Jointly Optimizing the IT and Cooling Systems for Data Center Energy Efficiency based on Multi-Agent Deep Reinforcement Learning

Ce Chi
University of Chinese Academy of Sciences
High Performance Computer Research Center, Institute of Computing Technology, Chinese Academy of Sciences
Beijing, China
chice18s@ict.ac.cn

Kaixuan Ji
University of Chinese Academy of Sciences
High Performance Computer Research Center, Institute of Computing Technology, Chinese Academy of Sciences
Beijing, China
jikaixuan@ict.ac.cn

Avinab Marahatta
University of Chinese Academy of Sciences
High Performance Computer Research Center, Institute of Computing Technology, Chinese Academy of Sciences
Beijing, China
avinab.marahatta@ict.ac.cn

Penglei Song
Capital Normal University
Beijing, China
spl_wx@126.com

Fa Zhang
High Performance Computer Research Center, Institute of Computing Technology, Chinese Academy of Sciences
Beijing, China
zhangfa@ict.ac.cn

Zhiyong Liu
High Performance Computer Research Center, Institute of Computing Technology, Chinese Academy of Sciences
Beijing, China
zyliu@ict.ac.cn

ABSTRACT

With the development and application of cloud computing, the increasing amount of data centers has resulted in huge energy consumption and severe environmental problems. Improving the energy efficiency of data centers has become a necessity. In this paper, in order to improve the energy efficiency of both IT and cooling systems for data centers, a model-free deep reinforcement learning (DRL) based joint optimization approach MACEEC is proposed. To improve the cooperation between IT and cooling system while handling the high-dimensional state space and the large hybrid discrete-continuous action space, a hybrid AC-DDPG multi-agent structure is developed. A scheduling baseline comparison method is proposed to enhance the stability of the architecture. And an asynchronous control optimization algorithm is developed to solve the different responding time issue between IT and cooling system. Experiments based on real-world traces data validate that MACEEC can effectively improve the overall energy efficiency for data centers while ensuring the temperature constraint and service quality compared with existing joint optimization approaches.

e-Energy'20, June 22–26, 2020, Virtual Event, Australia

ACM ISBN 978-1-4503-8009-6/20/06…$15.00
https://doi.org/10.1145/3396851.3402658

CCS CONCEPTS

• **Computer systems organization** → **Cloud computing**; • **Computing methodologies** → *Multi-agent systems.*

KEYWORDS

data center, energy efficiency, deep reinforcement learning, multi-agent, scheduling algorithm, cooling system

ACM Reference Format:
Ce Chi, Kaixuan Ji, Avinab Marahatta, Penglei Song, Fa Zhang, and Zhiyong Liu. 2020. Jointly Optimizing the IT and Cooling Systems for Data Center Energy Efficiency based on Multi-Agent Deep Reinforcement Learning. In *The Eleventh ACM International Conference on Future Energy Systems (e-Energy'20), June 22–26, 2020, Virtual Event, Australia.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3396851.3402658

1 INTRODUCTION

With the increasing demand for cloud computing, the number and scale of data centers are expanding rapidly, resulting in a large amount of power demand and severe environmental problems. As reported in [16] and [7], data centers in U.S. consumed about 70 billion kWh in 2014, accounting for 1.8% of total U.S. electricity consumption, and data centers emit about 0.3% of the world's carbon emissions each year. To protect the power grid and the environment, it has become crucial to improve the energy efficiency and reduce the energy consumption of data centers. The major energy consumption of a data center comes from two important subsystems: one is IT system (e.g., servers, networks), and the other is cooling system. Generally, the IT system consumes around 56% of the total energy consumption, while the cooling system accounts for about 30% energy consumption of a data center [22]. Notice that the actual proportion of the energy consumption of the IT and

cooling systems is dependent upon many factors, especially the environment where the data center is located, and some special energy saving technique applied for the IT and cooling systems. Thus, effective energy-saving technologies for data centers need to be developed by jointly optimizing the energy efficiency of IT system and cooling system.

There are four major challenges hindering efficient joint energy efficiency optimization of data centers [14]. Firstly, due to the complexity of dynamic changes of workload, heat and temperature in a data center environment, models are usually difficult to adequately describe a real data center. Therefore, based on models, the optimized IT and cooling system control strategies may not perform as well as theoretical analysis in practice. Secondly, as the tasks arrive online, the IT system is required to decide proper servers to run the tasks in a short time, according to the current system state. However, tens of thousands of servers make the dimension of the system state very high and make it difficult to make an optimal decision of server selection as quickly as required. As a result, an optimal strategy for task scheduling and low computational complexity are usually incompatible. Thirdly, selecting a suitable server for a task is a discrete action control, while adjusting the cooling facilities (e.g., air flow rate, air supply temperature) is a continuous action control. Jointly optimizing both discrete and continuous variables makes it more difficult to develop an optimal energy saving strategy. At last, in practice there usually exists a significant responding time difference between IT and cooling system. The IT system can usually respond within seconds or microseconds, while it could take minutes for the cooling system to respond. Thus, a joint control strategy needs to fully consider the backwardness of the cooling facilities to prevent hot-spots and realize high energy efficiency.

There have been many works focusing on improving the energy efficiency of IT system [2, 20, 21]. Meanwhile, many researches have been done for the energy consumption optimization of cooling system in data centers [3, 18, 24]. In these works, improving energy efficiency of data centers is achieved by optimizing either IT system or cooling system, while joint optimization of IT and cooling systems can usually bring more energy efficiency for data centers. Therefore, to efficiently reduce the total energy consumption of a data center, some joint optimization strategies have been proposed [1, 14, 17, 19]. In [1], servers were divided into multiple zones such as hot zone, warm zone and cool zone, based on which a heuristic task scheduling algorithm was developed to reduce energy consumption of both IT and cooling systems. A cross-layer energy consumption optimization method JOINT was proposed in [17], and models were built for IT and cooling systems, based on which two optimization algorithms were proposed. Based on superlinear cooling power models, a task classification algorithm was developed in [19] to keep the load of multiple cooling units balanced with less active servers so that total energy consumption can be optimized. In these works, energy saving strategies are achieved by optimizing the built models, where some models are simplified for optimization. And some heuristic algorithms are developed to reduce the computational complexity at the expense of some optimality. Instead of building models, based on deep reinforcement learning (DRL), a novel model-free approach DeepEE was proposed in [14]. Although DeepEE jointly considers the optimization of IT and cooling systems, it makes a decision for IT system after the cooling decision has been made. Therefore, DeepEE possibly misses the global optimal decision for IT and cooling systems, while cooperative decisions could improve the situation and further improve the energy efficiency of IT and cooling systems. Therefore, a joint optimization technique that optimizes IT and cooling systems cooperatively for data centers will be significant for energy saving in data centers.

In this paper, a joint optimization framework, MACEEC (Multi-Agent deep reinforcement learning-based Cooperative Energy Efficient Control of IT and cooling systems), is proposed to solve the challenges above and optimize the energy efficiency of both IT and cooling systems for data centers. MACEEC is built based on model-free DRL methods, by which the strategy is learnt through direct interactions with the real environment instead of modeling the data centers. Different from [14], a hybrid AC-DDPG multi-agent architecture is developed in this paper, so that cooperative decisions for IT and cooling systems can be generated for further energy efficiency. In order to reduce the instability of AC in the multi-agent architecture and reduce computational overhead, a scheduling baseline comparison method is proposed. To solve the different responding time issue between IT and cooling system, an asynchronous control optimization algorithm is proposed for the hybrid AC-DDPG architecture so that cooperative decisions for IT and cooling systems can be made in an asynchronous way. At last, experiments are conducted based on real world traces data to validate the performance of MACEEC, and the results validate that MACEEC can effectively improve the energy efficiency of data centers.

2 SYSTEM MODELS AND PROBLEM FORMULATION

2.1 System Architecture

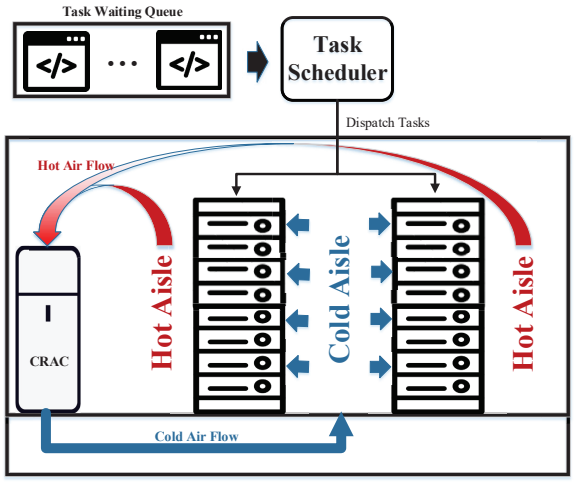

Figure 1: System architecture of a data center.

As shown in Fig. 1, the main components of a data center include a task waiting queue, a task scheduler, racks with servers and

Computer Room Air Conditioners (CRACs). Servers are the main component of the IT system and account for most of the energy consumption of the IT system. Generally multiple servers are installed on a rack, and there are usually multiple racks in a data center to provide sufficient computing power. When users submit their tasks to the data center for processing, with the arrival of tasks, the tasks are pushed into a task waiting queue to wait for resource allocation. In our model, we consider a first-come-first-service (FCFS) queue strategy, which means tasks that arrive early will be scheduled first. The task scheduler is responsible for selecting servers from all the servers for each task in the task waiting queue. After deciding the server for a task, the task scheduler will schedule the task to the target server, and a virtual machine (VM) will be created for the task so that the resources are provided to run the task. The server selection policy of the task scheduler have a great influence on the energy efficiency of both IT and cooling systems, which should be designed carefully as a result. While running tasks, servers need to consume a lot of power and will generate a lot of heat, which should be dissipated by the cooling system.

Although some advanced cooling techniques such as water cooling are becoming increasingly popular, air cooling is still the primary cooling approach in many data centers due to its low construction cost. We adopt CRAC architecture as the considered cooling system in this paper, which is widely used in practice and has been concerned by many works [1, 11, 17, 23, 24]. In the CRAC architecture, the servers and perforated floor tiles are installed on the raised floor. The space around the racks can be divided into two types: cold aisle and hot aisle. CRACs use the raised floor to deliver the cold air to the cold aisle through the perforated floor tiles under the cold aisle. The chassis fans installed on the racks can inhale the cold air for the servers. After absorbing the heat from the servers, the cold air becomes hot and comes out to the hot aisle as hot air. At last the hot air is collected by the CRACs, and then is expelled to the outside. The air flow rate and the supply air temperature of the CRACs are adjustable [14, 23], making it possible to improve the energy efficiency of the cooling system.

2.2 System Models

2.2.1 Server Model. Assume that there are M racks in a data center. Each rack i contains N_i servers. The j^{th} server in the rack i is denoted as s_{ij}, with total resources $\boldsymbol{r}_{ij}^{total} = (r_{ij,1}^{total}, ..., r_{ij,D}^{total})$. $r_{ij,d}^{total}$ $(d = 1, ..., D)$ is the amount of the d^{th} type of resource owned by s_{ij}, and D types of server resources are considered, such as CPU, RAM and disk. If a server is running some tasks, some of its resources are occupied. Then the available amount of resources of a server s_{ij} can be denoted as $\boldsymbol{r}_{ij}^{avail} = (r_{ij,1}^{avail}, ..., r_{ij,D}^{avail})$. According to $r_{ij,d}^{total}$ and $r_{ij,d}^{avail}$, the utilization of the d^{th} resource of a server s_{ij} can be calculated by $u_{ij,d} = 1 - r_{ij,d}^{avail} / r_{ij,d}^{total}$, and the utilization of all types of resources is denoted as $\boldsymbol{u}_{ij} = (u_{ij,1}, ..., u_{ij,D})$. The power consumption of a server s_{ij} is denoted as P_{ij}^{server}, which is obtained by monitoring the servers via power meters theoretically. However, it may be costly to monitor the power consumption of all servers in a large data center in real time, thus a power model of servers can help for cost reduction [9]:

$$P_{ij}^{server} = P_{static} + P_{dynamic} \cdot u_{ij,c}, \tag{1}$$

where P_{static} and P_{static} are the static and dynamic power consumption of a server respectively, and $u_{ij,c}$ is the CPU utilization of the server s_{ij}. Note that the power model of servers is used for the purpose of equipment cost reduction and is not indispensable for our proposed model-free DRL-based optimization framework. Finally, the total power consumption of the IT system can be calculated by $P^{IT} = \sum_{i=0}^{M} \sum_{j=0}^{N_i} P_{ij}^{server}$.

2.2.2 Cooling Model. As described in Section 2.1, the cold air flow rate supplied by the CRACs is denoted as f, and can be adjusted within a certain range:

$$f_{low} \leq f \leq f_{high}. \tag{2}$$

Similarly, the cold air temperature supplied by the CRACs is denoted as T^{sup}, and can be adjusted in a certain range:

$$T_{low}^{sup} \leq T^{sup} \leq T_{high}^{sup}. \tag{3}$$

The main responsibility of cooling system is to protect the servers from overheating, which can be formulated as [23]

$$T_i^{in} \leq T^{red}, \forall i \in \{1, ..., M\}, \tag{4}$$

where T_i^{in} is the inlet temperature of rack i, which is determined by the heat generated by the servers and T^{sup} and can be obtained by measuring. T^{red} is the redline threshold temperature, over which severe downtime incident and server damage may happen. The power consumption of the cooling system P^{cool} can be obtained from power meters [14].

2.2.3 Task Model. As the tasks arrive sequentially, the k^{th} task t_k can be characterized by its requested amount of resources $\boldsymbol{r}_k^{req} = (r_{ij,1}^{req}, ..., r_{ij,D}^{req})$. When assigning a task t_k to a server s_{ij}, the available resources should be no fewer than that requested by the task, i.e.,

$$r_{k,d}^{req} \leq r_{ij,d}^{avail}, \forall d \in \{1, ..., D\}. \tag{5}$$

2.3 Problem Formulation

Since the IT system and cooling system are the main energy-consuming components, both should be optimized for energy saving. In addition, to improve the energy efficiency of a data center as much as possible, it is significant to optimize the IT system and cooling system jointly rather than separately.

Formally, when assigning a waiting task t_k, the scheduling decision is to select a server z to run it, where

$$z \in \{0, 1, ..., \sum_{i=1}^{M} N_i\}, \tag{6}$$

and $z = 0$ means the task will not be assigned to any server and will be kept in the waiting queue until next scheduling time step.

The joint optimization objective is to minimize the energy consumption of the data center over a certain period of time by controlling both task scheduling z and cooling adjustment f and T^{sup}, while avoiding overload and overheat of servers. Thus, the joint energy consumption optimization problem can be formulated as follows:

$$\begin{aligned} \min \quad & \Gamma = \int_t P^{IT} + P^{cool} dt \\ s.t. \quad & (2), (3), (4), (5), (6) \end{aligned} \tag{7}$$

3 DRL-BASED JOINT OPTIMIZATION ALGORITHM DESIGN

3.1 DRL Definitions for Joint Optimization

Model-free DRL frameworks has shown their promising properties including that they can work perfectly without a priori knowing how the environment will respond to their actions. Namely, their policy is learnt by directly interacting the real environment, but not based on deterministic models. To fully utilize the excellent features of DRL and avoid modeling the complicated environment of data centers, we build the control strategies for IT and cooling systems based on DRL. We first make some definitions for the joint optimization problem based on DRL theory.

State Space: Let $\boldsymbol{r}^{avail} = (\boldsymbol{r}_{11}^{avail}, \boldsymbol{r}_{12}^{avail}, ..., \boldsymbol{r}_{MN_M}^{avail})$, $\boldsymbol{u} = (\boldsymbol{u}_{11}, \boldsymbol{u}_{12}, ..., \boldsymbol{u}_{MN_M})$, $\boldsymbol{P}^{server} = (P_{11}^{server}, P_{12}^{server}, ..., P_{MN_M}^{server})$, and $\boldsymbol{T}^{in} = (\boldsymbol{T}_1^{in}, \boldsymbol{T}_2^{in}, ..., \boldsymbol{T}_M^{in})$. Then, when scheduling a task t_k, the state can be represented as a combination of these vectors: $\boldsymbol{s} = (\boldsymbol{r}_k^{req}, \boldsymbol{r}^{avail}, \boldsymbol{u}, \boldsymbol{P}^{server}, \boldsymbol{T}^{in}, f, T^{sup})$.

Action Space: In our scenario, the action space contains two parts. The first one is the action of scheduling the current waiting task to which server z, where z should satisfy the constraint (6). Note that the server selection is a discrete action for the optimization problem. The second one is the action of adjusting the CRACs, including the air flow rate f and the air supply temperature T^{sup} with constraints (2) and (3) respectively. f and T^{sup} are continuous variables for the optimization problem.

Reward: The main target $P^{IT} + P^{cool}$ should be included in the reward for the optimization of energy consumption. Besides, the constraints (4), (5) should be considered in the reward as well to guide the agent to make decisions that avoid server overheating and overloading. Another variable δ is added to the reward to encourage the agent to assign tasks. When a task t_k is successfully assigned to a server at the current time step, $\delta = \delta_0 > 0$. Otherwise, if the task is kept in the task waiting queue, $\delta = 0$. In summary, the reward can be designed as follows:

$$\begin{aligned} r = \delta &- \beta_1 (P^{IT} + P^{cool}) \\ &- \beta_2 \ln(1 + \exp(T^{in} - T^{red})) \\ &- \beta_3 \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{d=1}^{D} \ln(1 + \exp(r_{ij,d}^{avail} - r^{QoS})) \\ &- \beta_4 \sum_{d=0}^{D} \ln(1 + \exp(r_{s,d}^{avail} - r^{QoS})), \end{aligned} \tag{8}$$

where r^{QoS} is a small constant, which is used to prevent overload of the servers.

3.2 Hybrid AC-DDPG Multi-Agent Model

For DRL, a discrete action space problem can usually be solved by DQN [13] or AC [8] framework. A continuous action space problem can usually be solved by AC or Deep Deterministic Policy Gradient (DDPG) [10] framework. DDPG is based on AC architecture but can usually produce actions with lower variance than AC because of its different parameter update method. Note that DDPG is only suitable for continuous action space problem. In these frameworks, a DRL agent is usually trained to produce either discrete action or continuous action, but uncapable of producing a hybrid discrete-continuous action. A state-of-the-art multi-agent framework is designed in [12] to train multiple agents to work cooperatively. However, it assumes that the agents' action spaces are the same, so the neuron networks (NNs) of the multiple agents are similar and the parameter update method can be designed intuitively. However, since there exists both discrete and continuous actions in the joint energy consumption optimization problem in data centers, the existing multi-agent frameworks are still inapplicable.

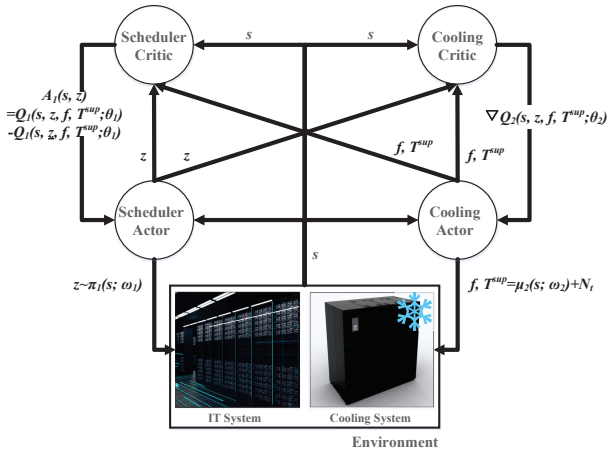

Figure 2: DRL Architecture of MACEEC.

In this paper, inspired by the previous works, we propose a heterogeneous cooperative multi-agent framework to deal with the hybrid discrete-continuous action control for the joint optimization in data centers. We choose AC and DDPG to build the heterogeneous multi-agent framework to control task scheduling and cooling adjustment respectively. The reason of not adopting DQN for task scheduling is that DQN needs to perform maximization operation for all actions in the action space, which is time-consuming. Thus, DQN is generally thought not suitable large discrete action space problems [5], whereas the task scheduling falls into this category due to the large number of candidate servers in a data center. Alternatively, AC is adopted for task scheduling in our framework. As described above, DDPG can produce actions with lower variance than AC, so DDPG is used for action decisions for cooling system. To combine the two frameworks reasonably and find a stable training method, a hybrid AC-DDPG multi-agent architecture is designed in this paper.

Fig. 2 illustrates the hybrid AC-DDPG multi-agent architecture. A scheduler actor, a cooling actor, a scheduler critic and a cooling critic are designed to interact with the environment and with each other for energy consumption optimization. The environment consists of the IT system and cooling system, and provides state information $\boldsymbol{s}$ to all the agents.

1) Scheduler Actor. The scheduler actor is built based on a normal actor structure of AC with NN parameter ω_1. It inputs the state $\boldsymbol{s}$, and outputs the target server $z \sim \pi_1(\boldsymbol{s}, z; \omega_1)$ that will run the current task.

2) Cooling Actor. The cooling actor is built based on a DDPG actor structure with NN parameter ω_2. It inputs the state $\boldsymbol{s}$ and outputs the decided air flow rate f and air supply temperature T^{sup} of the CRACs, where $(f, T^{sup}) = \mu_2(\boldsymbol{s}; \omega_2) + \mathcal{N}_t$, and $\mathcal{N}_t$ is a random value for DRL exploration. A target network of the cooling agent with parameter $\hat{\omega_2}$ is also implemented to make the training more stable [10].

After the scheduling action z and the cooling action f and T^{sup} are performed by the environment, the next system state $\boldsymbol{s'}$ can be observed, and the reward r can be calculated. The transition tuple $(\boldsymbol{s}, z, f, T^{sup}, r, \boldsymbol{s'})$ is saved to a replay buffer.

3) Scheduler Critic. The scheduler critic is responsible for scoring the actions made by the scheduler actor as well as the cooling actor, according to the current environment. The score can be represented as $Q_1(\boldsymbol{s}, z, f, T^{sup}; \theta_1)$. Only when the decisions made by both sides are good for the energy reduction will the score be high. After informed of the score, the scheduler actor will capture that in such a state what action can be good or bad, so that a cooperative policy can be trained gradually.

However, since both scheduler actor and cooling actor contribute to the score, directly using the score cannot instruct the scheduler actor to update in the proper direction. In addition, the random exploration of cooling actor can aggravate the training instability. Therefore, instead of the score, the actual contribution of the scheduler actor should be revealed to train the scheduler actor. Although in [6] a counterfactual baseline method is proposed to analyze contribution for multi-agent environment, it is achieved by going through all possible actions of an agent. So it is inapplicable for the optimization in data centers because of the large amount of candidate servers, i.e., large action space. To derive the agent's actual contribution and reduce the computational overhead in data center environment, in this paper, we propose to use a scheduling baseline action as a comparison. Specifically, the scheduling baseline action $\underline{z}$ can be produced by any available scheduling algorithm such as a heuristic scheduling algorithm that assigns the tasks to the coolest server, or other more advanced algorithm like [4]. Then, the scheduling critic runs again for the scheduling baseline action $\underline{z}$, and obtain the corresponding score $Q_1(\boldsymbol{s}, \underline{z}, f, T^{sup}; \theta_1)$. At last, an advantage function can be used for the scheduler actor's update:

$$A_1(\boldsymbol{s}, z) = Q_1(\boldsymbol{s}, z, f, T^{sup}; \theta_1) - Q_1(\boldsymbol{s}, \underline{z}, f, T^{sup}; \theta_1). \quad (9)$$

Specifically, based on AC theory, the update gradient of the scheduler actor can be calculated by

$$\nabla_{\omega_1} J_1 = A_1(\boldsymbol{s}, z) \cdot \nabla_{\omega_1} \log \pi_1(\boldsymbol{s}, z; \omega_1). \quad (10)$$

The update of the scheduler critic is realized by minimizing the mean squared loss:

$$L_1(\theta_1) = (Q_1(\boldsymbol{s}, z, f, T^{sup}; \theta_1) - y_1)^2, \quad (11)$$

$$y_1 = r + \gamma \cdot Q_1(\boldsymbol{s'}, z', f', T^{sup\prime}; \theta_1), \quad (12)$$

where $z' \sim \pi_1(\boldsymbol{s'}, z'; \omega_1)$ and $(f', T^{sup\prime}) = \mu_2(\boldsymbol{s'}; \hat{\omega_2})$. γ is the discount factor defined for the DRL framework.

4) Cooling Critic. A cooling critic is also designed, to score the action decisions produced by the scheduler actor and cooling actor under the state s, while the score is used to train the cooling actor. Based on DDPG critic, the cooling critic consists of two NNs: evaluation network with parameter θ_2 and target network with parameter $\hat{\theta_2}$. The training process is similar to the typical DDPG method as shown in Algorithm 1.

3.3 Asynchronous Control

To solve the different responding time issue between IT and cooling system, a two-time-scale control mechanism can be used for the IT and cooling systems [14]. To be specific, the agent is set to make a decision for task scheduling every time step and make a decision for the cooling system every L time steps. And the environment state is extended to $\boldsymbol{s} = (\boldsymbol{r}_k^{req}, \boldsymbol{r}^{avail}, \boldsymbol{u}, \boldsymbol{P}^{server}, \boldsymbol{T}^{in}, f, T^{sup}, \eta)$, where a variable η is added to represent the current system time state. Assume that the current system time step is t, and then the value of η can be determined as

$$\eta = \begin{cases} 1 & \text{if} \quad t \mod L = 0 \\ 0 & otherwise \end{cases} \quad (13)$$

In [14], a penalty factor is added to its reward function to gradually guide the agent to adjust the cooling system every L time steps, which not only could degrade the training performance but also is difficult to guarantee the agent to always follow the rule and could output a lot of noisy actions for the cooling system however. Different from [14], with the benefit of the high freedom of our proposed AC-DDPG architecture, there is no need to control the decision time step in a soft way. Instead, the cooling action is calculated only at its decision time step. Thus, the problem is solved without ambiguity.

Since the scheduler actor makes a series of decisions based on the current state $\boldsymbol{s}$ and a fixed policy ω_1, guided by the cooling critic, the cooling actor can gradually learn how to perform a better cooling decision based on the current state $\boldsymbol{s}$, so that a cooperative strategy ω_2 that is able to benefit the energy reduction of the scheduler agent for L time steps can be achieved by the cooling agent. Meanwhile, the scheduler actor strategy ω_1 is also optimized every L time steps, guided by a more reliable result from the scheduler critic, so that a cooperative strategy ω_1 that is able to facilitate the cooling agent to reduce energy consumption can be trained ultimately. Note that after the training, only scheduler actor and cooling actor are needed to control the systems, and the critics can be discarded for calculation reduction. Formally, Algorithm 1 presents the total process of the DRL-based joint optimization algorithm of MACEEC.

4 PERFORMANCE EVALUATION

In this section, we present the detailed environment settings for MACEEC. And the performance of MACEEC is evaluated.

4.1 Setting

The data center workload comes from real-world Google cluster traces data [15]. 10,000 tasks are extracted for training and another 10,000 tasks for test. For simplicity, we only take into account the CPU resource in the experiment. We consider a data center containing 20 racks, each rack equipped with homogeneous 42 1U servers. The P_{static} and $P_{dynamic}$ are set to $100W$ and $200W$ respectively [14]. The cooling model is built based on [22]. For the DRL model, the scheduler actor is built based on a 2-layer fully-connected NN, each layer with 400 and 300 units respectively. The first layer of the scheduler critic contains 400 units and takes

Algorithm 1 MACEEC

Initialize the networks scheduler actor $\pi_1(\boldsymbol{s}, z; \omega_1)$, scheduler critic $Q_1(\boldsymbol{s}, z, f, T^{sup}; \theta_1)$, cooling actor $\mu_2(\boldsymbol{s}; \omega_2)$ and cooling critic $Q_2(\boldsymbol{s}, z, f, T^{sup}; \theta_2)$ and their parameters ω_1, θ_1, ω_2 and θ_2 randomly
Initialize the target networks cooling actor $\mu_2(\boldsymbol{s}; \hat{\omega_2})$ and cooling critic $Q_2(\boldsymbol{s}, z, f, T^{sup}; \hat{\theta_2})$ and their parameters $\hat{\omega_2} = \omega_2$ and $\hat{\theta_2} = \theta_2$
Initialize the replay buffer $\mathcal{R}$
Initialize state $\boldsymbol{s}$
$\boldsymbol{g} = \boldsymbol{0}$
for t=1 to T **do**
  $z \sim \pi_1(\boldsymbol{s}, z; \omega_1)$
  if $t \mod L == 0$ **then**
    $f, T^{sup} = \mu_2(\boldsymbol{s}; \omega_2) + \mathcal{N}_t$
  end if
  Perform actions z, f and T^{sup}, and obtain the reward r and next state $\boldsymbol{s}'$
  Store transition $(\boldsymbol{s}, z, f, T^{sup}, r, \boldsymbol{s}')$ to $\mathcal{R}$
  Obtain an action $\underline{z}$ from a heuristic algorithm
  $\nabla_{\omega_1} J_1 = (Q_1(\boldsymbol{s}, z, f, T^{sup}; \theta_1) - Q_1(\boldsymbol{s}, \underline{z}, f, T^{sup}; \theta_1)) \cdot \nabla_{\omega_1} \log \pi_1(\boldsymbol{s}, z; \omega_1)$
  $\boldsymbol{g} = \boldsymbol{g} + \nabla_{\omega_1} J$
  $y_1 = r + \gamma \cdot Q_1(\boldsymbol{s}', z', f', T^{sup\prime}; \theta_1)|_{z' \sim \pi_1(\boldsymbol{s}', z'; \omega_1), (f', T^{sup\prime}) = \mu_2(\boldsymbol{s}'; \hat{\omega_2})}$
  Update the scheduler critic by minimizing the loss: $L_1(\theta_1) = (Q_1(\boldsymbol{s}, z, f, T^{sup}; \theta_1) - y_1)^2$
  $\boldsymbol{s} = \boldsymbol{s}'$
  Sample a random minibatch of transitions $\mathcal{B} = \{(\boldsymbol{s}_j, z_j, f_j, T_j^{sup}, r_j, \boldsymbol{s}_j')\}_{1 \le j \le B}$ from $\mathcal{R}$
  $y_{2,j} = r_j + \gamma \cdot Q(\boldsymbol{s}_j', z_j', f_j', T_j^{sup\prime}; \hat{\theta})|_{z_j' \sim \pi_1(\boldsymbol{s}_j', z_j'; \omega_1), (f_j', T_j^{sup\prime}) = \mu_2(\boldsymbol{s}_j'; \hat{\omega_2})}$ $j = 1, ..., B$
  Update the cooling critic by minimizing the loss: $L_2(\theta_2) = \frac{1}{B} \sum_{j=1}^{B} ((Q(\boldsymbol{s}_j, z_j, f_j, T_j^{sup}; \theta_2) - y_{2,j})^2)$
  if $t \mod L == 0$ **then**
    Apply update gradients $\frac{1}{L} \cdot \boldsymbol{g}$ to the scheduler actor ω_1
    $\boldsymbol{g} = \boldsymbol{0}$
    Update cooling actor ω_2 using the gradient: $\nabla_{\omega_2} J_2 = \frac{1}{B} \sum_{j=1}^{B} \nabla_{(f, T^{sup})} Q(\boldsymbol{s}_j, z_j, f, T^{sup}; \theta)|_{f, T^{sup} = \mu_2(s_j; \omega_2)} \cdot \nabla_{\omega_2} \mu_2(\boldsymbol{s}_j; \omega_2)$
  end if
  Update the target networks of cooling actor and cooling critic:
  $\hat{\omega_2} = \tau \omega_2 + (1 - \tau) \hat{\omega_2}$
  $\hat{\theta_2} = \tau \theta_2 + (1 - \tau) \hat{\theta_2}$
end for

the state vector as input. The second layer of the scheduler critic contains three parts: the output of the first layer, the scheduling action layer which takes the scheduling action as input and the cooling action layer which takes the cooling action as input. Each of the three part contains 300 neurons. Similarly, the cooling actor and cooling critic have the same structure and parameter as the scheduler actor and critic. When training, γ is set to 0.99, batch size $B = 64$, and $\tau = 0.001$. For the reward function, $\delta_0 = 1.0$, $\beta_2 = 10,000$, $\beta_3 = 10,000$, $\beta_4 = 10,000$ and $r^{QoS} = 0.1$ [14]. From [14], T^{red} is set to 30℃ to avoid overheat or damage of servers. And β_1 is set to 1 divided by the maximum IT power consumption of the data center. For the asynchronous control, the scheduling action is performed every 10 seconds and the cooling action is performed every 5 minutes [14], i.e., $L = 30$.

4.2 Experiment Results

Two joint control strategies are used to compare with our proposed algorithm. One is a model-based joint control algorithm PowerTrade-d [1], and the other is a DRL-based joint control algorithm DeepEE [14].

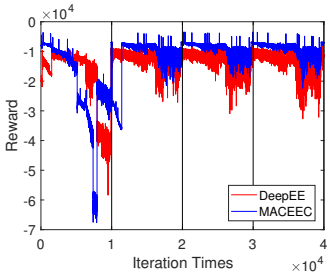

Figure 3: Reward of DeepEE and MACEEC while iterating.

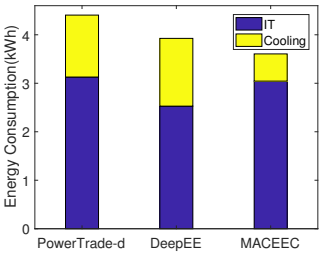

Figure 4: Energy consumption of PowerTrade-d, DeepEE and MACEEC.

4.2.1 Training Convergence. Firstly, we compare the convergence procedure of our proposed method with DeepEE. The experiments for DeepEE and MACEEC are performed 4 episodes, totally 40000 times of training to converge. As shown in Fig. 3, in the first episode, both DeepEE and MACEEC obtain low rewards. This is because initially both algorithms need to perform random actions for exploration, which results in poor energy efficient task scheduling and cooling control. However, with further training and policy updates, DeepEE converges quickly and gets high rewards in the following episodes. Note that the target of the agents is to maximize the accumulative sum of the rewards during an episode. Although MACEEC converges slower than DeepEE, it ends up with higher rewards and achieves more stable performance, compared with DeepEE.

4.2.2 Energy Consumption. Then, based on test data, the energy consumption of MACEEC, DeepEE and PowerTrade-d is compared in Fig. 4. From Fig. 4, we can find that MACEEC consumes more energy on IT system than DeepEE, but saves a lot of energy from cooling system. As a result, MACEEC outperforms DeepEE in energy saving. It is because MACEEC aims at developing a cooperative policy from the beginning, so it is trained from a global perspective and can find a policy that minimizes the sum of energy consumption of IT and cooling systems. In addition, both MACEEC and DeepEE consume less energy consumption than PowerTrade-d, which shows that the DRL-based methods can well handle the scheduling and cooling control in data centers.

4.2.3 Task waiting time and hotspots. The average task waiting time comparison of MACEEC, DeepEE and PowerTrade-d is illustrated in Fig. 5. As shown in Fig. 5, MACEEC produces results

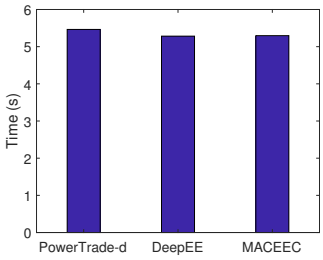

Figure 5: Average task waiting time of PowerTrade-d, DeepEE and MACEEC.

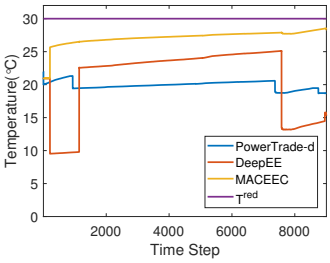

Figure 6: Average inlet air temperatures of PowerTrade-d, DeepEE and MACEEC.

similar to the other two algorithms, so the quality of service of MACEEC is verified. In addition, Fig. 6 presents the average inlet air temperatures of MACEEC, DeepEE and PowerTrade-d at each time step. By comparing the inlet air temperatures of the racks with T^{red}, we can see that the numbers of generated hotspots of the three algorithms during the test are all 0. So it validates that MACEEC can improve energy efficiency for data centers while guaranteeing the temperature constraint. In addition, combined with Fig. 5, it also validates that MACEEC reduces more energy by better coordinating the IT system and cooling system but not by generating more hotspots or by sacrificing some service quality.

5 CONCLUSION

Jointly optimizing both IT and cooling systems has become an effective way to improve the energy efficiency of data centers. However, there remains challenges in the joint optimization problem, such as inadequate model, high-dimensional state and action space, hybrid discrete-continuous variable optimization and different responding time issue. In this paper, we present a DRL-based control approach MACEEC to handle the challenges and improve energy efficiency of IT and cooling systems for data centers. In MACEEC, a hybrid AC-DDPG multi-agent structure is developed to better coordinate scheduling and cooling control. Meanwhile, a scheduling baseline comparison method is introduced to further stabilize the architecture. An asynchronous control optimization algorithm for the hybrid AC-DDPG multi-agent structure is proposed so that the scheduling decision and cooling decision can be produced asynchronously. Experiments based on real-world traces data show that MACEEC can save more energy than existing joint optimization approaches, and thus the effectiveness of MACEEC is validated.

ACKNOWLEDGMENTS

This work was partially supported by the National Key Research and Development Program of China (grant numbers 2017YFB1010001); the National Natural Science Foundation of China (grant numbers 61520106005, 61761136014). The Corresponding Author is Zhiyong Liu (zyliu@ict.ac.cn).

REFERENCES

[1] Faraz Ahmad and TN Vijaykumar. 2010. Joint optimization of idle and cooling power in data centers while maintaining response time. In *ACM Sigplan Notices*, Vol. 45. ACM, 243–256.
[2] G Prasad Babu and AK Tiwari. 2019. Energy Efficient Scheduling Algorithm for Cloud Computing Systems Based on Prediction Model. *International Journal of Advanced Networking and Applications* 10, 5 (2019), 4013–4018.
[3] Alessandro Beghi, Luca Cecchinato, Giuseppe Dalla Mana, Michele Lionello, Mirco Rampazzo, and Enrico Sisti. 2017. Modelling and control of a free cooling system for data centers. *Energy Procedia* 140 (2017), 447–457.
[4] Muhammad Tayyab Chaudhry, Teck Chaw Ling, Atif Manzoor, Syed Asad Hussain, and Jongwon Kim. 2015. Thermal-aware scheduling in green data centers. *ACM Computing Surveys (CSUR)* 47, 3 (2015), 1–48.
[5] Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2015. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679* (2015).
[6] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2018. Counterfactual multi-agent policy gradients. In *Thirty-second AAAI conference on artificial intelligence*.
[7] Nicola Jones. 2018. How to stop data centres from gobbling up the world's electricity. *Nature* 561, 7722 (2018), 163–167.
[8] Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*. 1008–1014.
[9] Fanxin Kong and Xue Liu. 2014. A survey on green-energy-aware power management for datacenters. *ACM Computing Surveys (CSUR)* 47, 2 (2014), 1–38.
[10] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).
[11] Zhenhua Liu, Yuan Chen, Cullen Bash, Adam Wierman, Daniel Gmach, Zhikui Wang, Manish Marwah, and Chris Hyser. 2012. Renewable and cooling aware workload management for sustainable data centers. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*. 175–186.
[12] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*. 6379–6390.
[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.
[14] Yongyi Ran, Han Hu, Xin Zhou, and Yonggang Wen. 2019. DeepEE: Joint Optimization of Job Scheduling and Cooling Control for Data Center Energy Efficiency Using Deep Reinforcement Learning. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 645–655.
[15] Charles Reiss, John Wilkes, and Joseph L Hellerstein. 2011. Google cluster-usage traces: format+ schema. *Google Inc., White Paper* (2011), 1–14.
[16] Arman Shehabi, Sarah Smith, Dale Sartor, Richard Brown, Magnus Herrlin, Jonathan Koomey, Eric Masanet, Nathaniel Horner, Inês Azevedo, and William Lintner. 2016. United states data center energy usage report. (2016).
[17] Jianxiong Wan, Xiang Gui, Ran Zhang, and Lijun Fu. 2017. Joint cooling and server control in data centers: A cross-layer framework for holistic energy minimization. *IEEE Systems Journal* 12, 3 (2017), 2461–2472.
[18] Qingzhu Wang, Yonghao Yu, Bin Li, and Yihai Zhu. 2019. Tensor-Based Optimal Temperature Control of CRACs in Multi-Datacenters. *IEEE Access* 7 (2019), 41445–41453.
[19] Youshi Wang, Fa Zhang, Rui Wang, Yangguang Shi, Hua Guo, and Zhiyong Liu. 2017. Real-time Task Scheduling for joint energy efficiency optimization in data centers. In *2017 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 838–843.
[20] Xiaolong Xu, Wanchun Dou, Xuyun Zhang, and Jinjun Chen. 2015. EnReal: An energy-aware resource allocation method for scientific workflow executions in cloud environment. *IEEE transactions on cloud computing* 4, 2 (2015), 166–179.
[21] Xiaolong Xu, Xuyun Zhang, Maqbool Khan, Wanchun Dou, Shengjun Xue, and Shui Yu. 2017. A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems. *Future Generation Computer Systems* (2017).
[22] Weiwen Zhang, Yonggang Wen, Yew Wah Wong, Kok Chuan Toh, and Chiu-Hao Chen. 2016. Towards joint optimization over ICT and cooling systems in data centre: A survey. *IEEE Communications Surveys & Tutorials* 18, 3 (2016), 1596–1616.
[23] Ziqi Zhao, Fan Wu, Shaolei Ren, Xiaofeng Gao, Guihai Chen, and Yong Cui. 2016. Tech: A thermal-aware and cost efficient mechanism for colocation demand response. In *2016 45th international conference on parallel processing (ICPP)*. IEEE, 464–473.
[24] Rongliang Zhou, Zhikui Wang, Cullen E Bash, Alan McReynolds, Christopher Hoover, Rocky Shih, Niru Kumari, and Ratnesh K Sharma. 2011. A holistic and optimal approach for data center cooling management. In *Proceedings of the 2011 American Control Conference*. IEEE, 1346–1351.