Towards Sustainability-Aware Edge Computing Systems: A Heterogeneous MADRL Approach

Yan Chen, *Member, IEEE,* Tarik Taleb, *Senior Member, IEEE,* Qize Guo, Ignacio Lacalle Úbeda, and Tarik Zakaria Benmerar

***Abstract*—The proliferation of computation-intensive services has raised growing concerns regarding the sustainability of future computing infrastructures. This paper investigates distributed edge computing systems powered by hybrid energy sources, where we jointly consider energy consumption from computation, cooling, and power source switching. To improve sustainability, we formulate an optimization problem that aims to minimize conventional energy consumption by jointly coordinating task distribution and power source switching. The problem involves complex resource constraints, heterogeneous dynamics, partial observability, and coupled task-power decisions, rendering conventional optimization methods ineffective. To address this challenge, we decompose the problem into task allocation, task selection, and power switching subproblems, following the system execution logic and constraints. Then, we propose an end-to-end solution based on heterogeneous cooperative multi-agent deep reinforcement learning, in which dedicated agents are designed to address individual subproblems while maintaining system coordination. Extensive simulations results demonstrate that the proposed approach achieves enhanced coordination and substantial reductions in conventional energy consumption, resulting in improved overall system sustainability.**

***Index Terms*—Edge Computing, Sustainability, Green Edge, Multi-Agent, and Reinforcement Learning.**

I. Introduction

MULTI-ACCESS edge computing (MEC) systems have been widely deployed to accommodate computation-intensive applications [1]. However, the growth of such applications, particularly those driven by artificial intelligence (AI), has substantially increased energy consumption, making sustainability a critical concern [2]. Therefore, extensive efforts have been made to develop energy-efficient technologies, including lightweight AI models [3]–[5], intelligent task offloading [6], [7], optimized cooling systems [8], [9], and renewable energy integration [10]–[13]. As computational workloads directly influence the operational states and energy cost of MEC subsystems, task allocation strategies play a pivotal role in modern MEC systems composed of interconnected components, such as cooling systems, power supplies, and computational modules, which require coordinated and integrated management.

In the MEC community, energy efficiency can be improved through task allocation, typically achieved via joint optimization of task offloading, scheduling, and resource allocation [14]–[16]. In parallel, renewable energy integration has been explored to further improve sustainability [11], [17]. By enabling resource sharing and load balancing, collaboration among distributed edge sites (ESs) increases system capacity and supports geographically distributed workload execution [18]. This geographically distributed execution allows workloads to be aligned with local renewable energy availability, contributing to improved sustainability. This potential has been demonstrated in practice: China's East-West Compute Transfer Project offloads computing workloads from eastern to western regions with abundant renewable energy resources [19], while companies such as Meta and Intel deploy data centers in regions with favorable natural cooling conditions to achieve carbon neutrality.

However, realizing sustainable MEC remains challenging due to uncertain renewable energy availability, dynamic task requirements, heterogeneous edge capabilities, and complex inter-dependencies among task processing, operational, and cooling subsystems. The joint consideration of distributed renewable energy management and non-computational energy consumption has not been sufficiently explored in existing studies. Specifically, the interactions between task allocation and cooling systems are intricate, as workload distribution directly affects thermal managements. Similarly, although dynamic power source switching is essential for optimizing renewable energy utilization, its associated energy overhead and stability implications remain insufficiently explored in existing MEC optimization frameworks [20], [21]. Furthermore, accurately modeling comprehensive energy consumption remains challenging, limiting the applicability of traditional optimization methods that rely on precise mathematical models and global information.

Deep reinforcement learning (DRL) has been widely adopted to address system heterogeneity, dynamics, and partial observability [22]. However, the complex dependencies among multiple action spaces (e.g., task allocation and power switching), together with system constraints (e.g., ES capacity limits), make joint optimization over multi decision variables highly intractable. To alleviate this complexity, ex-

Yan Chen, Tarik Taleb, and Qize Guo are with the Faculty of Electrical Engineering and Information Technology, Ruhr University of Bochum, Bochum, 44801, Germany. e-mail: yanchen@ieee.org, guoqize@hotmail.com, tarik.taleb@rub.de; Ignacio Lacalle Úbeda is with Communications Department, Universitat Politècnica de València, 46022 Valencia, Spain. e-mail:iglaub@upv.es; Tarik Zakaria Benmerar is with ICTFICIAL Oy, Espoo, 02130 Finland. e-mail: tarik.benmerar@ictficial.com. *Correspondng author Qize Guo*

This work is supported in part by the Federal Ministry of Research, Technology and Space (BMFTR), Germany, through the Project 6GEM+ under Grant No. 16KIS2411; in part by the European Union's HE research and innovation program HORIZON-JUSNS-2023 through the 6G-Path project under Grant No. 101139172; and in part by the European Union's Horizon 2020 Research and Innovation Program through the aerOS project under Grant No. 101069732.

Manuscript received October 10, 2025; revised February 4, 2026.

isting studies typically decomposes the original problem into several subproblems based on action spaces or execution logic, which are then addressed separately through conventional techniques [18], [23], [24]. Although such decomposition reduces action space dimensionality, many subproblems remain challenging due to practical constraints and tight subsystem coupling. Multi-agent deep reinforcement learning (MADRL) offers a promising alternative by allowing different agents to specialize in distinct decision dimensions, such as task allocation, energy management, and cooling control, while learning coordinated policies. This distributed paradigm effectively manages complex subsystem dependencies without requiring explicit mathematical models, addressing both the scalability and complexity challenges of modern MEC systems.

This paper investigates distributed MEC systems powered by hybrid energy supplies, and proposes an efficient approach for improving system sustainability through joint task allocation and power management. An end-to-end (E2E) MADRL-based framework is developed to addresses task-power interdependencies and distributed resource constraints via coordinated multi-stage decision making. The main contributions are summarized as:

- We formulate a sustainability-oriented optimization problem for distributed MEC systems powered by hybrid energy supplies, aiming to jointly optimize task allocation, power switching, and task selection. The system model comprehensively incorporates computation, cooling, and power management costs, while satisfying edge capacity constraints.
- We propose an E2E approach based on heterogeneous MADRL, in which specialized agents collaboratively address the subproblems of task allocation, task selection, and power switching, thereby handling the interdependencies among action spaces and individual constraints. Moreover, the approach enhances exploration in ultra-large discrete action spaces through individualized entropy regularization and parameter-space noise, and guarantees provable convergence to a soft Nash equilibrium.
- We conduct extensive evaluations based on numerical simulations and real-world renewable energy generation datasets. The results demonstrate that the proposed approach achieves significant improvements in system sustainability and consistently outperforms benchmarks.

The remainder of this paper is organized as follows. Section II reviews related works. Section III describes the system model and problem formulation. Section IV presents the proposed MADRL-based approach. Section V provides simulation results and performance analysis. Section VI concludes the paper and discusses future research directions.

II. Related Works

A. Sustainable Edge Computing

Energy efficiency is fundamental requirement for sustainable MEC. Recent studies have investigated energy optimization across various MEC scenarios using diverse approaches. For instance, DRL for heterogeneous vehicular-integrated MEC systems [14], [15], convex optimization and game theory for air-ground integrated systems [16], and Lyapunov optimization for collaborative vehicular environments [25].

With growing sustainability concerns, renewable energy integration in MEC has attracted increasing attention. Fang et al. [10] introduced the green edge concept, and leveraged DRL to enable energy-efficient cloud-edge collaboration. Despite inherent uncertainty and intermittency, renewable energy has been widely explored in Internet of Things (IoT) (often termed energy harvesting). Luo et al. [17] addressed solar energy variability through DRL-based power management under battery constraints, while Ma et al. [26] optimized carbon emissions and inference accuracy through task allocation between renewable-powered devices and grid-connected servers. However, these studies primarily focus on device-level renewable energy under simplified assumptions (single energy source, limited capacity), limiting their applicability to system-level MEC environments.

Edge data centers provide stable platforms for renewable energy integration. Ma et al. [11] investigated task offloading under hybrid power supply, while Liao et al. [27] proposed carbon-aware task placement with renewable energy sharing among distributed servers. However, these works overlook cooling energy, which constitutes a significant portion of data center consumption. Chen et al. [20] and Pei et al. [28] demonstrated that cooling constitutes a non-negligible component of overall energy consumption, and Pei et al. further achieved efficiency gains through DRL-based adaptive cooling control.

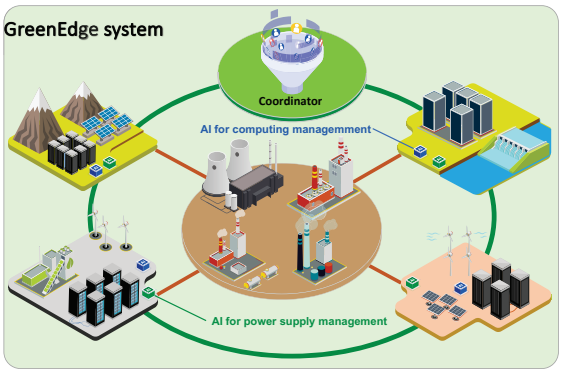
In contrast to existing studies that consider renewable energy and cooling in isolation, this work examines the combined effects of computation, cooling, and power switching costs under hybrid energy sources. This resulting integrated framework is designed to improve overall system sustainability.

B. Multi-Agent DRL in Edge Computing

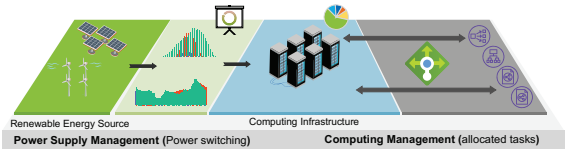
MADRL has demonstrated strong potential in handling partial observability and coordinating heterogeneous agents, and has been widely applied to resource allocation, task offloading, and content caching in MEC systems [29]. For example, Liu et al. [30] modeled each user as a Q-learning agent that autonomously selected wireless resources, guided by a global reward to minimize system costs. Zhang et al. [31] studied hierarchical multi-cloud MEC systems where each cloud server learned task scheduling strategies through agent interaction, significantly reducing processing latency. Recent advances have introduced specialized techniques to improve training efficiency. Gao et al. [32] enhanced the actor-critic framework with dual centralized critics, dueling networks, and attention mechanisms to compress observations, achieving significant cost reductions in large-scale offloading scenarios. Liu et al. [33] customized reward structures to enhance computation rate and convergence in IoT systems. Given the interdependence among offloading decisions, resource states, and network conditions, joint optimization of multiple objectives has received increasing attention. However, most existing works are limited to homogeneous agents with independent decision variables that can be optimized simultaneously [14], [34].

To tackle complex inter-dependencies, some studies adopt hybrid approaches, in which certain decomposed subproblems are addressed using conventional optimization methods, while DRL is employed to handle system heterogeneity, dynamics, and partial observability. For example, MADRL was applied to trajectory optimization, whereas offloading decisions were handled using threshold-based algorithms [35]. Similarly, MADRL was used for distributed task allocation, followed by sigmoidal programming for resource allocation [18]. More recently, hierarchical frameworks have been introduced to address decomposed subproblems, primarily with the aim of reducing the complexity associated with multiple action spaces [36]. However, these approaches typically decompose the overall problem into independent subproblems based on hierarchical principles and train each component separately using single-agent reinforcement learning. As a result, the couplings among subproblems or the coordination among the resulting policies are often overlooked.

Our work differs fundamentally by investigating inter-dependencies among subproblems and individual constraints, where the central agent's outputs directly influences subordinate agents' inputs, and joint actions collectively determine the global rewards. Moreover, to enhance exploration challenges in ultra-large discrete action spaces, we introduce individualized entropy regularization with parameter-space noise, which improves both learning efficiency and solution quality.



(a)



(b)

Fig. 1: System model of a distributed green MEC system. (a) distributed green edge infrastructure. (b) power supply management and load distribution.

III. System Model and Problem Formulation

We consider a distributed green MEC system (Fig. 1) consisting of a set of geographically distributed ESs denoted by $\mathcal{H}$. Each ES is powered by a hybrid energy system integrating renewable and conventional energy sources. A Power Supply Management System (PSMS) is deployed at each ES to ensures stable energy supply through coordinated power switching with an Renewable Energy Storage System (RESS), which stores harvested renewable energy. All ESs are connected to a central cloud server powered by conventional energy, which processes tasks offloaded from the edge to ensure their performance. We denote the complete server set, including both edge and cloud servers, as $\mathcal{H}^+$. Moreover, a centralized coordinator orchestrates inter-site collaboration, including system-wide optimization and resource allocation.

We assume that the system operates in discrete time slots $t \in \{1, 2, \ldots\}$ over an infinite horizon. At each time slot, the coordinator observes newly arriving tasks from a set of registered applications $\mathcal{U}$ running on heterogeneous devices and obtains the RESS states of all ESs. Then, the coordinator generates a global task allocation decision. Upon receiving the assigned tasks, each ES processes them according to its local policy and resource conditions. For notational simplicity, the subscript t denotes the time slot unless otherwise stated, and explicit references to time slots are omitted when clear from the context.

A. Quality of Service Model

A task request generated by application $u \in \mathcal{U}$ at time slot t is characterized by a tuple $\mathcal{T}_{u,t} = (H_t^u, V_t^u, R_t^u, \rho_t^u, \tau_t^u)$, where H_t^u denotes the ES directly associated with the source device of application u, V_t^u is the task data size, R_t^u is the result data size, ρ_t^u is the computing intensity defined as the number of floating-point operations (FLOPs) required per bit, and τ_t^u specifies the latency constraint. The service delay consists of the processing latency ($T_{u,t}^p$) and the communication latency ($T_{u,t}^f$) [18], [37], i.e.,

$$T_{u,t} = T_{u,t}^p + T_{u,t}^f = \frac{V_t^u \times \rho_t^u \times \eta_u^{\hat{H}_t^u}}{\mathcal{F}_{\hat{H}_t^u}} + \frac{V_t^u + R_t^u}{\mathcal{B}_{H_t^u, \hat{H}_t^u}}, \quad (1)$$

where $\hat{H}_t^u \in \mathcal{H}^+$ denotes the server to which $\mathcal{T}_{u,t}$ is allocated for processing. Since computational performance varies across different facilities in practical systems, the factor η_u^h refers to the deviation from the base intensity ρ_t^u when tasks from u are executed on server h [18]. We denote $\mathcal{F}_h$ as the computing capacity of server h, measured in Floating-Point Operations Per Second (FLOPS), which depends on its clock frequency f_h and processor architecture, i.e., $\mathcal{F}_h = f_h \times f_h^{ipc} \times f_h^{flop}$. Here, f_h^{ipc} is the number of instructions that the computing unit of server h can execute per clock cycle, and f_h^{flop} is the number of FLOPs performed per instruction. Note that $\mathcal{F}_h$ refers to the capacity of a single computing unit of h, and each server may comprise multiple such units, enabling parallel task execution. $\mathcal{B}_{H_t^u, \hat{H}_t^u}$ denotes the effective data transmission rate between server H_t^u and server $\hat{H}_t^u$. To maintain the Quality of Service (QoS), the total service delay should not exceed the specified latency constraint, i.e.,

$$\textbf{C1: } T_{u,t} \leq \tau_t^u, \forall u, \forall t. \quad (2)$$

B. Energy Consumption Model

Information technology (IT) facilities, cooling systems, and power switching operations account for over 91% of the total energy usage [21] in modern computing environments. Thus, we focus on these three components, as they dominate energy consumption and are directly impacted by the workload.

(1) IT facilities: IT facilities consist of computation, storage, and communication components. Activating these facilities incurs a baseline power consumption P_h^{active}, which varies across ESs and can be obtained through empirical measurement. Following common assumptions, we treat P_h^{active} as constant for each ES. Beyond P_h^{active}, a substantial amount of energy is expended during task processing. To ensure applications' QoS, we assume that each computing unit operates at full capacity. Following [38], [39], we model the energy consumed by the IT facilities of ES h at time slot t, denoted by $E_{h,t}^{IT}$, as the sum of computation energy consumption ($E_{h,t}^{comp}$), activation energy consumption ($E_{h,t}^{active}$), and data communication energy consumption ($E_{h,t}^{Tx,Rx}$), i.e.,

$$\begin{aligned} E_{h,t}^{IT} &= E_{h,t}^{comp} + E_{h,t}^{active} + E_{h,t}^{Tx,Rx} \\ &= \kappa_h f_h^3 \sum_{u\in\mathcal{U}} x_{u,t}^h T_{u,t}^p + P_h^{active} \max_{u\in\mathcal{U}_t^h}\{T_{u,t}\} + E_{h,t}^{Tx,Rx}. \end{aligned} \tag{3}$$

Here, κ_h denotes the effective switched capacitance [39]. The binary variable $x_{u,t}^h$ indicates whether $\mathcal{T}_{u,t}$ is allocated to ES h. The set $\mathcal{U}_t^h$ includes all applications whose source device are associated with or whose tasks are offloaded to ES h during slot t. The data communication energy consumption $E_{h,t}^{Tx,Rx}$ (including data transmission and reception) can be modeled as

$$E_{h,t}^{Tx,Rx} = P_h^{Tx} V_{h,t}^{Tx} + P_h^{Rx} V_{t,h}^{Rx}, \tag{4}$$

where P_h^{Tx} and P_h^{Rx} represent the energy consumption of sending and receiving one bit of data at ES h, respectively. $V_{h,t}^{Tx}$ and $V_{h,t}^{Rx}$ denote the data size transmitted from and received by ES h.

(2) Cooling system: Cooling energy consumption depends on the thermal load generated by IT facilities and is influenced by factors such as fan placement, cooling medium, and management policies. Due to these complex dependencies, deriving a unified mathematical model is challenging. Nevertheless, prior studies show that cooling energy of an ES h at time slot t ($E_{h,t}^{cool}$) is predominantly determined by heat dissipation from IT facilities, i.e.,

$$E_{h,t}^{cool} = \frac{E_{h,t}^{IT}}{\boldsymbol{CoP}_{h,t}}. \tag{5}$$

The Coefficient of Performance (CoP) of a cooling system quantifies its cooling efficiency and is defined as the ratio of cooling output to energy input [38], [40]. A higher $\boldsymbol{CoP}_{h,t}$ indicates better cooling efficiency, implying lower energy consumption for the same cooling output. In practice, $\boldsymbol{CoP}_{h,t}$ is commonly modeled as a function of the supply air temperature $\mathbb{T}_{h,t}^{in}$ [41], i.e.,

$$\boldsymbol{CoP}_{h,t} = \boldsymbol{CoP}_h(\mathbb{T}_{h,t}^{in}, \mathbb{T}_{h,t}^{c}) = \zeta_h \frac{\mathbb{T}_{h,t}^{in}}{\mathbb{T}_{h,t}^{c} - \mathbb{T}_{h,t}^{in}}. \tag{6}$$

Here, $\mathbb{T}_{h,t}^{c}$ represents the coolant temperature at ES h, which is typically related to ambient temperature. We assume that all power consumed by IT facilities is fully converted into heat. Both $\mathbb{T}_{h,t}^{in}$ and $\mathbb{T}_{h,t}^{c}$ are expressed in Kelvin (K). The coefficient $\zeta_h < 1$ reflects the practical cooling efficiency at ES h, accounting for deviations from theoretical CoP due to deployment- and environment-specific factors [40].

(3) Power switching: Each power switching operation incurs non-negligible energy consumption due to auxiliary subsystems such as uninterruptible power supplies or microgrid reconfiguration. We assume that the PSMS operates on conventional energy to ensure reliability. Accordingly, the energy consumption related to each switching operation at ES h is modeled as a constant $E_h^{switch} = C$. During system operation, the energy supply dynamically switches between renewable and conventional sources. When powered by renewable energy, IT facilities are powered by the on-site RESS. Once residual renewable energy $\mathbf{E}_{h,t}^{\boldsymbol{g}}$ is depleted, the PSMS automatically switches to conventional sources. Therefore, we have

$$\textbf{C2: } \mathbf{E}_{h,t}^{\boldsymbol{g}} \geq 0 \quad \text{or} \quad E_{h,t}^{green} \leq \mathbf{E}_{\boldsymbol{h,t}}^{\mathbf{green,c}}, \forall h \in \mathcal{H}, \forall t. \tag{7}$$

Here, $E_{h,t}^{green}$ denotes the amount of renewable energy consumed by ES h. $\mathbf{E}_{\boldsymbol{h,t}}^{\mathbf{green,c}}$ represents available renewable energy stored in the RESS at ES h, which satisfies $\mathbf{E}_{\boldsymbol{h,t}}^{\mathbf{green,c}} \leq \mathbf{E}_h^{\mathbf{green}}$. $\mathbf{E}_h^{\mathbf{green}}$ denotes the maximum RESS capacity of ES h. At the beginning of each time slot, switching energy consumption $E_{h,t}^{\boldsymbol{s}}$ is initialized to zero. Each time a power switching operation occurs, $E_{h,t}^{s}$ is increased by $E_{h,t}^{\boldsymbol{s}} = E_{h,t}^{\boldsymbol{s}} + E_h^{switch}$.

Additionally, the number of tasks processed at each ES in a given time slot is limited by its processing capacity. Tasks exceeding this capacity are offloaded to the cloud. Therefore, we have

$$\textbf{C3: } \sum_{u\in\mathcal{U}} x_{u,t}^h \leq \mathbf{X}^{\boldsymbol{h}}, \forall t, \forall h, \tag{8}$$

where $\mathbf{X}^{\boldsymbol{h}}$ denotes the maximum number of tasks that ES h can process simultaneously. Moreover, each task can be allocated to at most one ES, i.e.,

$$\textbf{C4: } \sum_{h\in\mathcal{H}} x_{u,t}^h \leq 1, \forall t, \forall u. \tag{9}$$

C. Problem Formulation

According to (3) and (5), system energy consumption is given by the sum of IT, cooling, and power switching energy consumption across all ESs, i.e.,

$$\begin{aligned} E_t^{total} &= \sum_{h\in\mathcal{H}} E_{h,t}^{IT} + E_{h,t}^{cool} + E_{h,t}^{\boldsymbol{s}} \\ &= \sum_{h\in\mathcal{H}} \Big(\big(1 + \frac{1}{\boldsymbol{CoP}_h(\mathbb{T}_{h,t}^{in}, \mathbb{T}_{h,t}^{c})}\big) E_{h,t}^{IT} + E_{h,t}^{\boldsymbol{s}} \Big). \end{aligned} \tag{10}$$

To promote sustainability, we consider two primary objectives: (a) maximizing the utilization of renewable energy, and (b) minimizing the overall energy consumption, particularly non-computational consumption (e.g., cooling). These objectives may conflict with each other. For instance, allocating more tasks to an ES with abundant renewable energy improves renewable energy utilization, but may increase the overall energy

consumption if that ES operates with lower energy efficiency. Therefore, we formulate a unified objective that balances these goals by minimizing the consumption of conventional energy:

$$\textbf{P1:} \quad \min_{\boldsymbol{x},\boldsymbol{y}} \frac{1}{T}\sum_{t=1}^{T}(E_t^{total} - E_t^{green}), \quad s.t., \textbf{C1, C2, C3, C4}. \tag{11}$$

The decision variables $\boldsymbol{x} = \{x_{u,t}^h \mid u \in \mathcal{U}, t \in T, h \in \mathcal{H}^+\}$ and $\boldsymbol{y} = \{y_t^h \mid h \in \mathcal{H}, t \in T\}$ represent task allocation (TA) and power switching (PS) decisions, respectively. $y_t^h \in \{0,1\}$ indicates whether a power switching operation to renewable energy occurs at ES h. Moreover, we use E_t^{green} to denote the renewable energy consumed in time slot t, which is the sum of renewable energy consumed by each ES h ($E_{h,t}^{green}$), i.e., $E_t^{green} = \sum_{h\in\mathcal{H}} E_{h,t}^{green}$. In addition, we focus on hybrid energy systems where renewable sources complement conventional grid power, and thus exclude scenarios powered exclusively by renewable energy.

D. Problem analysis

Before proceeding to solution design, we first analyze the computational complexity of **P1**. To this end, we consider a restricted version of **P1** by focusing on a single time slot. Besides, we ignore constraints **C1** and **C2**, as well as the variation in transmission and computation costs incurred when a task is allocated to different servers. Moreover, we assume that all servers are powered by renewable energy at the beginning of the time slot and set $E_h^{switch} = 0, \forall h \in \mathcal{H}^+$. Under these assumptions, the resulting problem reduces to a multiple knapsack problem (MKP), which is known to be NP-hard. Therefore, **P1** is NP-hard even in the single-time-slot case with simplified restrictions. However, the decision-making process can be formulated as a Markov Decision Process (MDP). Since decision time for a batch of task requests is short, the system remains stable during this interval, i.e., task and ES states remain consistent throughout decision-making and execution. Furthermore, given the observed state at each time slot, subsequent system transitions depend solely on the current state, selected actions, and system dynamics, satisfying the Markov property. This makes the MDP framework well-suited for this problem. However, achieving sustainability in edge systems presents the following key challenges:

- **Limited prior knowledge and system dynamics:** In practical MEC systems, many parameters, such as exact computing intensity or subsystem energy models, are not directly observable. Conventional optimization methods that rely on strong modeling assumptions and global knowledge thus become impractical. In addition, dynamic task requests and fluctuating resources require real-time decision-making under evolving system states, posing major challenges for traditional approaches, which often need extensive iterations even under fixed snapshots.
- **Coupling between decisions and constraints:** Local decisions at ESs are inherently coupled with global TA. ESs face multi-dimensional constraints, such as energy availability and task concurrency limits, which are not fully observable by the central coordinator. As a result, global TA decisions may violate local constraints, requiring ESs to further adapt their executions based on local states. Furthermore, even after problem decomposition, the resulting subproblems remain challenging due to mutually interdependent decisions. For example, task selection and power switching are tightly coupled and subject to uncertain constraints (e.g., QoS and resource availability), rendering the resulting subproblems intractable even after decomposition.

As a result, the formulated problem requires a decision-making framework that can operate under partial state observability, handle strongly coupled decision spaces and individual constraints, as well as adapt to dynamic and uncertain environments. To address the above challenges, we propose an E2E MADRL-based approach that operates effectively under dynamic system conditions and partial state observability. For clarity, the notations used in the system model and problem formulation are summarized in TABLE I.

IV. Multi-Agent DRL-based E2E Approach

A. Framework

Building upon the analysis of problem **P1**, we propose a MADRL-based approach that is aligned with the E2E operational workflow of distributed MEC systems and respects to individual system constraints. As illustrated in Fig. 2, the proposed ***GreenEdge*** architecture consists of a meta agent and edge agents, corresponding to the central coordinator and distributed ESs, respectively.

- **Meta agent:** The meta agent is operated by the centralized coordinator and is responsible for global TA decisions. At each time slot t, the coordinator retrieves observable state information of all tasks and ESs, denoted by $\dot{s}_t$, and feeds it into the meta policy $\dot{\pi}$ to generate a meta action $\dot{a}_t$, from which the TA action a_t^{TA} is derived.
- **Edge agent:** Each edge agent is operated by its corresponding ES to optimize local task selection and power management policies after the execution of global TA. Specifically, each edge agent inputs its local state $\ddot{s}_t^h$ (including allocated task features and local states) into the edge policy $\ddot{\pi}^h$ and output $\ddot{a}_t^h$, from which the task selection (TS) $\ddot{a}_t^{TS}$ and the PS $\ddot{a}_t^{PS}$ actions are derived.

After executing TA, TS, and PS actions, tasks are either processed at designated ESs or forwarded to the cloud. Upon task completion, each ES reports its energy consumption to the central coordinator. Then, the experience is uploaded and stored for policy training.

During this workflow, tasks whose estimated edge processing delay exceeds their latency constraints are offloaded to the cloud to satisfy **C1**. This mechanism implicitly encourages agents to learn policies adhering to latency requirements, as cloud offloading results in higher conventional energy consumption, which is contrary to the optimization objective and thus acts as an implicit negative reward signal.

TABLE I: List of Notations

Notation	Definition
$\mathcal{H}/\mathcal{H}^+$	Set of edge sites/Set of all servers
$\mathcal{U}$	Set of registered applications (arrived tasks)
$\mathcal{U}_t^h$	Applications related to ES h
$\mathcal{T}_{u,t}$	Task request from application u
H_t^u	ES associated with application u's device
V_t^u	Data size from application u's task
R_t^u	Data size of application u's computing result
ρ_t^u	Computing intensity application u's task
τ_t^u	Latency constraint of application u's task
$\hat{H}_t^u$	Server to which u's task is allocated
$\mathcal{B}_{h,\hat{h}}$	Effective data rate between serve h and $\hat{h}$
η_u^h	Performance deviation of u's tasks on ES h
$\mathcal{F}_h$	Capacity of server h's computing unit
f_h	Clock frequency of ES h's computing unit
f_h^{ipc}	Instructions per clock cycle of ES h
f_h^{flop}	FlOPs per instruction of ES h
κ_h	Effective switched capacitance of ES h
$T_{u,t}$	Service latency of application u's task
$T_{u,t}^p$	Processing latency of application u's task
$T_{u,t}^f$	Communication delay of u's task and result
$E_{h,t}^{IT}$	Energy consumption of ES h's IT facilities
$E_{h,t}^{comp}$	Computation energy consumption of ES h
P_h^{active}	Activation power of ES h
$E_{h,t}^{active}$	Activation energy consumption of ES h
$V_{h,t}^{Tx}$	Data size transmitted from ES h
$V_{h,t}^{Rx}$	Data size received by ES h
$E_{h,t}^{Tx,Rx}$	Communication energy consumption of h
$E_{h,t}^{cool}$	Cooling energy consumption of ES h
E_h^{switch}	Energy cost per switching operation at h
$E_{h,t}^{\boldsymbol{s}}$	Total switching energy consumption at h
$E_{h,t}^{green}$	Renewable energy consumption of edge h
$\mathbf{E}_{h,t}^{\boldsymbol{g}}$	Residual renewable energy of ES h
$\mathbf{E}_{\boldsymbol{h,t}}^{\mathbf{green,c}}$	Maximum available renewable energy of h
$\mathbf{E}_h^{\mathbf{green}}$	Maximum capacity of ES h's RESS
E_t^{total}	System total energy consumption
E_t^{green}	System total renewable energy consumption
$\mathbb{T}_{h,t}^{in}$	Target air temperature of cooling system
$\mathbb{T}_{h,t}^{c}$	Temperature of coolant
ζ_h	CoP adjustment coefficient
$x_{u,t}^h$	Binary TA indicator of application u to h
$\mathbf{X}^{\boldsymbol{h}}$	Maximum number of concurrent tasks on h
y_t^h	Renewable energy switching indicator of h

Note: All symbols with subscript t are defined with respect to time slot t; u denotes the application u, and h denotes the server h.

B. Definitions

To implement the proposed MADRL-based GreenEdge framework, we define the state spaces, action spaces, and a unified reward function as follows:

Meta state: The observable state of the meta agent encompasses real-time attributes of all task requests and partial status

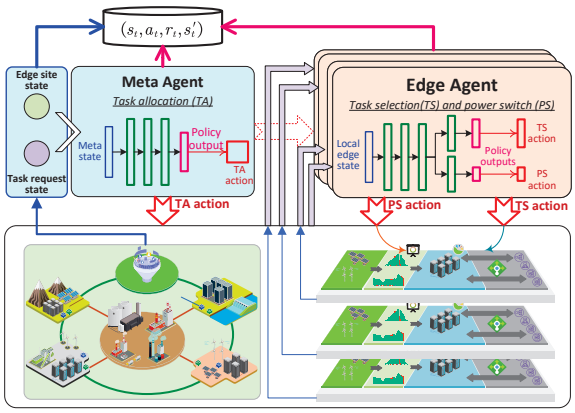


Fig. 2: The proposed E2E GreenEdge architecture.

of ESs, i.e.,

$$\dot{s}_t = \{H_t^u, V_t^u, R_t^u, \tau_t^u | u \in \mathcal{U}\} \cup \{\mathbf{E}_{h,t}^{\boldsymbol{g}}, \dot{y}_t^h, \mathbb{T}_{h,t}^{in}, \mathbb{T}_{h,t}^{\boldsymbol{c}} | h \in \mathcal{H}\}, \tag{12}$$

where $\dot{y}_t^h \in \{0,1\}$ denotes whether ES h uses renewable ($\dot{y}_t^h = 1$) or conventional ($\dot{y}_t^h = 0$) energy at the beginning of time slot t. Computing intensity and coefficient ζ_h are excluded due to measurement difficulties and treated as unobservable environmental factors that policies must implicitly adapt to.

Meta Action: The meta agent assigns incoming task requests from applications to appropriate ESs. The meta action at time slot t is defined as

$$\dot{a}_t = [\dot{x}_{u,t}^h]_{|\mathcal{U}| \times |\mathcal{H}|}, \tag{13}$$

where $\dot{x}_{u,t}^h \in \{0,1\}$ indicates whether the task from application u is allocated to ES h at time slot t (by the meta agent). For each task, exactly one ES is selected (**C4**), i.e., $\sum_{h \in \mathcal{H}} \dot{x}_{u,t}^h = 1, \forall u$.

Local edge state: Each edge agent's local state observation includes the real-time features of the tasks allocated to it ($\mathcal{U}_t^{h\downarrow}$), as well as the internal state, i.e.,

$$\ddot{s}_t^h = \{H_t^u, V_t^u, R_t^u, \tau_t^u | u \in \mathcal{U}_t^{h\downarrow}\} \cup \{\mathbf{E}_{h,t}^{\boldsymbol{g}}, \dot{y}_t^h, \mathbb{T}_{h,t}^{in}, \mathbb{T}_{h,t}^{\boldsymbol{c}}\}. \tag{14}$$

Local edge action: Each edge agent h selects a subset of allocated tasks for local processing and determines its power supply strategy, i.e.,

$$\ddot{a}_t^h = \left\{[\ddot{x}_{u,t}^h]_{1 \times |\mathcal{U}|}, y_t^h\right\}, \forall h \in \mathcal{H}. \tag{15}$$

where $\ddot{x}_{u,t}^h \in \{0,1\}$ denotes whether task from application u is selected for local processing by ES h. The TS action is defined over the maximum number of applications to ensure adaptability under all possible TA conditions. During execution, only entries corresponding to tasks allocated to ES h are activated, while the remaining entries are masked, i.e., $x_{u,t}^h = \dot{x}_{u,t}^h \cdot \ddot{x}_{u,t}^h$. Here, y_t^h represents the PS action, indicating the selection between conventional and renewable energy by ES h at time slot t.

Reward: To align the minimization objective in (11) with the reward maximization paradigm of RL, we define the

reward as the negative of conventional energy consumption:

$$r_t = -\mathcal{R}^s(E_t^{total} - E_t^{green}), \tag{16}$$

Additionally, we define an auxiliary reward for cases where renewable energy fully satisfies demand (i.e., $E_t^{green} = E_t^{total}$). This auxiliary reward is non-negative since $E_t^{green} \leq E_t^{total}$, and decreases with energy consumption to encourage conservation in fully green scenarios. Then, the reward function is

$$r_t = \begin{cases} -\mathcal{R}^s(E_t^{total} - E_t^{green}), & \text{if } E_t^{total} > E_t^{green} \\ 5e^{-0.1\frac{E_t^{total}}{\mathcal{R}^s}}, & \text{if } E_t^{total} = E_t^{green} \end{cases} \tag{17}$$

where $\mathcal{R}^s$ is a scaling factor to normalize the reward signal and reduce gradient explosion risk during training. We set $\mathcal{R}^s = 1/720$, which can be adjusted based on the reward range.

C. Centralized training of GreenEdge policies

In the investigated problem, agents are assigned distinct tasks and operate with heterogeneous state and action spaces, forming a highly dynamic multi-agent environment. All agents face large stochastic state-action spaces with strong inter-agent dependencies and require synchronized coordination at each decision step. For example, the state of each edge agent is directly influenced by meta actions, necessitating global cooperation for system-level optimization. Furthermore, online data sampling in networked systems is often costly. To address these challenges, we adopt an off-policy MADRL framework with centralized training and distributed execution.

As illustrated in Fig. 2, the coordinator collects and stores interaction experiences in a shared replay buffer and reused it for policy training, thereby improving data efficiency. We use $\mathbf{\Pi_\theta}$ to denote the set of all policies, i.e., $\mathbf{\Pi_\theta} = \{\dot{\pi}_{\bar{\theta}}, \ddot{\pi}^1_{\theta_1}, \ddot{\pi}^2_{\theta_2}, \ldots, \ddot{\pi}^{|\mathcal{H}|}_{\theta_{|\mathcal{H}|}}\}$, where $\bar{\theta}$ and θ_h denote parameters of the meta policy $\dot{\pi}$ and the edge policy $\ddot{\pi}^h$, respectively. All agents collaboratively optimize a shared objective of minimizing conventional energy consumption. Specifically, each edge agent makes decisions based on its local observations conditioned on the meta actions, while the resulting joint action determines the global system reward. Consequently, the objective of each policy $\pi_\theta \in \mathbf{\Pi_\theta}$ is to maximize the expected long-term cumulative reward given the observed states, i.e.,

$$\mathcal{J}(\theta) = \mathbb{E}_{\substack{s_t \sim \mathcal{S}, \\ \boldsymbol{a}_t \sim \mathbf{\Pi_\theta}}} \left[\sum_{t=1}^{T} r_t\big(s_t, \mathbf{\Pi_\theta}(s_t)\big) \right], \tag{18}$$

where $\mathbf{\Pi_\theta}(s_t) = (\dot{\pi}_{\bar{\theta}}(\dot{s}_t), \ddot{\pi}^1_{\theta_1}(\ddot{s}^1_t), \cdots, \ddot{\pi}^{|\mathcal{H}|}_{\theta_{|\mathcal{H}|}}(\ddot{s}^{|\mathcal{H}|}_t))$ denotes the joint actions of all agents. For brevity, we use $\boldsymbol{a}_t$ to denote $\mathbf{\Pi_\theta}(s_t)$. $\mathcal{S}$ denotes the system state space. The local state observations of edge agents are determined by the meta agent's action together with their internal states, i.e., $s_t \times \dot{\pi}_\theta(s_t) \Rightarrow \ddot{\boldsymbol{s}}^{\boldsymbol{h}}_{\boldsymbol{t}}, h \in \mathcal{H}$. Given a system state s_t, the expected cumulative reward after implementing the joint actions $\boldsymbol{a}_t$ is captured by the joint state-action value (Q-value) function, which satisfies the Bellman equation with discount factor $\gamma \in [0, 1)$, i.e.,

$$Q(s_t, \boldsymbol{a}_t) = r_t(s_t, \boldsymbol{a}_t) + \gamma \mathbb{E}_{s_{t+1} \sim \mathcal{S}; \mathbf{\Pi_\theta}} \left[V\left(s_{t+1}\right) \right], \tag{19}$$

where $r_t(s_t, \boldsymbol{a}_t)$ is the system reward given s_t and $\boldsymbol{a}_t$. $V\left(s_{t+1}\right)$ is the state value of s_{t+1}. To improve training stability under non-stationary multi-agent policies, we adopt centralized state-action value estimation during training. Specifically, the Q-function is conditioned on the global system state and the joint actions of all agents, incorporating all available observations and decisions. Such centralized value estimation provides provides more stable and accurate learning signals, since the mapping from joint state-action pairs to rewards is more precisely captured at the system level than from individual agent perspectives. The joint Q-values are approximated by deep neural networks to accommodate the vast state and action spaces. Policy gradients are then computed by backpropagation through the centralized Q-function, i.e.,

$$\nabla_\theta \mathcal{J}_\theta = \mathbb{E}_{\mathcal{S}, \mathbf{\Pi_\theta}}[\nabla_\theta \nabla_{\pi_\theta(\hat{s}^\theta_t)} Q(s_t, \dot{a}_t, \ddot{a}^1_t, \cdots, \ddot{a}^{|\mathcal{H}|}_t)]. \tag{20}$$

For notational simplicity, we use θ_i to denote the parameters of policy $\pi_{\theta_i} \in \mathbf{\Pi_\theta}$ to be updated. Let $\hat{s}^{\theta_i}_t$ denotes the local state observation of the agent associated with policy π_{θ_i}, which is extracted from the global system state s_t. To enhance exploration, action entropy is incorporated into policy updates via soft policy iteration [42]. Accordingly, the objective is to learn the optimal policy $\pi^*_{\theta_i}$, i.e.,

$$\pi^*_{\theta_i} = \arg\max_{\pi_\theta} \mathbb{E}_{\substack{s_t \sim \mathcal{S}, \\ \boldsymbol{a}_t \sim \mathbf{\Pi_\theta}}} \left[\sum_{t=1}^{T} \Big(r(s_t, \boldsymbol{a}_t) + \alpha \mathbb{H}\big(\pi_{\theta_i}(\cdot|s_t)\big) \Big) \right], \tag{21}$$

where α is a learnable temperature parameter that controls the influence of entropy term. $\mathbb{H}(\cdot)$ denotes the entropy of action distribution induced by π_{θ_i} given s_t. We implement a multi-agent soft policy iteration scheme, which alternates between multi-agent soft policy evaluation and improvement [42].

Q-values are estimated using deep neural networks (critic) to handle large state and action spaces. In policy evaluation, the critic with parameter ψ_{θ_i} is updated by minimizing

$$\mathcal{L}_{\psi_{\theta_i}} = \mathbb{E}_{\mathcal{D}}[(\widehat{Q}_{\psi_{\theta_i}}(s_t, \boldsymbol{a}_t) - Y_t)^2] \tag{22}$$

$$Y_t = r_t + \gamma(\widehat{Q}_{\psi_{\theta_i}}(s_{t+1}, \widehat{\mathbf{\Pi}}_\theta(s_{t+1}))), \tag{23}$$

where $\widehat{Q}_{\psi_{\theta_i}}(s_t, \boldsymbol{a}_t)$ is the soft Q-value, which can be calculated based on (21) and the corresponding soft value function, i.e., $\widehat{V}_i(s_t) \triangleq \mathbb{E}_{\boldsymbol{a}_t \sim \mathbf{\Pi_\theta}}[Q_{\psi_{\theta_i}}(s_t, \boldsymbol{a}_t) + \alpha \mathbb{H}\big(\pi_{\theta_i}(\cdot|s_t)\big)]$. Then, the loss function for soft policy improvement is

$$\mathcal{L}_{\theta_i} = \mathbb{E}_{\mathcal{D}}\big[\pi_{\theta_i}(s^{\theta i}_t)^\top \big[\alpha \log \pi_{\theta_i}(a^{\theta_i}_t | s^{\theta i}_t) - Q_{\psi_{\theta_i}}(s_t, \widehat{\mathbf{\Pi}}_\theta(s_t))\big]\big], \tag{24}$$

All actions are regenerated by the current policies, where only the action sampled from policy π_{θ_i} retains gradients for its update. To improve update efficiency, the multi-agent soft policy improvement adopts a sequential updates scheme, in which previously updated policies are reused when optimizing subsequent agents. Accordingly, the joint policies in (24) is rewritten as, $\widehat{\mathbf{\Pi}}_{\boldsymbol{\theta}}(\cdot) := \{\pi^{new}_{\boldsymbol{\theta}_{\boldsymbol{i}-}}, \pi_{\theta_i}, \pi^{old}_{\boldsymbol{\theta}_{\boldsymbol{i}^+}}\}$. Here, $\pi^{new}_{\boldsymbol{\theta}_{\boldsymbol{i}-}}$ represents the set of policies that have been updated prior to updating π_{θ_i}, while $\pi^{old}_{\boldsymbol{\theta}_{\boldsymbol{i}^+}}$ denotes the set of policies that are yet to be updated after π_{θ_i}. The policies are updated following an arbitrary random permutation order.

D. Enhanced exploration

As problem **P1** features large state and action spaces, conventional exploration strategies such as stochastic action sampling and ϵ-greedy selection commonly used in continuous or small discrete action spaces are ineffective. To address this challenge, we adopt Gumbel noise and parameter-space noise in action generation [43], which promote effective exploration while while preserving differentiability.

Parameter-space noise is injected directly into policy parameters, inducing consistent behavioral perturbations that enhance exploration without introducing additional hyperparameter. We employ NoisyNet for the output layer of each policy [43]. Specifically, we replace standard linear layers with NoisyLinear layers, which inject noise into both weights and biases. The noise parameters are perturbed as $W = \mu_W + \sigma_W \odot \varepsilon_W$ and $b = \mu_b + \sigma_b \odot \varepsilon_b$, where μ_W and μ_b are the deterministic (mean) parameters, σ_W and σ_b are learnable noise scales parameters, and ε_W, ε_b are zero-mean noise variables sampled at each forward pass. To ensure computational efficiency, we use factorized Gaussian noise [43]. Let $\epsilon_{\text{in}} \in \mathbb{R}^n$ and $\epsilon_{\text{out}} \in \mathbb{R}^m$ be independently sampled from a standard normal distribution. The noise is then scaled as

$$f(x) = \text{sign}(x)\sqrt{|x|},\ \varepsilon_W = f(\epsilon_{\text{out}}) \cdot f(\epsilon_{\text{in}})^\top, \varepsilon_b = f(\epsilon_{\text{out}}). \tag{25}$$

The parameters σ_W and σ_b are updated via backpropagation together with policy parameters, enabling agents to adaptively control exploration during training. This leads to more stable and efficient learning than heuristic strategies like ε-greedy, especially in high-dimensional or non-stationary environments.

Gumbel noise $\mathcal{N}_{gumble}$ is added to the policy outputs to encourage exploration, i.e., $a_t^{\theta_i} = \pi_{\theta_i}(s_t^{\theta_i}) + \mathcal{N}_{gumble}, \forall \pi_{\theta_i} \in \boldsymbol{\Pi_\theta}$. The noisy actions are then mapped to discrete decisions by implicit activation functions with the Straight-Through Estimator (STE) to preserve gradient information. Specifically, $a_t^{TA} = \text{Gumbel-softmax}(\dot{\pi}(\dot{s}_t))_{\text{hard}}$, $a_{h,t}^{PS} = \text{Gumbel-softmax}(\ddot{\pi}^h(\ddot{s}_t^h)_{PS})_{\text{hard}}$, and $a_{h,t}^{TS} = \text{Gumbel-sigmoid}(\ddot{\pi}^h(\ddot{s}_t^h)_{TS})$. Here, $\text{Gumbel-softmax}(\cdot)_{\text{hard}}$ produces one-hot actions while enabling backpropagation, effectively reducing action-space complexity during training [18], [44]. For the PS action, $\ddot{\pi}^h(\ddot{s}_t^h)_{PS}$ adopts a binary classification approach to determine the final action. Since TS involves selecting multiple tasks with varying quantities under different conditions, Gumbel-sigmoid is used to generate selection probabilities for candidate tasks, and a threshold of 0.5 is used to determine whether a task is selected. To satisfy ES constraints (**C3**), the top $K = \mathbf{X}^h$ valid candidate tasks with highest scores are selected for execution when the TS action violates constraint(s). Gumbel noise promotes exploration in early training and is gradually annealed as the policy converges.

The training process for GreenEdge policies is detailed in Algorithm 1. Policies are separately deployed at the central coordinator and individual ESs. During training, the initialized policies interact with the environment for T_c episodes to collect experiences. Once the replay buffer $\boldsymbol{\mathcal{D}}$ contains sufficient samples, policy updates are triggered. To alleviate Q-value overestimation, double critics are employed, and target networks with soft updates and delayed policy updates can improve training stability [45]. The delayed update frequency is controlled by λ_1.

Algorithm 1 Centralized Training of GreenEdge policies

Input: Distributed Edge system, Coordinator, meta-policy, edge policies, and replay buffer $\boldsymbol{\mathcal{D}}$

Initialize actors and target actors: $\dot{\pi}_\theta, \{\ddot{\pi}_{\theta_h}^h | \forall h \in \mathcal{H}\}$
Initialize critics and target critics: $\{(\psi_{1,\theta_i}, \psi_{2,\theta_i}) | \forall \pi_{\theta_i} \in \boldsymbol{\Pi}_\theta\}$
Distribute actor policies to corresponding facilities
for $\bar{t} = 1, 2, 3, \cdots$ **do**
  Operate deployed actors to collect experience
  if $\bar{t} > T_c$ **then**
    Sample experience: $< \boldsymbol{s_t}, \boldsymbol{a_t}, \boldsymbol{r_t}, \boldsymbol{s_{s+1}} > \sim \boldsymbol{\mathcal{D}}$
    Draw a random permutation $i = 1 : |\mathcal{H}|+1$ of agents
    ===Update Critics===
    for $\pi_{\theta_i} \in \boldsymbol{\Pi}_\theta$ **do**
      Generate new a_{t+1} for s_{t+1} by latest actors $\widehat{\boldsymbol{\Pi}}_\theta$
      Calculate $\mathcal{L}_{\psi_{\theta_i}}$ based on (22) and (23)
      $\psi_{1,\theta_i}, \psi_{2,\theta_i} = \text{Optimizer}(\mathcal{L}_{\psi_{\theta_i}})$
      $\psi_{j,\theta_i}^- = (1-\mu)\psi_{j,\theta_i}^- + \mu\psi_{j,\theta_i}, j \in \{1, 2\}$
    ===Update Actors===
    if $\bar{t}\%\lambda_1 == 0$ **then**
      for $\pi_{\theta_i} \in \boldsymbol{\Pi}_\theta$ **do**
        Generate new a_t for s_t with updated actors $\widehat{\boldsymbol{\Pi}}_\theta$
        calculate $\mathcal{L}_{\theta_i}$ based on (24)
        $\theta_i = \text{Optimizer}(\mathcal{L}_{\theta_i})$
      Reset the noise of all NoisyLinear layers.
    if Actors are updated **then**
      Synchronize parameters to corresponding actors

E. Algorithm Analysis

The computational complexity encompasses both training and inference phases. Both the meta and edge agents adopt MLP-based actors and critics, where M_a and M_c denote the number of hidden layers, and N_l^a and N_l^c denote the hidden size of the l-th layer of the actor and critic networks, respectively.

(1) Inference Complexity: Gumbel noise and parameter-space noise are disabled during inference. Each edge actor has input dimension $4|\mathcal{U}| + 4$ and output dimension $|\mathcal{U}| + 2$. With softmax and sigmoid operations at the output, the inference complexity is: $\mathcal{O}((4|\mathcal{U}|+4)N_1^a + \sum_{l=1}^{M_a-1} N_l^a N_{l+1}^a + N_{M_a}^a(|\mathcal{U}|+2) + |\mathcal{U}|+2)$ The meta actor has input dimension $4|\mathcal{U}| + 4|\mathcal{H}|$ and output dimension $|\mathcal{U}| \times |\mathcal{H}|$. With softmax for action selection, the inference complexity is: $((4|\mathcal{U}| + 4|\mathcal{H}|)N_1^a + \sum_{l=1}^{M_a-1} N_l^a N_{l+1}^a + N_{M_a}^a(|\mathcal{U}||\mathcal{H}|) + |\mathcal{U}||\mathcal{H}|)$.

(2) Training Complexity: Training involves forward and backward propagation through both actor and critic networks, constituting the dominant computational overhead. Auxiliary operations (e.g., optimizer updates, target network synchronization, and experience replay) contribute lower-order complexity terms.

By combining TS from all edge policies, the critic input dimension becomes $4|\mathcal{U}| + 4|\mathcal{H}| + |\mathcal{U}||\mathcal{H}| + |\mathcal{U}| + 2|\mathcal{H}|$ with

output dimension 1. The forward propagation complexity per sample is $(((4|\mathcal{U}| + 4|\mathcal{H}|) + |\mathcal{U}||\mathcal{H}| + |\mathcal{U}| + 2|\mathcal{H}|)N_1^c + \sum_{l=1}^{M_c-1} N_l^c N_{l+1}^c + N_{M_c}^c)$ The NoisyLinear layer computational complexity comprises three components: (a) $\mathcal{O}(m+n)$ for factorized Gaussian noise sampling, (b) $\mathcal{O}(mn)$ for applying sampled noise to network weights, and (c) $\mathcal{O}(mn)$ for forward computation. Here, m and n are the output and input dimensions, respectively. The overall asymptotic complexity remains $\mathcal{O}(mn)$, with factorized noise schemes providing computational efficiency through reduced constant factors. Gumbel noise adds $\mathcal{O}(k)$ cost per forward pass, where k is the action dimensionality. Backward propagation exhibits identical asymptotic complexity to forward propagation.

(3) Convergence: Under finite state and action spaces, the multi-agent soft policy iteration scheme, where each agent iteratively updates its policy to the softmax of its current Q-function while keeping other agents' policies fixed, converges to joint policies whose limit points are *soft Nash equilibria* under the shared-reward cooperative setting.

Definition 1 (Soft Nash Equilibrium)**.** *A policy profile $\boldsymbol{\pi}^* = (\pi^*_{\theta_1}, \ldots, \pi^*_{\theta_N})$ is a soft Nash equilibrium if, for all agents i with finite action space $\mathcal{A}_i$:*

$$\pi^*_{\theta_i} \in \arg\max_{\pi_{\theta_i}} \mathcal{J}_i(\pi_{\theta_i} \mid \boldsymbol{\pi}^*_{-i}), \tag{26}$$

where $\mathcal{J}_i$ is the entropy-augmented objective as (21)*. Equivalently [45], for all agent i and $s \in \mathcal{S}$:*

$$\pi^*_{\theta_i}(a_i \mid s) = \frac{\exp\big(\bar{Q}_i^{\boldsymbol{\pi}^*}(s, a_i, \boldsymbol{a}^*_{-i})/\alpha\big)}{\sum_{\tilde{a}_i \in \mathcal{A}_i} \exp\big(\bar{Q}_i^{\boldsymbol{\pi}^*}(s, \tilde{a}_i, \boldsymbol{a}^*_{-i})/\alpha\big)}. \tag{27}$$

Proof. The proof consists of four parts.

i. Contraction of the soft Bellman operator. For a fixed joint policy $\boldsymbol{\pi}$, the soft Bellman operator of agent i is

$$(\mathcal{T}_i^{\boldsymbol{\pi}} Q_i)(s, \mathbf{a}) = r(s, \mathbf{a}) + \gamma\, \mathbb{E}_{s', \mathbf{a}' \sim P(\cdot|s,\mathbf{a}), \boldsymbol{\pi}}[Q_i(s', \mathbf{a}') + \alpha \mathbb{H}(\pi_i(\cdot|s'))],$$

where $P(s'|s, \mathbf{a})$ denotes the state transition probability kernel. For any $Q_i^{(1)}, Q_i^{(2)}$ and any $(s, \mathbf{a})$,

$$|(\mathcal{T}_i^{\boldsymbol{\pi}} Q_i^{(1)})(s, \mathbf{a}) - (\mathcal{T}_i^{\boldsymbol{\pi}} Q_i^{(2)})(s, \mathbf{a})| = \gamma |\mathbb{E}_{s', \mathbf{a}'}[Q_i^{(1)}(s', \mathbf{a}') - Q_i^{(2)}(s', \mathbf{a}')]| \le \gamma \|Q_i^{(1)} - Q_i^{(2)}\|_\infty.$$

Taking the supremum norm yields

$$\|\mathcal{T}_i^{\boldsymbol{\pi}} Q_i^{(1)} - \mathcal{T}_i^{\boldsymbol{\pi}} Q_i^{(2)}\|_\infty \le \gamma \|Q_i^{(1)} - Q_i^{(2)}\|_\infty.$$

Thus $\mathcal{T}_i^{\boldsymbol{\pi}}$ is a γ-contraction and admits a unique fixed point $Q_i^{\boldsymbol{\pi}}$ [46]. Moreover, $Q_i^{t+1} = \mathcal{T}_i^{\boldsymbol{\pi}} Q_i^t$ converges geometrically to $Q_i^{\boldsymbol{\pi}}$ (*soft policy evaluation*) [45].

ii. Soft policy improvement. Define the expected Q-value with respect to other agents' policies as

$$\bar{Q}_i^{\boldsymbol{\pi}}(s, a_i) = \mathbb{E}_{\mathbf{a}_{-i} \sim \pi_{-i}(\cdot|s)}\big[Q_i^{\boldsymbol{\pi}}(s, a_i, \mathbf{a}_{-i})\big].$$

If agent i updates its policy to

$$\pi_i^{\text{new}}(\cdot|s) = \frac{\exp(\bar{Q}_i^{\boldsymbol{\pi}}(s, \cdot)/\alpha)}{\sum \exp(\bar{Q}_i^{\boldsymbol{\pi}}(s, \cdot)/\alpha)},$$

then, holding other agents' policies $\boldsymbol{\pi}_{-i}$ fixed, its entropy-augmented value function is non-decreasing: $\widehat{V}_i^{\pi^{\text{new}}}(s) \ge \widehat{V}_i^{\pi^{\text{old}}}(s), \forall s$. This follows from the soft policy improvement lemma in [45].

iii. Boundedness. Since the rewards are bounded by $R_{\max}$ and the entropy of a discrete action space is bounded by $\log|\mathcal{A}_i|$, for any policy $\boldsymbol{\pi}$ and state s,

$$|\widehat{V}_i^{\boldsymbol{\pi}}(s)| \le \sum_{t=0}^{\infty} \gamma^t (R_{\max} + \alpha \log|\mathcal{A}_i|) = \frac{R_{\max} + \alpha \log|\mathcal{A}_i|}{1-\gamma}.$$

Thus, for each agent i, the entropy-augmented state value function $\widehat{V}_i^{\boldsymbol{\pi}}(s)$ is bounded.

iv. Convergence to equilibrium. Under the shared-reward cooperative setting considered in this work, all agents share the same state value function $\widehat{V}_i^{\boldsymbol{\pi}} = V^{\boldsymbol{\pi}}, \forall i$ when employing a shared critic. With individual critics, each agent maintains its own value estimate incorporating local entropy, yet the shared reward component ensures convergence to the team-optimal solution. Consider a round-robin (asynchronous) update scheme: in step k, only agent i updates its policy to the softmax form (i.e., (27)), others remain fixed. By Step ii, each asynchronous soft policy update ensures $\widehat{V}_i^{\boldsymbol{\pi}^{(k+1)}} \ge \widehat{V}_i^{\boldsymbol{\pi}^{(k)}}$, yielding monotonic improvement of the shared reward component. By Step iii all values are bounded. Hence the value sequence converges. Since the joint policy space $\Pi = \prod_{i,s} \Delta(\mathcal{A}_i)$ is a finite product of simplices, it is compact. Therefore $\{\boldsymbol{\pi}^{(k)}\}$ has at least one convergent subsequence. Let $\boldsymbol{\pi}^*$ be any limit point. By continuity of the softmax operator and uniqueness of the soft policy evaluation fixed point, we obtain for all i and s:

$$\pi_i^*(a_i|s) = \frac{\exp(\bar{Q}_i^{\boldsymbol{\pi}^*}(s, a_i)/\alpha)}{\sum_{\tilde{a}_i \in \mathcal{A}_i} \exp(\bar{Q}_i^{\boldsymbol{\pi}^*}(s, \tilde{a}_i)/\alpha)}.$$

That is, π_i^* is the best (entropy-regularized) response to π^*_{-i}. Thus, $\boldsymbol{\pi}^*$ is a soft Nash equilibrium. □

This ensures convergence to a stable solution, while entropy regularization alleviates the risk of premature convergence.

V. Performance Evaluation

To evaluate the proposed approach, we conduct simulations using PyTorch 2.5 with CUDA 12 on a server equipped with an NVIDIA RTX 4070 Ti GPU, and compare it against the following benchmarks:

- **Power GA:** Since the TS at each ES is intractable using conventional optimization methods, we adopt a greedy heuristic algorithm (GA) that minimizes the combined consumption of computation and cooling energy. Under this setting, the TS problem reduces to a knapsack-variant problem without considering the activation energy consumption [44]. To promote renewable energy utilization, the system prioritizes switching to renewable energy at each decision point.
- **Local Edge+GA:** This baseline allocates tasks to their associated local ESs without inter-ES offloading. Subsequently, the power GA algorithm is applied to determine TS and PS decisions.

- **TD3/DSAC+GA**: Twin-Delayed Deep Deterministic Policy Gradient (TD3) and Soft Actor-Critic (SAC) in discrete action settings [45], [47] are applied for TA, combined with the power GA algorithm for TS and PS.
- **MATD3**: This is an enhanced MADDPG with dual critics and soft delayed updates. We also evaluate the MATD3_Noisy, which incorporates parameter-space noise for exploration.
- **HDRL**: The hierarchical DRL approach in [36] employs specialized exploration and action-generation strategies, with agents trained solely on local experiences.

Since the problem involves large discrete action spaces, value-based DRL methods are not applicable, and therefore, we do not include comparisons with them. Besides, purely single-agent DRL is not included as a baseline because constructing a joint action space that satisfies each edge server's local constraints requires prior knowledge that is often unavailable in practice. Moreover, a fair comparison would require a deeper policy network, resulting in substantially increased training time and memory consumption.

A. Simulation Setup

We consider a system with five ESs, where the pairwise transmission rates are uniformly sampled from $[50, 80]$ Mbps. In addition, we introduce $\mathcal{W}^{\text{up}}$ and $\mathcal{W}^{\text{s}} \in \mathbb{R}^+$ to dynamically scale the RESS capacity and the power switching cost, respectively. Following standard practice, we neglect transmission energy, as its contribution is negligible compared with computation energy. The configurations of these ESs are randomly generated and summarized in TABLE II.

TABLE II: Simulation Settings for Deployed Edge Servers

f_h	$[3.5, 3.5, 3.5, 3.5, 4.0]$ GHz
κ^h	$[2.2, 3.4, 4.4, 5.6, 4.8]$
f_h^{ipc}	$[3, 3, 3, 3, 4]$/cycle
ζ_h	$[0.2, 0.3, 0.5, 0.45, 0.35]$
f_h^{flop}	$[4, 8, 8, 16, 16]$/instruction
E_h^{switch}	$[40, 60, 60, 35, 35] \times W^s$ J
P_h^{active}	$[8, 10, 13, 15, 12] \times 10$ J/s
$\mathbf{E}_h^{\mathbf{green}}$,	$[200, 200, 300, 400, 400] \times W^{\text{up}}$ J

We adopt the Beta distribution to model the ratio of available renewable energy,enabling the simulation of diverse and extreme scenarios. Five representative levels of renewable energy availability are generated, as illustrated in Fig. 3. These scenarios capture the uneven distribution of renewable energy across different load levels, time periods, and geographic locations.

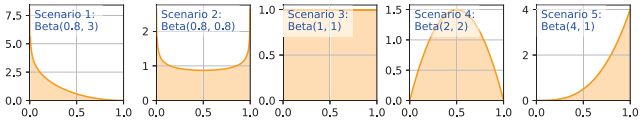


Fig. 3: Illustration of parameters for selected scenarios.

At each time slot, task requests, available renewable energy, coolant temperature, target input air temperature, and current power supply state are reinitialized as random values drawn from uniform distributions (see Table III). During initialization, $\rho_t^u \sim \text{U}[500, 800]$ FLOPs/bit and $\eta_u^h \sim \text{U}[0.3, 6]$ are sampled once and remain fixed throughout all simulations. For the cloud server, we set $\text{CoP}_{\text{cloud}} = 6$, $\kappa_{\text{cloud}} = 5$, $f_{\text{cloud}}^{ipc} = 4$, $f_{\text{cloud}}^{flop} = 4$/instruction, $f_{\text{cloud}} = 4$GHz. The cloud server is permanently activated, and its activation power is excluded from energy analysis. To ensure the feasibility, each task's latency constraint is lower bounded as $\tau_t^u = \max(\tau_t^u, T_{H_t^u}^p + 100)$ ms, where $T_{H_t^u}^p$ denotes the estimated processing delay at its associated ES.

TABLE III: Simulation Parameters (Par) for Task Features

Par	Value	Par	Value
V_t^u	[1000, 1500] Kbyte	R_t^u	[10, 20] Kbyte
τ_t^u	[800, 1000] ms	$\mathbb{T}_{h,t}^{in}$	$[5, 10]^\circ$C
$\mathbb{T}_{h,t}^c$	$[10, 15]^\circ$C	ρ_t^u	$[500, 800]$ FLOPs/bit

B. Simulation results

Training convergence: First, we evaluate the convergence of the proposed approach under Scenario 3 with $|\mathcal{U}| = 100$. An extreme setting is considered with $\mathcal{W}^s = 15$ and $\mathcal{W}^{up} = 3$, which incurs high power switching costs while renewable energy can only occasionally satisfy system demands. Results are displayed using a moving average with a window size of 100 episodes. In each episode, 30 experiences are collected before policy updates, followed by 10 update rounds. To improve training stability, the AutoClip technique [48] is applied with a percentile threshold of 10. Fig. 4 presents the convergence behavior under different hyperparameter settings. The results report the average conventional energy consumption ('Trad. power cost (Avg.) / Wh') computed using (11), where 1 Wh = 3600 J.

Fig. 4a examines the impact of discount factor γ. Larger γ lead to improved convergence by assigning greater weight to future rewards, thereby strengthening the influence of the target network and stabilizing training in complex multi-agent environments. Fig. 4b shows the effect of the learning rate (lr). Given the large state-action space and multi-agent heterogeneity, learning rates below 10^{-4} are evaluated. The results indicates that $lr = 3 \times 10^{-5}$ achieves superior convergence, striking a balance between fine-grained exploration and training efficiency. Fig. 4c shows that larger batch sizes improve convergence due to more accurate gradient estimates. However, overly large batches are discouraged because of increased cost and diminished exploration. Overall, these results demonstrate the convergence of the proposed algorithm. The selected hyper-parameters for subsequent experiments are summarized in Table IV. The entropy temperature for TA is learnable and updated following the SAC guidelines [45], [47].

Evaluation of training dynamics We compare the training processes of DRL-based approaches under the above configurations. Fig. 5 presents the results smoothed using a moving average with a window size of 500 episodes.

As illustrated in Fig. 5, TD3+GA performs poorly under the given configuration, whereas incorporating action entropy

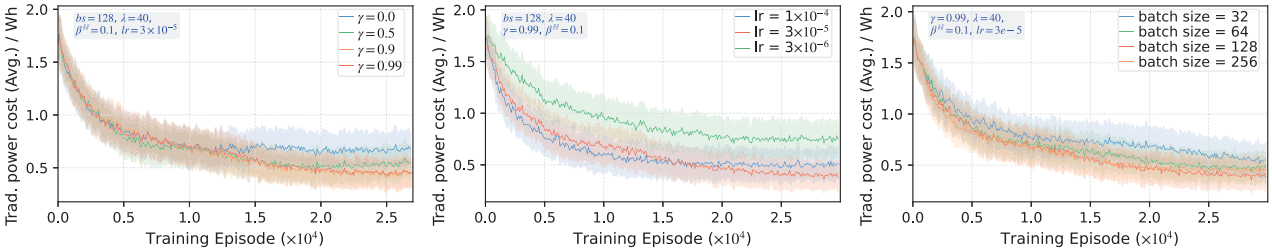


Fig. 4: Convergence *vs.* hyper-parameters. (a) discount factor. (b) learning rate. (c) batch size.

TABLE IV: Simulation Parameters for Policy Training

Configuration	**Value**
Hidden layers of actor	[512, 256, 256, 256]
Hidden layers of critic	[512, 256, 256, 256, 256]
Activation layer	LeakyReLu
Replay Buffer Capacity	6000
batch size	128
Optimizer	Adam
learning rate: actor, critic,α	$\{3, 9, 6\} \times 10^{-5}$
Discount factor γ	0.9
Entropy temperature α_i	TA: learning, {TS,PS}:0.1
Delay update frequency λ	40
soft update μ	0.001

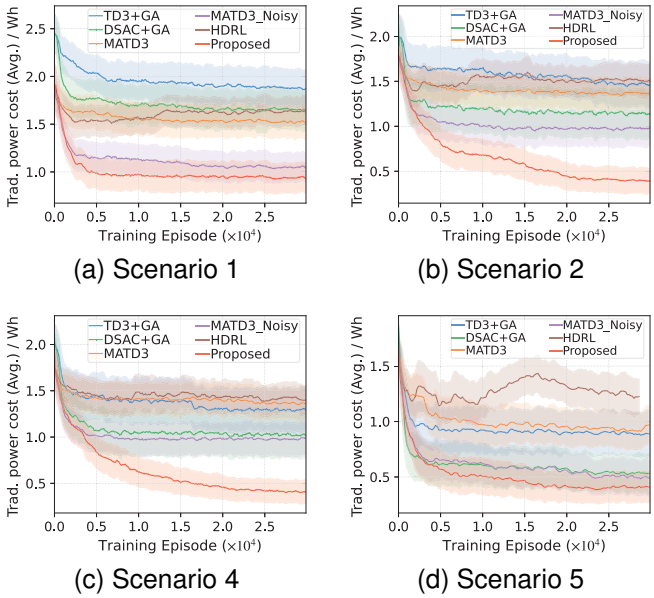


Fig. 5: Training comparison of DRL-based approaches.

(DSAC+GA) significantly enhances exploration and improves performance. However, DSAC+GA exhibits substantial performance variability across different renewable energy distributions. This is mainly because GA neglects renewable energy availability, a critical factor whose influence depends on the underlying distribution. Multi-agent approaches (HDRL and MATD3) do not consistently outperform single-agent baselines. HDRL is fundamentally limited by learning state–action values from local observations under a global reward signal, which can map identical local states to different rewards due to the joint dependence on all agents' actions. This coupling induces environment non-stationarity and degrades training stability. Parameter-space noise injection improves exploration by encouraging diverse experience collection, leading to more accurate value estimation and more stable policy updates. Consequently, the approach alleviate premature convergence and fosters better agent coordination. Action entropy further improves exploration efficiency.

Across scenarios, several consistent patterns can be observed. In scenarios with complex power management (Scenarios 2 and 4), where ESs experience heterogeneous renewable availability, policy convergence is slower due to the enlarged action space jointly involving task allocation and power switching. Under scarce renewable energy (Scenario 1), energy-aware methods (MATD3_Noisy and our approach) significantly outperform other baselines. In contrast, under abundant renewable energy (Scenario 5), their advantage over DSAC+GA becomes marginal, as most ESs can primarily rely on renewable sources. Our approach achieves comparable performance in Scenarios 2, 4, and 5, mainly attributed to task concurrency constraints at ESs. Even with sufficient renewable energy, task concurrency limits at each ES force excess tasks to be offloaded to nodes with insufficient renewable energy, thereby increasing conventional energy consumption. Moreover, random initialization of power supply states induces frequent power switching, making a non-negligible amount of conventional energy consumption unavoidable.

Impact of RESS capacity: The capacity of the RESS critically determines renewable energy availability, thereby influencing TA decisions and system sustainability. To investigate this effect, we vary the capacity factor $\mathcal{W}^{up}$, with the results shown in Fig. 6. To mitigate the impact of task concurrency constraints, the maximum number of concurrent tasks per ES is doubled compared with previous simulations. Simulations are conducted under Scenario 2 with $\mathcal{W}^s = 15$ and $|\mathcal{U}| = 90$, and results are averaged over the final 100 out of 30,000 training episodes.

As shown in Fig. 6a, conventional energy consumption decreases with increasing RESS capacity due to improved renewable energy availability. Inter-ES collaboration further enhances system sustainability by exploiting spatially dis-

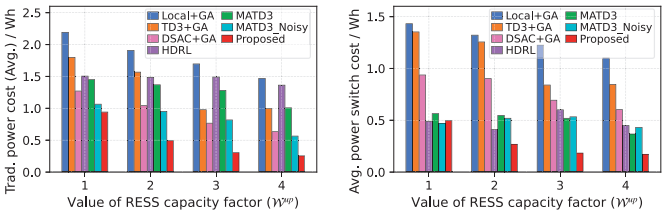


Fig. 6: Impact of RESS capacity.

tributed renewable resources, thereby reducing reliance on conventional energy compared with systems without task allocation. However, multi-agent methods with limited exploration capability (HDRL and MATD3) yield only marginal and inconsistent gains over centralized DRL baselines with GA power management (TD3+GA and DSAC+GA). In contrast, introducing parameter-space noise significantly improves MATD3 performance. Overall, the proposed algorithm consistently achieves the lowest conventional energy consumption across all scenarios, demonstrating superior sustainability. Fig. 6b shows that power switching energy consumption decreases as RESS capacity increases, since higher renewable availability reduces the frequency of switching to conventional sources. Compared with TD3, DSAC enables more rational TA, mitigating renewable energy exhaustion at ESs and thereby reducing switching overhead. Approaches that jointly consider power management costs and renewable energy availability are more effective in reducing switching-related energy consumption. Accordingly, the proposed approach achieves the lowest power switching costs in most cases and enhances overall sustainability.

Impact of power switching cost: Since power switching costs significantly impact both conventional energy consumption and power management decisions, we investigate their effect by varying $\mathcal{W}^s$. Fig. 7 presents simulation results under Scenario 4 with $\mathcal{W}^{up} = 2.5$, while other configurations remain consistent with previous simulations.

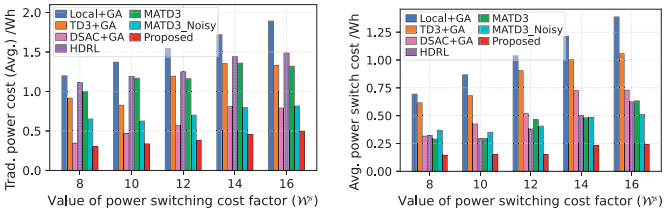


Fig. 7: Impact of power switching cost.

Fig. 7a shows that conventional energy cost generally increases with power switching costs. TD3+GA, MATD3, and HDRL exhibit limited improvement over the local+GA. In contrast, DSAC+GA and MATD3_Noisy significantly improve system sustainability through enhanced exploration. The advantage of DSAC+GA mainly stems from more effective TA, which mitigates the impact of power management decisions. When $\mathcal{W}^s < 12$, DSAC+GA achieves lower conventional energy consumption than MATD3_Noisy, since the benefits of explicit power management diminish as switching costs decrease. Our proposed algorithm consistently achieves the lowest conventional energy consumption across all $\mathcal{W}^s$ settings, outperforming all benchmarks. Fig. 7b shows that higher per-operation switching costs result in increased total switching costs. Nevertheless, our approach effectively minimizes switching costs under all evaluated conditions. These results demonstrate that jointly optimizing task allocation and power management is crucial for achieving higher system sustainability.

Impact of application number: Since the number of applications directly affects system resource utilization, we evaluate performance under different application loads. All approaches are evaluated under Scenario 4 with $\mathcal{W}^{up} = 2.5$ and $\mathcal{W}^s = 13$. The results are shown in Fig. 8.

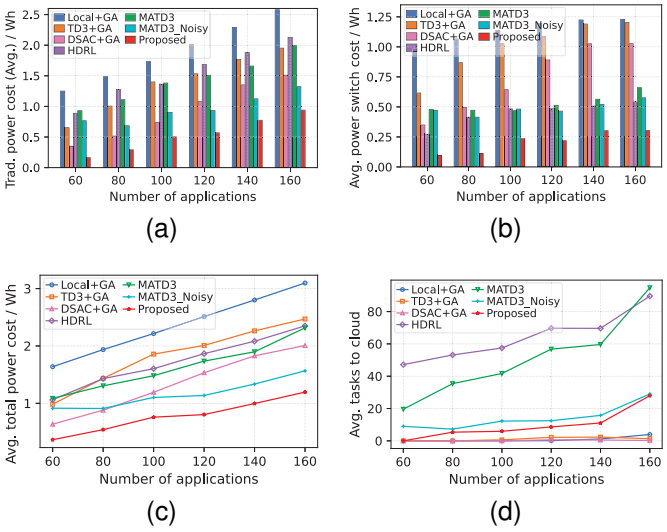


Fig. 8: Impact of application number.

The results show that both conventional energy consumption and power switching costs increased as application number increase. As the workload grows, more ESs are activated and handle additional tasks, increasing the likelihood of renewable energy depletion and subsequent switching to conventional sources. Our proposed method consistently outperforms DSAC+GA, with the performance gap widening as application numbers increase. This trend highlights the growing importance of effective power management under larger and more complex action spaces, particularly when frequent power switching incurs high costs. Fig. 8c shows that total energy consumption increases with the number of applications, while our approach minimizes total energy consumption compared with other benchmarks. Fig. 8d illustrates cloud offloading behavior of different approaches. Approaches with greedy power management (Local+GA, TD3+GA, DSAC+GA) exhibit minimal cloud offloading, as decisions are based solely on per-task energy comparison. Since edge processing generally incurs lower energy costs than cloud execution in our setup, these approaches consistently prefer local processing

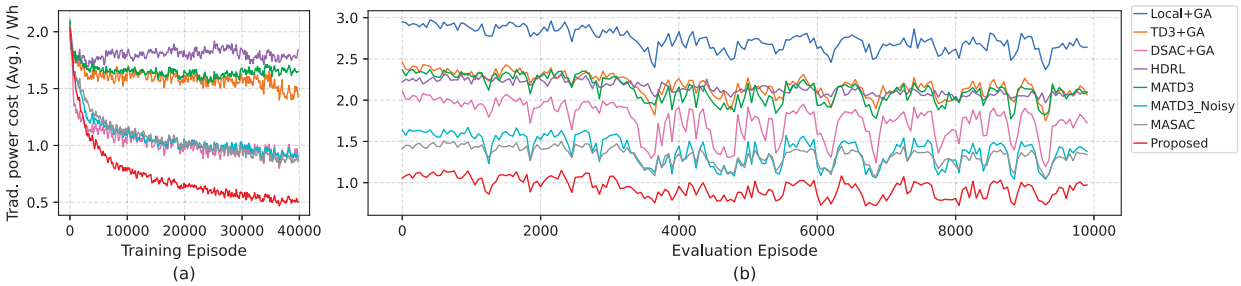


Fig. 9: Performance evaluation based on real-world data.

whenever capacity permits. In contrast, AI-based methods that jointly optimize task selection and power management exhibit strategic cloud offloading. Our proposed method minimizes cloud offloading while maintaining performance, reflecting holistic sustainability optimization. For example, a task may be offloaded to cloud when edge execution would trigger power switching due to renewable energy depletion. Despite this increase per-task energy cost, it avoids expensive switching operations and improves overall sustainability. Notably, MATD3 and HDRL exhibit significantly higher cloud offloading, mainly due to suboptimal TA actions that violates edge constraints (e.g., QoS and capacity), leading to excessive cloud migration. This behavior indicates insufficient coordination and exploration, which degrades multi-agent cooperation.

Evaluation on real renewable energy generation dataset: To further evaluate robustness and generalization, we conduct additional evaluations using real-world renewable energy generation datasets. To enable adaptation to diverse energy distributions, the policy is trained under randomized Beta distributions, where both parameters are uniformly sampled from $[0.2, 5]$ at each observation. Training is performed over 40,000 episodes using synthetic data, and then evaluated over 10,000 episodes using real-world datasets. We collected publicly available energy generation data, including wind energy records from four German transmission system operators (50Hertz, Amprion, TenneT TSO, and TransnetBW)[1][2] and solar power generation data from 2021 and 2022 obtained from the Sheffield Solar Platform[3]. From each dataset, the first 10,000 records are extracted and normalized by their maximum values, representing the relative availability of renewable energy. These normalized traces are mapped to different ESs: ES 1 uses TenneT TSO data (wind), ES 2 uses TransnetBW data (wind), ES 3 uses Amprion data (wind), ES 4 uses Solar PV (2021) data, and ES 5 uses Solar PV (2022) data. We set $|\mathcal{U}| = 150$, $\mathcal{W}^{up} = 4$, and $\mathcal{W}^{s} = 13$. Fig. 9 illustrates both training and evaluation performance, with results smoothed over 200 training episodes and 50 evaluation episodes. In addition, we include MASAC algorithm [42] as a baseline, which incorporates action entropy but does not employ parameter-space noise.

Fig. 9 presents the average conventional energy cost cost during training (Fig. 9a) and evaluation (Fig. 9b). During training, the proposed method achieves significant and consistent energy cost reduction, exhibiting faster convergence and higher learning efficiency than all baselines. In contrast, other benchmarks converge to noticeably higher cost levels, indicating limited adaptability to renewable-aware environments. During evaluation on real-world datasets, the proposed method consistently attains the lowest average conventional energy consumption across over 10,000 episodes, demonstrating strong generalization capability. It also shows smaller fluctuations, suggesting improved robustness under noisy and non-stationary renewable energy conditions. The persistently high cost of Local+GA further highlights the necessity of coordinated task and power management in MEC systems with heterogeneous and time-varying renewable availability.

These results demonstrate that the proposed approach can learn robust and sustainability-aware policies for distributed green MEC systems under realistic dynamics of renewable. A key insight is that heterogeneous MADRL enables the joint optimization of intrinsically coupled subproblems in an E2E manner. Such coordinated optimization relies on efficient exploration, for which parameter-space noise provides an effective complement to action entropy, particularly in ultra-large-scale discrete action spaces where conventional exploration strategies face scalability and convergence challenges.

VI. Conclusion and Discussion

This paper investigated distributed green MEC systems powered by hybrid energy sources. To promote system sustainability, we formulated a joint optimization problem that minimizes conventional energy consumption through coordinated task allocation and power switching. Due to edge infrastructure constraints and strong operational coupling, the problem is computationally intractable and involves high-dimensional, interdependent action spaces. To address these challenges, we proposed an E2E AI-driven approach based on heterogeneous cooperative MADRL, where agents independently solve local subproblems while collectively optimizing a global objective. In addition, individualized entropy regularization and parameter-space noise were incorporated to enhance exploration efficiency in ultra-large discrete action spaces. Theoretical analysis proves the convergence of the proposed approach to a soft Nash equilibrium. Extensive simulations demonstrate that the proposed approach significantly improves system sustainability compared with baseline methods. Meanwhile, the algorithm exhibits superior exploration capability

[1]https://www.netztransparenz.de/

[2]https://www.kaggle.com/datasets/jorgesandoval/wind-power-generation?utm_source=chatgpt.com

[3]https://www.solar.sheffield.ac.uk/pvlive/

and enhances learning stability in heterogeneous MADRL settings with large discrete action spaces, thereby validating both its theoretical soundness and practical effectiveness.

This work focused on optimizing single decision snapshots under a short time-horizon assumption, where tasks are completed promptly. As a result, long-term continuous task execution and queue dynamics are beyond the scope of this study. Future work will extend the framework to long-horizon system management in more practical MEC settings. In particular, incorporating system states such as task execution queues would enable queue-aware scheduling and more realistic modeling of traffic-intensive applications. Moreover, integrating observations and predictions of uncertain renewable energy generation can further improve adaptive and efficient renewable energy utilization. Investigating the dynamic trade-off between sustainability and QoS under queue dynamics and time-varying QoS requirements remains a promising research direction. Collectively, these extensions provide a foundation for resilient and autonomous off-grid MEC systems.

REFERENCES

[1] S. Li, S. Li, Y. Sun, B. Wang, B. Wang, and B. Zhang, "Digital twin-assisted computation offloading and resource allocation for multi-device collaborative tasks in industrial internet of things," *IEEE Transactions on Network Science and Engineering*, vol. 13, pp. 581–596, 2026.

[2] Y. Mao, X. Yu, K. Huang, Y.-J. Angela Zhang, and J. Zhang, "Green edge ai: A contemporary survey," *Proceedings of the IEEE*, vol. 112, no. 7, pp. 880–911, 2024.

[3] V. Bolón-Canedo, L. Morán-Fernández, B. Cancela, and A. Alonso-Betanzos, "A review of green artificial intelligence: Towards a more sustainable future," *Neurocomputing*, vol. 599, p. 128096, 2024.

[4] C.-C. J. Kuo and A. M. Madni, "Green learning: Introduction, examples and outlook," *Journal of Visual Communication and Image Representation*, vol. 90, p. 103685, 2023.

[5] A. Tabbakh, L. Al Amin, M. Islam, G. M. I. Mahmud, I. K. Chowdhury, and M. S. H. Mukta, "Towards sustainable AI: a comprehensive framework for green AI," *Discover Sustainability*, vol. 5, no. 1, p. 408, Nov. 2024.

[6] M. K. Mondal, S. Banerjee, D. Das, U. Ghosh, M. S. Al-Numay, and U. Biswas, "Toward energy-efficient and cost-effective task offloading in mobile edge computing for intelligent surveillance systems," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 4087–4094, 2024.

[7] Z. Lin, J. Yang, C. Wu, and P. Chen, "Energy-efficient task offloading for distributed edge computing in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 9, pp. 14 056–14 061, 2024.

[8] S. Ma, Y. Liu, Y. Liu, J. Wang, Q. Fang, and Y. Huang, "Artificial intelligence-enabled predictive energy saving planning of liquid cooling system for data centers," *Advanced Engineering Informatics*, vol. 65, p. 103283, 2025.

[9] B. R. Park, Y. J. Choi, E. J. Choi, and J. W. Moon, "Adaptive control algorithm with a retraining technique to predict the optimal amount of chilled water in a data center cooling system," *Journal of Building Engineering*, vol. 50, p. 104167, 2022.

[10] C. Fang, X. Meng, and et al., "DRL-based green task offloading for content distribution in NOMA-enabled cloud-edge-end cooperation environments," in *ICC 2023 - IEEE International Conference on Communications*. IEEE, 2023, pp. 6126–6131.

[11] H. Ma, P. Huang, Z. Zhou, X. Zhang, and X. Chen, "GreenEdge: Joint green energy scheduling and dynamic task offloading in multi-tier edge computing systems," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4322–4335, 2022.

[12] Y. Yu, Z. Zhao, Y. Zhao, Y. Zhong, I. Humar, and X. Ge, "Carbon-efficiency optimization in cellular networks with a hybrid-energy supply," *IEEE Transactions on Green Communications and Networking*, vol. 10, pp. 1462–1477, 2026.

[13] Y. Lin, Y. Zhao, Y. Zhong, X. Ge, and I. Humar, "Carbon-aware modeling and optimization of renewable-energy cellular networks based on the exergy-balance theory," *IEEE Transactions on Vehicular Technology*, pp. 1–16, 2025.

[14] Y. Xiao, Y. Song, and J. Liu, "Collaborative multi-agent deep reinforcement learning for energy-efficient resource allocation in heterogeneous mobile edge computing networks," *IEEE Transactions on Wireless Communications*, vol. 23, no. 6, pp. 6653–6668, 2024.

[15] Y. Li, A. S. Madhukumar, T. Z. H. Ernest, G. Zheng, W. Saad, and A. Hamid Aghvami, "Energy-efficient UAV-driven multi-access edge computing: A distributed many-agent perspective," *IEEE Transactions on Communications*, pp. 1–1, 2025.

[16] Y. Chen, K. Li, Y. Wu, J. Huang, and L. Zhao, "Energy efficient task offloading and resource allocation in air-ground integrated mec systems: A distributed online approach," *IEEE Transactions on Mobile Computing*, vol. 23, no. 8, pp. 8129–8142, 2024.

[17] Y. Luo, L. Pu, and C.-H. Liu, "Computing power and battery charging management for solar energy powered edge computing," *IEEE Transactions on Mobile Computing*, vol. 24, no. 3, pp. 1913–1927, 2025.

[18] Y. Chen, Y. Sun, and et al, "Joint task and computing resource allocation in distributed edge computing systems via multi-agent deep reinforcement learning," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 4, pp. 3479–3494, 2024.

[19] X. Xie, Y. Han, and H. Tan, "Greening china's digital economy: exploring the contribution of the east–west computing resources transmission project to co2 reduction," *Humanities and Social Sciences Communications*, vol. 11, no. 1, p. 466, Mar. 2024.

[20] X. Chen, Z. Lu, and et al, "Cooling-aware optimization of edge server configuration and edge computation offloading for wirelessly powered devices," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 5, pp. 5043–5056, 2021.

[21] K. M. U. Ahmed, M. H. J. Bollen, and M. Alvarez, "A review of data centers energy consumption and reliability modeling," *IEEE Access*, vol. 9, pp. 152 536–152 563, 2021.

[22] K. Peng, C. Ling, S. Wang, and V. C. Leung, "An online computation offloading approach with dual stability guarantee for heterogeneous tasks in mec-enabled iiot," *IEEE Transactions on Mobile Computing*, pp. 1–12, 2025.

[23] S. Lei, H. Tang, C. Li, X. Zhang, C. Xu, and H. Wu, "Federated maddpg-based collaborative scheduling strategy in vehicular edge computing," *IEEE Transactions on Mobile Computing*, pp. 1–13, 2025.

[24] C. Tang, Y. Ding, S. Xiao, Z. Huang, and H. Wu, "Collaborative service caching, task offloading, and resource allocation in caching-assisted mobile edge computing," *IEEE Transactions on Services Computing*, vol. 18, no. 4, pp. 1966–1981, 2025.

[25] P. Qin, Y. Fu, G. Tang, X. Zhao, and S. Geng, "Learning based energy efficient task offloading for vehicular collaborative edge computing," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 8, pp. 8398–8413, 2022.

[26] H. Ma, Z. Huang, G. Lu, D. Fu, and C. Shi, "Greening edge AI: Optimizing inference accuracy and reducing carbon emissions with renewable energy," *IEEE Internet of Things Journal*, pp. 1–1, 2025.

[27] H. Liao, G. Tang, D. Guo, Y. Wang, and R. Cao, "Rethinking low-carbon edge computing system design with renewable energy sharing," in *Proceedings of the 53rd International Conference on Parallel Processing*. ACM, 2024, pp. 950–960.

[28] Q. Pei, Y. Yuan, H. Hu, L. Wang, D. Zhang, B. Yan, C. Yu, and F. Liu, "Working smarter not harder: Hybrid cooling for deep learning in edge datacenters," *IEEE Transactions on Sustainable Computing*, pp. 1–16, 2025.

[29] T. Li, K. Zhu, N. C. Luong, D. Niyato, Q. Wu, Y. Zhang, and B. Chen, "Applications of multi-agent reinforcement learning in future internet: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 1240–1279, 2022.

[30] X. Liu, J. Yu, Z. Feng, and Y. Gao, "Multi-agent reinforcement learning for resource allocation in iot networks with edge computing," *China Communications*, vol. 17, no. 9, pp. 220–236, 2020.

[31] Y. Zhang, B. Di, Z. Zheng, J. Lin, and L. Song, "Distributed multi-cloud multi-access edge computing by multi-agent reinforcement learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2565–2578, 2021.

[32] Z. Gao, L. Yang, and Y. Dai, "Large-scale computation offloading using a multi-agent reinforcement learning in heterogeneous multi-access edge computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 6, pp. 3425–3443, Jun. 2023.

[33] P. Liu, K. An, and et al., "Computation rate maximization for scma-aided edge computing in iot networks: A multi-agent reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 23, no. 8, pp. 10 414–10 429, 2024.

[34] C. Zeng, X. Wang, R. Zeng, Y. Li, J. Shi, and M. Huang, "Joint optimization of multi-dimensional resource allocation and task offloading for

QoE enhancement in cloud-edge-end collaboration," *Future Generation Computer Systems*, vol. 155, pp. 121–131, 2024.
[35] H. Yu, S. Leng, and F. Wu, "Joint cooperative computation offloading and trajectory optimization in heterogeneous UAV-swarm-enabled aerial edge computing networks," *IEEE Internet of Things Journal*, vol. 11, no. 10, pp. 17 700–17 711, May 2024.
[36] C. Sun, X. Li, and et al., "Hierarchical deep reinforcement learning for joint service caching and computation offloading in mobile edge-cloud computing," *IEEE Transactions on Services Computing*, vol. 17, no. 4, pp. 1548–1564, 2024.
[37] Z. He, Y. Sun, B. Zhang, K. Dong, B. Wang, and H. Xu, "Qoi-aware configuration adaptation and heterogeneous resource allocation for edge video analytics," *IEEE Internet of Things Journal*, vol. 12, no. 12, pp. 19 215–19 230, 2025.
[38] S. M. M. Nejad, G. Badawy, and D. G. Down, "Eawa: Energy-aware workload assignment in data centers," in *2018 International Conference on High Performance Computing & Simulation (HPCS)*, 2018, pp. 260–267.
[39] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, 2013.
[40] Y. A. Çengel, M. A. Boles, and M. Kanoglu, *Thermodynamics: An Engineering Approach*, 9th ed. McGraw-Hill Education, 2019.
[41] P. Zhang, B. Wang, W. Wu, W. Shi, and X. Li, "Heat recovery from internet data centers for space heating based on an integrated air conditioner with thermosyphon," *Renewable Energy*, vol. 80, pp. 396–406, 2015.
[42] J. Liu, Y. Zhong, S. Hu, H. Fu, Q. FU, X. Chang, and Y. Yang, "Maximum entropy heterogeneous-agent reinforcement learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Vienna, Austria, April 2024.
[43] M. Fortunato, M. G. Azar, and et al., "Noisy networks for exploration," in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, April 30–May 3 2018.
[44] Y. Chen, Y. Sun, C. Wang, and T. Taleb, "Dynamic task allocation and service migration in edge-cloud iot system based on deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 16 742–16 757, 2022.
[45] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*. Stockholm, Sweden: PMLR, 2018, pp. 1861–1870.
[46] E. Kreyszig, *Introductory Functional Analysis with Applications*, 1st ed. Wiley, 1978.
[47] P. Christodoulou, "Soft actor-critic for discrete action settings," 2019. [Online]. Available: https://arxiv.org/abs/1910.07207
[48] P. Seetharaman, G. Wichern, B. Pardo, and J. L. Roux, "Autoclip: Adaptive gradient clipping for source separation networks," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2020, pp. 1–6.

Yan Chen received the B.E. degree in Information Engineering and the Ph.D. degree in Information and Communication Engineering from China University of Mining and Technology, China, in 2016 and 2022. He is currently a Postdoctoral Researcher with Ruhr University Bochum, Bochum, Germany. He is also a senior researcher with the ICTFICIAL Oy, Espoo, Finland. From December 2020 to December 2021, he was also a visiting Ph.D. student at Aalto University, Finland. His research interests include Edge Computing, Internet of Things, Network Intelligence, and Multi-agent systems.

Tarik Taleb (Senior Member, IEEE) received the B.E. degree (with distinction) in Information Engineering and the M.Sc. and Ph.D. degrees in Information Sciences from Tohoku University in 2001, 2003, and 2005, respectively. He is currently a Full Professor at Ruhr University Bochum, where he heads the Institute of Networked Energy-Efficient Systems. Previously, he was a Professor with the Center for Wireless Communications at the University of Oulu. From October 2014 to December 2021, he served as an Associate Professor with the School of Electrical Engineering at Aalto University. Prior to joining Aalto University, he worked as a Senior Researcher and 3GPP Standards Expert with NEC Europe Ltd.. Until March 2009, he was an Assistant Professor with the Graduate School of Information Sciences at Tohoku University, where he worked in a laboratory fully funded by KDDI. From 2005 to 2006, he was a Research Fellow with the Intelligent Cosmos Research Institute. He was directly involved in the development and standardization of the Evolved Packet System as a member of the 3GPP System Architecture Working Group. His current research interests include AI-based network management, autonomous edge–cloud continuum management, architectural enhancements to mobile core networks, network softwarization and slicing, software-defined security, and mobile multimedia streaming.

Qize Guo received the Ph.D. degree in Information and Communication Engineering from the Beijing University of Posts and Telecommunications (BUPT), China, in 2023. From 2018 to 2019, he was a visiting Ph.D. student at the University of Texas at Dallas, USA, where he conducted research in advanced communication networks. He is currently a Postdoctoral Researcher at Ruhr University Bochum, Germany, and also serves as a Senior Researcher at ICTFICIAL Oy, Espoo, Finland. His research interests span a broad range of topics in communication and networking, including transport networks, the Internet of Things (IoT), network efficiency optimization, and multi-agent systems.

Ignacio Lacalle Úbeda is a Researcher and Teaching Assistant at the Universitat Politècnica de València (UPV), a public University in the South-East of Spain. Ignacio is a Telecommunications Engineer (2014) from UPV and received his Ph.D. (Dr. Ing.) in 2022. The expertise of Ignacio is mainly rooted in the Internet of Things field, having participated in 10 national and EU research projects such as HE aerOS, MSCA AIAS, HE MISSION, H2020 ASSIST-IoT and others, related mainly to distributed systems, interoperability, added value services, data processing, cybersecurity and manageability, inter alia. Ignacio has performed various roles in those projects (ranging from Developer to Community Manager or Project Manager). In addition, most of those projects focused on applying IoT-related innovations, particularly in the fields of energy, maritime ports, manufacturing, agriculture, and others.

Tarik Zakaria Benmerar received the Engineering degree in computer engineering, the master's degree in networks and distributed systems, and the Ph.D. degree in parallelism and cloud computing (SaaS) applied for cerebral connectivity using diffusion MRI from the University of Science and Technology Houari Boumediene (USTHB), Algiers, Algeria, in 2010, 2011, and 2019, respectively. He is currently an Associate Professor (HDR) with USTHB and a Research Engineer with ICTIFICIAL Oy. His current research interests include cloud computing/edge/IoT architecture and orchestration, WebRTC-based streaming architectures, and web browser-based parallelism.