preserve the structure of the old global optimal particle. Subsequently, the fitness values of $X_{i,j}^{elite}$ and $X_{i,j}^{gbest}$ are compared. If $Fit(X_{i,j}^{elite}) < Fit(X_{i,j}^{gbest})$, the new elite particle $X_{i,j}^{elite}$ will replace the old optimal particle and become the new leader of the population. Conversely, the old optimal particle $X_{i,j}^{gbest}$ will be kept and the age will be reduced by 1.

The proposed DE-ERPSO is shown in Algorithm 1. Initialize the velocity and position of the swarm particles in Line 1. Subsequently, G iterations are started (Lines 2–17). The fitness of each particle is calculated using Eq. (26) and sorted in ascending order (Lines 3–4). The sorted particle population was divided into $BetterSwarm$ and $WorseSwarm$ according to P%. Mutation (Eq. (15)), crossover (Eq. (16)), and selection (Eq. (17)) operations are performed on $WorseSwarm$. The fitness of the newly generated individuals is calculated to update X^{ibest} and X^{gbest}. A greedy strategy is used to select the

better individuals to form $newWorseSwarm$ (Lines 6–9). Besides, the velocity and position of the particles in the $BetterSwarm$ are updated according to Eqs. (18) and (19) and the latest X^{ibest} and X^{gbest} are determined. Then, the age of the X^{gbest} is adjusted according to the evaluated values of Eqs. (20) and (21). If the age exceeds the lifespan, a re-select elite operation is performed. A greedy strategy is used to determine the new leader of the population and individuals to form $newBetterSwarm$. Finally, $newWorseSwarm$ and $newBetterSwarm$ are aggregated into a new swarm $newSwarm$. Loop all the above steps until the maximum number of iterations or the end condition.

5. Energy-aware VM placement method

As Fig. 5 shows, VM placement in IT systems is divided into five major steps, including overloaded and hot server detection, migrated VM selection, target server selection, VM placement, and underloaded server processing. The CPU thermal model evaluates the CPU temperature for overheated server detection and VM placement. Additionally, the server inlet thermal model is employed to predict the server inlet temperature distribution and to guide the supply air temperature control of the CRAC to satisfy the thermal constraint. The details of each step are described below.

Algorithm 1: DE-ERPSO-based VM placement

input : S_{target}, $VM_{migarte}$
output: $MigrationMap$: $VM_{migarte} \rightarrow S_{target}$;

1 Initialize the $Swarm$;
2 **for** $g \leftarrow 1$ **to** G **do**
3 Use Eq. (26) to evaluate Fit(X_i^g) of X_i^g;
4 Sort $Swarm$ by Fit(X_i^g) in ascending order;
5 $BetterSwarm, WorseSwarm \leftarrow$ Split $Swarm$ according to P%;
6 $M_g \leftarrow$Mutation($WorseSwarm$);
7 $C_g \leftarrow$Crossover(M^g);
8 Update X^{ibest} and X^{gbest} by M^g and C^g;
9 $newWorseSwarm \leftarrow$ GreedySelect($WorseSwarm$, M^g, C^g);
10 $V^g \leftarrow$VelocityUpdate($BetterSwarm$);
11 $X^{g+1} \leftarrow$PositionUpdate(V^g);
12 Update X^{ibest} and X^{gbest} by X^{g+1};
13 **if** X^{gbest} *has* $\theta > \Theta$ **then**
14 $E^g \leftarrow$EliteReelection(X^{gbest});
15 **end**
16 $newBetterSwarm \leftarrow$GreedySelection($E^g$, X^{g+1});
17 $newSwarm \leftarrow$Aggregate($newBetterSwarm$, $newWorseSwarm$)
18 **end**
19 Construct and return the $MigrationMap$;

5.1. Overloaded server detection

VM consolidation and migration in clusters may lead to performance degradation and thermal risks for a part of high-load servers. Therefore, this work adopts a utilization-temperature-based overloaded server detection strategy. We analyze the energy efficiency curves of multiple types of servers in SPECpower [34], from which we find that most servers have a specific CPU utilization $u^{cpu}_{opt\_PPR}$ with optimal performance-to-power ratio (PPR). The optimal utilization $u^{cpu}_{opt\_PPR}$ of most servers is distributed in [0.6, 0.9], with a few being fully loaded. In other words, the server energy efficiency ratio increases at [0, $u^{cpu}_{opt\_PPR}$] and decreases at [$u^{cpu}_{opt\_PPR}$, 1.0]. Therefore, to ensure that the server is at a high energy efficiency level while avoiding overloading, $u^{cpu}_{opt\_PPR}$ is set to the trigger threshold UP_THR for overload detection. In addition, a CPU critical temperature threshold $T^{cpu}_{critical}$ is given to prevent overheating the server. Thus, if a server s_i has a CPU utilization $u_i^{cpu} > UP\_THR$ or a temperature $T_i^{cpu} > T^{cpu}_{critical}$, the server will be added to the list of overloaded and overheated servers $S_{overload\_hot}$.

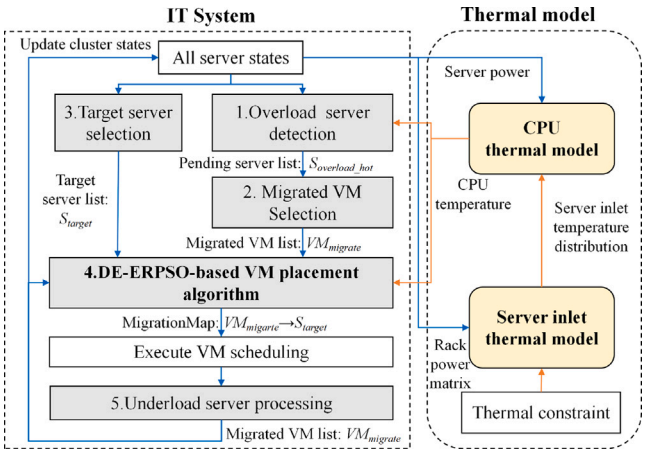

Fig. 5. Schematic diagram of energy-aware VM placement method.

5.2. Migrated VM selection

Considering that VM migration leads to performance degradation and SLAV, this work adopts the migrated VM selection strategy based on the migration value ratio (MVR) proposed in the work [7] to reduce the number of VM migrations. MVR measures the overhead of VM migration by considering three critical resources, namely, CPU, memory, and bandwidth, defined as,

$$MVR_k = |(U_i^{CPU} - UP\_THR_i) \cdot CPU_i^{\max} - CPU_k^{req}| \cdot RAM_k^{req} / BW_i^{avail}, \forall s_i \in \mathbb{S}, vm_k \in \text{VmList}_i, \tag{24}$$

where u_i^{cpu}, UP_THR_i denote the CPU utilization and overload threshold of server s_i, respectively. $CPU_i^{\max}, BW_i^{\text{avail}}$ denote the total CPU resources and available bandwidth of server s_i, respectively; CPU_k^{req} and RAM_k^{req} indicate the amount of CPU and memory resources requested by vm_k, respectively. For each overloaded host, the VM with the smallest MVR on the overloaded host is selected to be added to the migrated VM list $VM_{migrate}$ for each operation until $u_i^{cpu} < UP\_THR_i$.

5.3. Target server selection

After determining the migrated VMs, $VM_{migrate}$, some active servers that satisfy the constraints are selected as target servers, S_{target}.

$$\forall s_i \in S_{active}, \forall s_i \notin S_{overload\_hot}, \exists vm_k \in VM_{migrate}, \tag{25}$$

Eq. (25) indicates that the target server is not overloaded, while its remaining resources need to satisfy the resource requests of at least one migrated VM.

5.4. DE-ERPSO-based VM placement algorithm

Given a migrated VM list, $VM_{migrate}$, and a target server list, S_{target}, the proposed DE-ERPSO-based VM placement algorithm is employed to generate a new VM and server mapping. The optimization objective of VM placement is to minimize the overall energy consumption of the IT and cooling systems. To this regard, the fitness function evaluates the VM placement solution using a weighted metric considering the power consumption and temperature of the servers, designed as,

$$Fit(X_i) = \sum_i^D \sqrt{(\frac{P_i}{P_{max\_power}})^2 + (\frac{T^i_{cpu}}{T_{cpu\_max}})^2}, \tag{26}$$

$$P_{max\_power} = Max(P_i^{peak}, \ldots, P_D^{peak}), \tag{26a}$$

$$T_{cpu\_\max} = Max(T^i_{cpu}, \ldots, T^D_{cpu}). \tag{26b}$$

Table 3
Server and VM instances.

Type	Model	Cores	Mips/coreRAM (GB)	BW (Gb/s)	Storage (GB)	$u^{cpu}_{opt\_PPR}$
Server	HP ProLiant ML110 G3	2	300016	100	1000	1.0
	Dell PowerEdge R240	6	370032	100	1000	0.6
	ProLiant DL20	8	320016	100	1000	0.7
	Acer AR380 F2	12	250032	100	1000	1.0
VM	a1.m	1	23002	10	3	
	a1.l	2	23004	10	3	
	a1.xl	4	23008	10	3	
	g4dn.2xl	8	250016	10	3	

where $P_i, P_i^{peak}, T^i_{cpu}$ denote the real-time power consumption, maximum power consumption, and CPU temperature of s_i, respectively. Besides, P_{max_power} and T_{cpu_max} represent the maximum peak power consumption and maximum temperature of all active servers in the cluster, respectively. Therefore, smaller fitness values indicate that the solution is closer to the optimization objective.

5.5. Low-loaded server processing

With the termination of the life cycle and relocation of some VMs, the cluster will be left with underutilized servers that consume unnecessary energy. Low-load or idle servers consume about 50% of the peak power [42]. Hence, to minimize the number of active servers to reduce energy consumption, a dynamic lower threshold (LW_THR) is used to detect low-loaded servers. Specifically, the average CPU utilization u_i^{aver} of each active server in the last l time steps is calculated. Subsequently, a statistical method is used to calculate the quartiles of u^{aver} for all active servers, and the lower threshold LW_THR is set to the lower quartile. Finally, all VMs in servers with $u_i^{aver} < LW\_THR$ are added to $VM_{migrate}$, relocated to the target server, and shut down the source server.

6. Simulation experiment and results

6.1. Experimental configuration

In this work, simulation experiments are conducted to evaluate the performance and effectiveness of the proposed TEVP. We extend the cloud simulation tool Cloudsim V4.0 [43] with a server thermal model, a server inlet thermal model, and a cooling power model. All simulation experiments and code are run on a laptop computer configured with Intel (R) Core (TM) i7-12700H, 2.30 GHz, and 16 GB of RAM. Additionally, the experiments use four VM instances from Amazon EC2 [44] service and four servers from SPECpower [34] to simulate a heterogeneous cluster. The VM instances and server configurations are shown in Table 3. We select VM load traces from two real-world workload datasets, PlanetLab [45] and Azure [46], to simulate the cluster load. The PlanetLab dataset collects load traces from 1353 network nodes for 10 days from March to mid-April 2011 at 5-minute intervals. The Azure public dataset covers a representative subset of more than 2.6 million VM workloads from the 2019 Azure service region. Five VM sets were constructed by extracting 800, 1000, 1200, 1400, and 1600 VM load traces from this dataset.

6.2. Experimental schemes and evaluation metrics

6.2.1. Baselines

This work selects six advanced heuristic and swarm intelligence-based VM placement methods for baseline. **FFD**: First Fit Decreasing is a heuristic algorithm [47] that sorts servers in descending order according to CPU utilization and subsequently places VMs on the first server that matches their resource requirements. **PEAP**: A heuristic-based VM placement method [7], which places VMs on the server whose amount of remaining CPU resources best matches its resource requirements. If the VM placement fails due to insufficient RAM and Bandwidth resources, the VM is preferentially placed on the server with the highest peak energy efficiency ratio. **HGAPSO**: A VM placement method based on a swarm intelligence algorithm with hybrid GA and PSO [8] to minimize cluster energy consumption and maximize resource utilization. **GRANITE-MMT**: Li et al. [11] proposed a thermal-aware VM placement and migration scheme. The scheme always selects the VM placement policy with minimum server and CRAC power consumption and uses a minimum migration time (MMT) algorithm to determine migrated VMs in hotspot servers. **ETAS-MMT**: Ilager et al. [12] proposed an energy and thermal-aware VM consolidation approach to reduce overall energy consumption while proactively preventing thermal risks. The approach employs a meta-heuristic VM allocation method and an MMT-based VM selection algorithm. **PTACO-MM**: Chen et al. [13] developed a power- and thermal-aware VM placement scheme to reduce the total energy consumption of IT and cooling systems. The scheme employs a VM selection algorithm based on the minimization algorithm (MM) and a VM placement algorithm based on improved ant colony optimization (ACO).

6.2.2. Evaluation metrics

Referring to the evaluation metrics of related works [11–13], the experiments use five metrics such as overall energy consumption (IT and cooling energy), SLAV, VM migrations, number of active hosts and thermal critical violations to evaluate the performance of the proposed TEVP. SLAV indicates that if a VM is allocated fewer CPU resources than its requested resources in a time slice, it is determined that the VM has a service violation in the current time slice. SLAV adopts the definition in CloudSim 4.0 as,

$$SLAV = \frac{\sum_{k=1}^{T}\sum_{j=1}^{S_k}\left(\left(CPU_{k,j}^{Req} - CPU_{k,j}^{Allo}\right)\cdot t\cdot S_k\right)}{\sum_{k=1}^{T}\sum_{j=1}^{S_k}\left(CPU_{k,j}^{Req}\cdot t\cdot S_k\right)}, \tag{27}$$

where T denotes the number of time slices for the whole simulation, and S_k indicates the number of overloaded VMs in the kth time slice. $CPU_{k,j}^{Req}, CPU_{k,j}^{Allo}$ denote the amount of CPU resources requested and actually allocated to VM_j in time slice k, respectively. $t \cdot S_k$ represents the time slice length between two CloudSim events. In addition, considering that VM migration leads to performance degradation, the simulation experiments set the performance degradation rate to 10%, which indicates that each VM migration only allocates 90% of the requested CPU and RAM resources. Therefore, the higher the VM migrations, the higher the SLAV.

6.3. Experimental results and analysis

6.3.1. Experiments on the PlanetLab dataset

The results in Fig. 6 show that the total energy consumption of the proposed TEVP outperforms the other baselines under different VM loads, with an average of 29.2% lower. Specifically, compared to the worst benchmark (FFD) and the best benchmark (PEAP), the 10-day total energy consumption of the proposed TEVP is 45.5% and 5.6% lower, respectively. Besides, it can be seen from Fig. 7 that the proposed TEVP saves 5.7% to 45.7% of IT energy consumption and 6.8% to 45.3% of cooling energy consumption over ten days compared to the benchmarks. This illustrates that TEVP uses an efficient VM placement

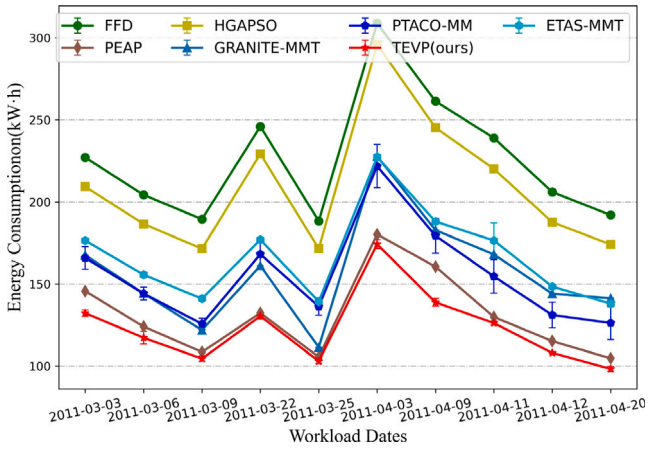

Fig. 6. Total energy consumption for scheduling schemes on different workload dates.

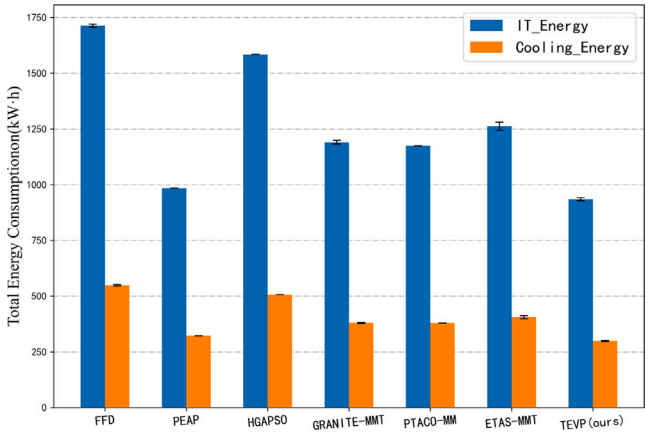

Fig. 7. Total IT and cooling energy consumption for all workload dates.

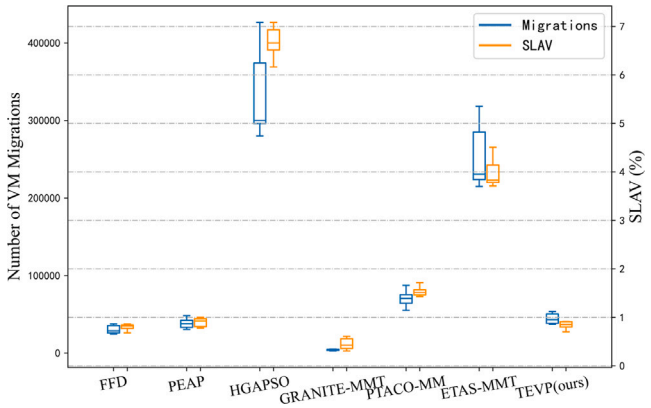

Fig. 8. Distribution of VM migrations and SLAV.

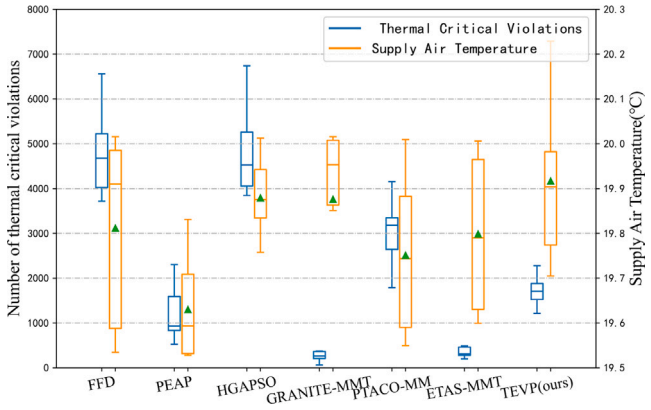

Fig. 9. Thermal critical violations and supply air temperature.

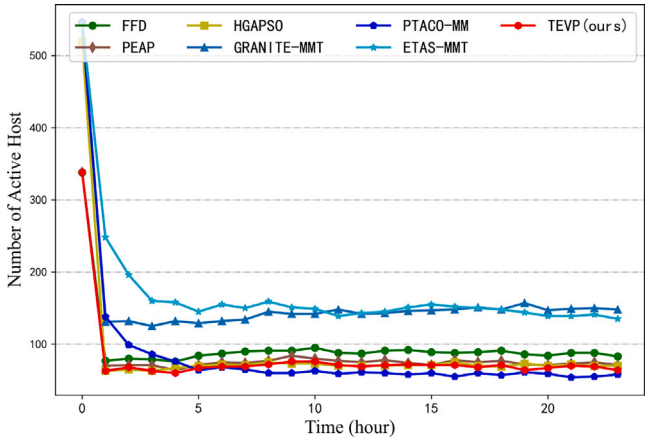

Fig. 10. Number of active hosts of different schemes on 2022-04-03.

approach to reduce the power consumption of IT equipment while also reducing the cooling demand under satisfying thermal constraints.

From the distribution of VM migrations and SLAV demonstrated in Fig. 8, it can be inferred that the higher the number of VM migrations, the more severe the performance degradation, which ultimately leads to high SLAV. Since the HGAPSO scheme reschedules all VMs at each scheduling operation to find a better VM placement scheme, it leads to frequent VM migration and high SLAV. However, the VM migration and SLAV of the proposed TEVP are kept at a low level, mainly due to the fact that setting the overload threshold based on the energy-efficiency characteristics of the servers can leverage the load capacity of the energy-efficient servers to run more VMs, which effectively reduces the VM migration. Moreover, the MVR-based VM selection strategy considers three key resources (CPU, memory, and bandwidth) to more accurately evaluate the overhead of VM migration, thus mitigating performance degradation. Fig. 9 shows that TEVP raises the cold air supply temperature, resulting in a slightly higher TCV than PEAP, GRANITE-MMT, and ETAS-MMT, but also within the low risk range.

The curve of variation in the number of active hosts in Fig. 10 shows that most VM placement methods consolidate VMs to fewer active hosts and shut down idle hosts to save IT energy. However, over-consolidation leads to some hosts being overloaded and overheated. Combined with the TCV distribution in Fig. 9, it can be inferred that GRANITE-MMT and ETAS-MMT start more hosts to maintain a low TCV, which also results in higher overall energy consumption. Note that PTACO-MM achieves the minimum number of active hosts, but the active hosts are overloaded and overheated due to excessive VM consolidation. Note that PTACO-MM achieves the minimum number of active hosts, but the active hosts are overloaded and overheated due to excessive VM consolidation. Moreover, overheated hosts force the cooling system to lower the supply air temperature, increasing cooling energy consumption. Overall, the proposed TEVP significantly outperforms the baseline regarding energy consumption while maintaining lower SLAV and TCV, thus achieving a trade-off between energy consumption, SLA, and thermal risk.

As seen from the thermal profile in Fig. 11, the higher the position, the higher the rack inlet temperature. The main reason is that the top of the rack inlet is more significantly affected by the hot air return

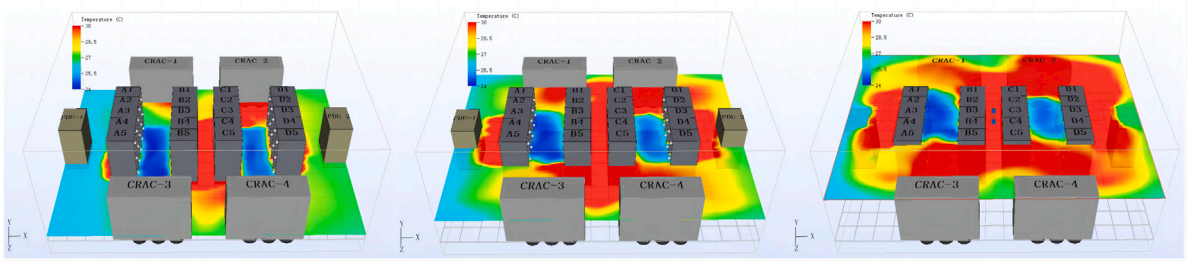

Fig. 11. Thermal profiles of bottom, middle and upper rack.

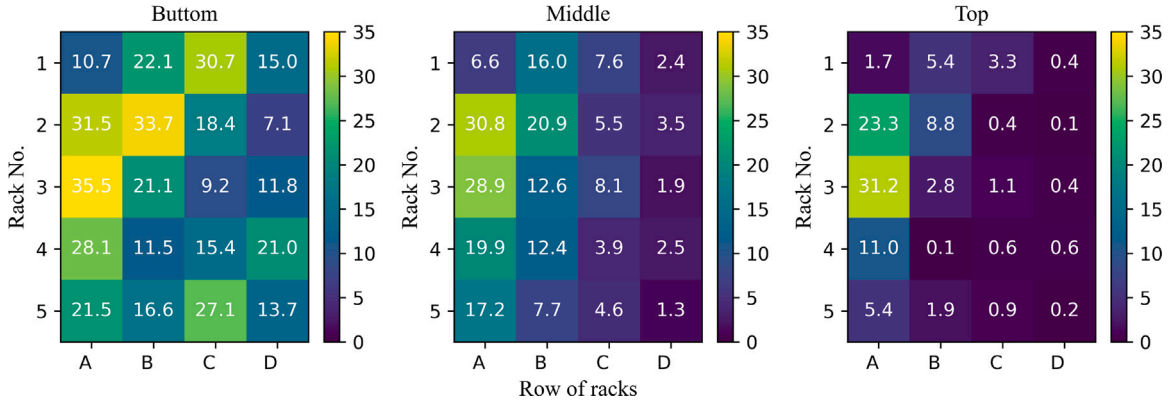

Fig. 12. Distribution of average CPU utilization for servers in the bottom, middle, and top of the rack.

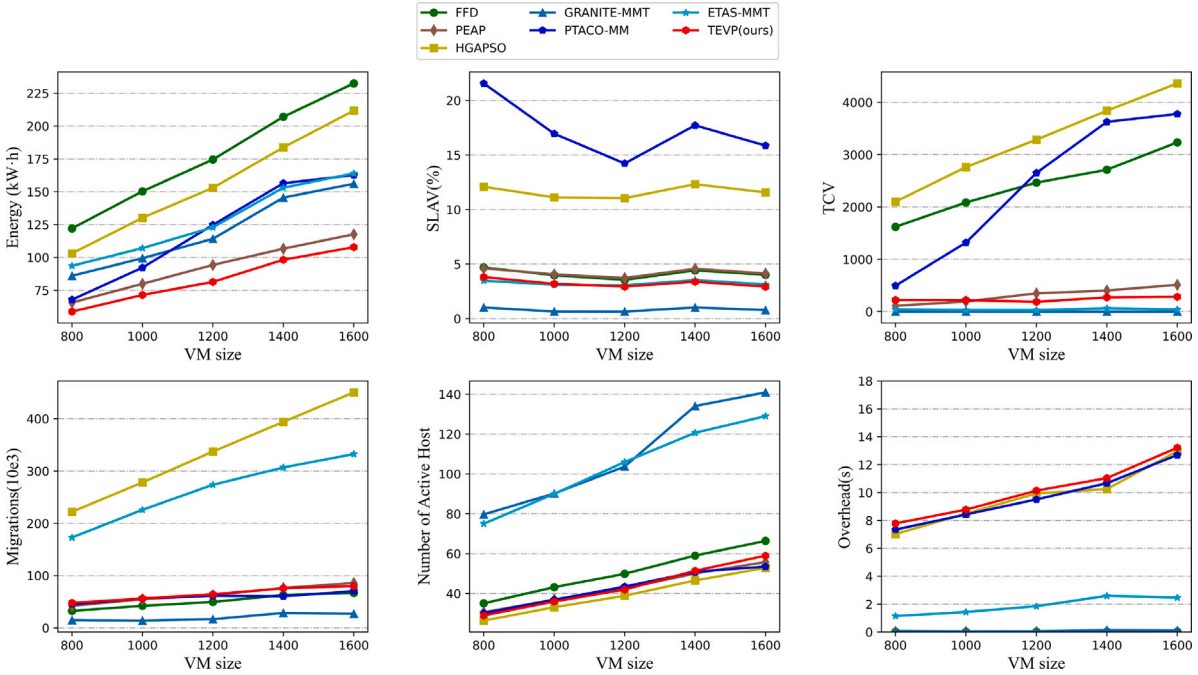

Fig. 13. Comparison of six performance metrics under five VM-scale datasets.

from the server outlet. The mixing of hot and cold air leads to higher inlet temperatures, especially more apparent at the ends of the rack rows. Besides, the cold aisle temperature between Row_A and Row_B is significantly lower than that of Row_C and Row_D due to the impact of the airflow organization and building layout of the server room.

Fig. 12 shows that the higher the server placement, the lower the average CPU utilization. Specifically, the average load of servers at the rack's bottom, middle, and top is 20.1%, 10.7%, and 5.1%, respectively. The average load at the bottom is 15.0% higher than that at the top. Also, the average load of servers in Row_A and Row_B is 9.6% higher than Row_C and Row_D. It can be inferred that the proposed thermal-aware VM placement policy prefers to place VMs on servers with lower inlet temperatures. The strategy improves heat transfer efficiency by increasing the temperature difference between the cold air and the surface of the IT equipment, which reduces the servers' temperature gradient and the cooling supply.

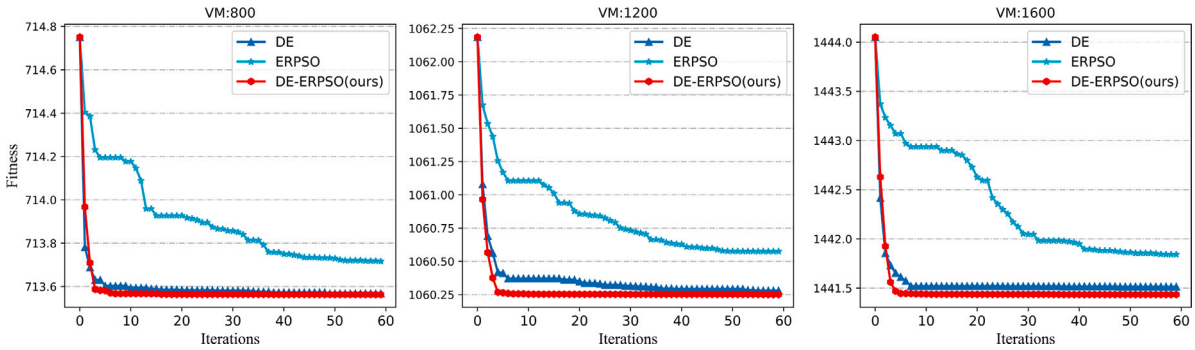

Fig. 14. Fitness curves of the three algorithms under different VM-scale.

6.3.2. Experiments on the Azure dataset

The following simulation experiments compare six performance metrics of the various schemes under five VM-scale datasets. Fig. 13 shows that as the cluster VM size increases, the energy consumption and active hosts increase significantly for all scheduling schemes. The proposed TEVP consumes less energy than the other baselines at different VM sizes, while SLAV, TCV, and VM migration are kept low. GRANITE-MMT and ETAS-MMT outperform in SLAV and TCV, but since these two schemes emphasize the thermal risk of hosts, the number of active hosts is higher and consumes more energy than other baselines. More specifically, as the VM size increases from 800 to 1600, the energy consumption of TEVP increases by 83.5%, while PTACO increases by a whopping 139%. Besides, compared to the average energy consumption of the benchmark, the energy saving ratio of TEVP increases from 34.6% to 38.1%. This indicates that the proposed DE-ERPSO is superior and energy-efficient for solving large-scale problems. However, the computational overhead of the proposed method is more than the other baselines since the swarm intelligence algorithms (DE-ERPSO, HGAPSO, PTACO) need to perform multiple iterations and operations to explore better solutions. For this issue, we use the multi-threading technique to reduce the algorithm time overhead and ensure it is within the acceptable scheduling time of realistic production environments.

6.3.3. Convergence of the algorithm

To compare the convergence ability of DE, ERPSO, and DE-ERPSO algorithms, we collected the fitness values for the first scheduling of the three algorithms in the Azure dataset at the scale of VMs 800, 1200, and 1600, respectively. As shown in Fig. 14, DE-ERPSO always finds the scheduling solution with the minimum fitness value after many iterations. This suggests that DE-ERPSO can find a more efficient VM placement scheme in large-scale clusters than DE and ERPSO. The reason is that DE-ERPSO not only has the superiority of fast convergence of PSO algorithm but also adopts the population diversity of DE to avoid PSO precocity and falling into local optimum dilemma. Meanwhile, the elite re-selection mechanism somewhat enhances the algorithm's optimal solution-searching ability.

6.3.4. Time complexity analysis

The time overhead of the proposed TEVP originates from the VM placement algorithm, which consists of three parts. (1) The overhead of overloaded host detection and migrated VM selection originates from the MVR-based sorting operation for X VMs on each server. Therefore, the worst-case sorting time complexity is $O(X^2)$. (2) DE-ERPSO occupies the primary time overhead of the VM relocation phase. Assume that the maximum iteration of DE-ERPSO is G, the number of migrated VMs and target servers are N and M, respectively, and the number of population particles is S. Therefore, the time complexity is $O(G \cdot S \cdot N \cdot M)$. (3) The low-loaded server processing overhead is derived from calculating and ranking the CPU utilization averages over l time steps for M servers, so the worst-case time complexity is $O(l \cdot M^2)$. Overall, the time complexity of the proposed DE-ERPSO algorithm is close to PSO and DE algorithms. Additionally, the time complexity of the two sorting operations (1 and 3) is much less than $O(G \cdot S \cdot N \cdot M)$ is negligible. Meanwhile, Fig. 13 shows that the computation overhead of DE-ERPSO and other swarm intelligence algorithms is close to 10s, which satisfies the practical cloud scenario with a 5-minute scheduling interval.

7. Conclusion

This work proposes a thermal and energy-aware VM placement method to minimize the holistic energy consumption of DCs while guaranteeing SLA and thermal constraints. Firstly, we construct a data-driven-based server inlet thermal model and an RC-based CPU thermal model to represent the dynamic, non-uniform thermodynamic environment for guiding the active thermal management of DCs. Second, this work designs a VM placement method, including a utilization-temperature-based overload detection strategy, an MVR-based migrated VM selection strategy, a DE-ERPSO-based VM placement algorithm, and a low-loaded server processing strategy, to achieve energy-efficient VM migration and placement. Extensive results show that the proposed TEVP effectively saves over 5.6% of total energy consumption while maintaining acceptable SLAV and TCV.

However, the DE-ERPSO-based VM placement algorithm can explore a better VM placement scheme, but the computational overhead is high. Therefore, further simplifying the hybrid algorithm operation and adopting multi-threading techniques is necessary to reduce the computational overhead. Additionally, considering that load volatility and cooling time delay can lead to cluster overloading and thermal risk problems, future research will construct load prediction models to guide VM placement and active cooling control.

CRediT authorship contribution statement

Jianpeng Lin: Writing – original draft, Methodology. **Weiwei Lin:** Writing – review & editing, Methodology, Funding acquisition. **Wentai Wu:** Writing – review & editing, Methodology. **Wenjun Lin:** Writing – review & editing, Investigation. **Keqin Li:** Writing – review & editing, Supervision.

Declaration of competing 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.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (62072187), The Major Key Project of PCL, China (PCL2023A09), Guangdong Major Project of Basic and Applied Basic Research, China (2019B030302002), and Guangzhou Development Zone Science and Technology, China (2021GH10).

Appendix

Abbreviation	Full name
DCs	Data centers
VM	Virtual machine
IT	Information technology
CRAC	Computer room air conditioner
ML	Machine learning
ANN	Artificial neural network
LSTM	Long short-term memory
CFD	Computational fluid dynamics
DVFS	Dynamic voltage frequency scaling
DRL	Deep reinforcement learning
RL	Reinforcement learning
DQN	Deep Q network
LHS	Latin hypercube sampling
SLA	Service level agreement
QoS	Quality of service
SLAV	Service level agreement violation
TCV	Thermal critical violation
MAPE	Mean absolute percentage error
RMSE	Root mean square error
RC	Resistor-Capacitance
DTM	Dynamic thermal management
XGBoost	eXtreme gradient boosting
DE	Differential evolution
PSO	Particle swarm optimization

References

[1] E. Masanet, A. Shehabi, N. Lei, Recalibrating global data center energy-use estimates, Science 367 (6481) (2020) 984–986.
[2] Z. Cao, X. Zhou, H. Hu, Z. Wang, Y. Wen, Toward a systematic survey for carbon neutral data centers, IEEE Commun. Surv. Tutor. 24 (2) (2022) 895–936.
[3] B. Nogrady, China launches world's largest carbon market: but is it ambitious enough? Nature 595 (869) (2021) 637.
[4] Climate Neutral Data Centre Pact, [Online]. Available: https://www.climateneutraldatacentre.net/. (Accessed 23 October 2023).
[5] Q. Zhang, Z. Meng, X. Hong, A survey on data center cooling systems: Technology, power consumption modeling and control strategy optimization, J. Syst. Archit. 119 (2021) 102253.
[6] W. Lin, F. Shi, W. Wu, K. Li, G. Wu, A taxonomy and survey of power models and power modeling for cloud servers, ACM Comput. Surv. 53 (2020) 1–41.
[7] W. Lin, W. Wu, L. He, An on-line virtual machine consolidation strategy for dual improvement in performance and energy conservation of server clusters in cloud data centers, IEEE Trans. Serv. Comput. 15 (2) (2019) 766–777.
[8] Neeraj Kumar Sharma, G. Ram Mohana Reddy, Multi-objective energy efficient virtual machines allocation at the cloud data center, IEEE Trans. Serv. Comput. 12 (2019) 158–171.
[9] Jianpeng Lin, Lin Weiwei, Thermal prediction for air-cooled data center using data driven-based model, Appl. Therm. Eng. 217 (2022) 119207.
[10] J. Athavale, M. Yoda, Y. Joshi, Comparison of data driven modeling approaches for temperature prediction in data centers, Int. J. Heat Mass Transfer 135 (2019) 1039–1052.
[11] Xiang Li, Peter Garraghan, Xiaohong Jiang, Zhaohui Wu, Jie Xu, Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy, IEEE Trans. Parallel Distrib. Syst. 29 (2018) 1317–1331.
[12] Shashikant Ilager, Kotagiri Ramamohanarao, Rajkumar Buyya, ETAS: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation, Concurr. Comput.: Pract. Exper. 31 (2019).
[13] R. Chen, Bo Liu, Weiwei Lin, Jianpeng Lin, Hui-Hui Cheng, Keqin Li, Power and thermal-aware virtual machine scheduling optimization in cloud data center, Future Gener. Comput. Syst. 145 (2023) 578–589.
[14] S. Wang, Z. Liu, Z. Zheng, Q. Sun, F. Yang, Particle swarm optimization for energy-aware virtual machine placement optimization in virtualized data centers, in: Proceedings of 2013 International Conference on Parallel and Distributed Systems, 2013, pp. 102–109.
[15] A.K. Singh, S.R. Swain, C.N. Lee, A metaheuristic virtual machine placement framework toward power efficiency of sustainable cloud environment, Soft Comput. 27 (2023) 3817–3828.
[16] Q. Tang, S.K.S. Gupta, G. Varsamopoulos, Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers:a cyber–physical approach, IEEE Trans. Parallel Distrib. Syst. 19 (11) (2008) 1458–1472.
[17] Lee, Eun Kyung, Hariharasudhan Viswanathan, Dario Pompili, Proactive thermal-aware resource management in virtualized HPC cloud datacenters, IEEE Trans. Cloud Comput. 5 (2017) 234–248.
[18] Shashikant Ilager, Kotagiri Ramamohanarao, Rajkumar Buyya, Thermal prediction for efficient energy management of clouds using machine learning, IEEE Trans. Parallel Distrib. Syst. 32 (2020) 1044–1056.
[19] Young Geun Kim, Seon Young Kim, Seung Hun Choi, Sung Woo Chung, Thermal-aware adaptive VM allocation considering server locations in heterogeneous data centers, J. Syst. Archit. 117 (2021) 102071.
[20] Jie Li, Yuhui Deng, Yi Zhou, Zhen Zhang, Geyong Min, Xiaodan Qin, Towards thermal-aware workload distribution in cloud data centers based on failure models, IEEE Trans. Comput. 72 (2023) 586–599.
[21] Alireza Aghasi, Kamal jamshidi and ali bohlooli. a thermal-aware energy-efficient virtual machine placement algorithm based on fuzzy controlled binary gravitational search algorithm (FC-BGSA), Cluster Comput. (2022) 1–19.
[22] Hao Feng, Yuhui Deng, Jie Li, A global-energy-aware virtual machine placement strategy for cloud data centers, J. Syst. Archit. 116 (2021) 102048.
[23] Bo Liu, Rui-Zhong Chen, Weiwei Lin, Wentai Wu, Jianpeng Lin, Keqin Li, Thermal-aware virtual machine placement based on multi-objective optimization, J. Supercomput. (2023) 1–28.
[24] P. Xiao, Z. Ni, D. Liu, A power and thermal-aware virtual machine management framework based on machine learning, Clust. Comput. 24 (2021) 2231–2248.
[25] D. Yi, X. Zhou, Y. Wen, R. Tan, Efficient compute-intensive job allocation in data centers via deep reinforcement learning, IEEE Trans. Parallel Distrib. Syst. 31 (6) (2020) 1474–1485.
[26] Xin Zhou, Joint IT-facility optimization for green data centers via deep reinforcement learning, IEEE Netw. 35 (2021) 255–262.
[27] Z. peng, J. Lin, D. Cui, Q. Li, J. He, A multi-objective trade-off framework for cloud resource scheduling based on the deep Q-network algorithm, Cluster Comput. 23 (4) (2020) 2753–2767.
[28] Ashrae, Thermal Guidelines for Data Processing Environments, fourth ed., 2015.
[29] 6SigmaRoom, [Online]. Available:https://www.futurefacilities.com/products/6sigmaroom/. (Accessed 23 October 2023).
[30] Kaicheng Zhang, Machine learning-based temperature prediction for runtime thermal management across system components, IEEE Trans. Parallel Distrib. Syst. 29 (2018) 405–419.
[31] M. Stein, Large sample properties of simulations using latin hypercube sampling, Technometrics 29 (2) (1987) 143–151.
[32] W. Piątek, A. Oleksiak, G. Da Costa, Energy and thermal models for simulation of workload and resource management in computing systems, Simul. Model. Pract. Theory 58 (2015) 40–54.
[33] Z. Li, C. Yan, L. Yu, X. Yu, Energy-aware and multi-resource overload probability constraint-based virtual machine dynamic consolidation method, Future Gener. Comput. Syst. 80 (2018) 139–156.
[34] SPECpower_ssj2008, [Online]. Available: https://www.spec.org/power_ssj2008/results/power_ssj2008.html. (Accessed 23 October 2023).
[35] D. Shin, S.W. Chung, E. Chung, N. Chang, Energy-optimal dynamic thermal management: Computation and cooling power co-optimization, IEEE Trans. Ind. Inform. 6 (2010) 340–351.
[36] E. Lee, I. Kulkarni, D. Pompili, Proactive thermal management in green data centers, J. Supercomput. 60 (2) (2012) 165–195.
[37] R. Storn, K. Price, Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[38] Millie Pant Bilal, Hira Zaheer, Laura Garcí a Hernández, Ajith Abraham, Differential evolution: A review of more than two decades of research, Eng. Appl. Artif. Intell. 90 (2020) 103479.
[39] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of 6th Int. Symp. Micromachine Hum. Sci., 1995, pp. 39–43.
[40] D. Wang, D. Tan, L. Liu, Particle swarm optimization algorithm: an overview, Soft Comput. 22 (2018) 387–408.
[41] W. Chen, J. Zhang, Y. Lin, N. Chen, Z. Zhan, H.S. Chung, Y. Li, Y. Shi, Particle swarm optimization with an aging leader and challengers, IEEE Trans. Evol. Comput. 17 (2013) 241–258.

[42] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, F. Zhao, Energy-aware server provisioning and load dispatching for connection-intensive internet services, in: Proceedings of Usenix Symposium on Networked Systems Design & Implementation USENIX Association, 8, 2008, pp. 337–350.
[43] R.N. Calheiros, R. Ranjan, A. Beloglazov, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, Softw. - Pract. Exp. 41 (1) (2011) 23–50.
[44] Amazon EC2, [Online]. Available: https://aws.amazon.com/cn/ec2/instance-types/ (Accessed 23 October 2023).
[45] A. Beloglazov, A set of CPU utilization traces from PlanetLab VMs collected during 10 random days in March and April 2011, [Online]. Available: https://github.com/beloglazov/planetlab-workload-traces. (Accessed 23 October 2023).
[46] M. Shahrad, R. Fonseca, I. Goiri, Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider, in: Proceedings of the 2020 USENIX Annual Technical Conference, 2020, pp. 205–218, Available: https://github.com/Azure/AzurePublicDataset. (Accessed 20 March 2024).
[47] A. Alahmadi, A. Alnowiser, M.M. Zhu, D. Che, P. Ghodous, Enhanced first-fit decreasing algorithm for energy-aware job scheduling in cloud, in: Proceedings of the Int'L Conf. on Computational Science and Computational Intelligence, IEEE, 2014, pp. 69–74.

Jianpeng Lin received the M.S. degree in computer science at Guangdong University of Technology in 2019. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. His research interests include cloud computing, data center thermal management and sustainable computing.

Weiwei Lin received his B.S. and M.S. degrees from Nanchang University in 2001 and 2004, respectively, and his Ph.D. in Computer Application from South China University of Technology in 2007. He has been a visiting scholar at Clemson University from 2016 to 2017. Currently, he is a professor in the School of Computer Science and Engineering at South China University of Technology. His research interests include distributed systems, cloud computing, and AI application technologies. He has published more than 150 papers in refereed journals and conference proceedings. He has been a reviewer for many international journals, including IEEE TPDS, TSC, TCC, TC, TCYB, etc. He is a distinguished member of CCF and a senior member of the IEEE.

Wentai Wu received his Bachelor and Master degrees from South China University of Technology in 2015 and 2018, respectively. Sponsored by CSC, he received the Ph.D. degree in Computer Science in 2022 from the University of Warwick, United Kingdom. Since 2024 he has been with the College of Information Science and Technology, Jinan University, as an Associate Professor. His research interests mainly include distributed systems, edge intelligence, sustainable computing and collaborative machine learning. He has published over 20 research papers and serves as reviewer for high-impact journals and conferences such as IEEE TPDS, TMC, TBD, ICML and NeurIPS. He was listed among the top 2% scientists in Distributed Computing subfield in 2023 per composite citation indicator.

Wenjun Lin is currently working toward the master's degree in computer science from South China University of Technology. His research interests include cloud computing, AI application technologies.

Keqin Li is a SUNY Distinguished Professor of Computer Science with the State University of New York. He is also a National Distinguished Professor with Hunan University, China. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication. He has authored or coauthored over 870 journal articles, book chapters, and refereed conference papers. He is among the world's top 5 most influential scientists in parallel and distributed computing in terms of both single-year impact and career-long impact based on a composite indicator of Scopus citation database. He is currently an associate editor of the ACM Computing Surveys and the CCF Transactions on High Performance Computing. He has served on the editorial boards of the IEEE Transactions on Parallel and Distributed Systems, the IEEE Transactions on Computers, the IEEE Transactions on Cloud Computing, the IEEE Transactions on Services Computing, and the IEEE Transactions on Sustainable Computing. He is an IEEE Fellow and an AAIA Fellow. He is also a Member of Academia Europaea (Academician of the Academy of Europe).