Scalable Multi-Agent Reinforcement Learning for Residential Load Scheduling under Data Governance

Zhaoming Qin, *Graduate Student Member, IEEE*, Nanqing Dong, Di Liu, Zhefan Wang, and Junwei Cao, *Senior Member, IEEE*

***Abstract*—As a data-driven approach, multi-agent reinforcement learning (MARL) has made remarkable advances in solving cooperative residential load scheduling problems. However, centralized training, the most common paradigm for MARL, limits large-scale deployment in communication-constrained cloud-edge environments. As a remedy, distributed training shows unparalleled advantages in real-world applications but still faces challenge with system scalability, *e.g.*, the high cost of communication overhead during coordinating individual agents, and needs to comply with data governance in terms of privacy. In this work, we propose a novel MARL solution to address these two practical issues. Our proposed approach is based on actor-critic methods, where the global critic is a learned function of individual critics computed solely based on local observations of households. This scheme preserves household privacy completely and significantly reduces communication cost. Simulation experiments demonstrate that the proposed framework achieves comparable performance to the state-of-the-art actor-critic framework without data governance and communication constraints.**

I. Introduction

As ultimate consumers in the electricity transmission chain, residential loads account for nearly 40% of total electricity consumption in the developed countries (*e.g.*, about 38.4% in the U.S. in 2022 [1]). The large flexibility of residential loads provides a great potential for energy regulation and scheduling, promoting the vigorous development of smart homes [2]. By integrating multiple smart homes, the residential microgrid can aggregate the capacity of load scheduling and reduce the total energy costs. So far, the load scheduling of the residential microgrid has gained increasing attention [3]–[5].

In recent years, the breakthroughs in multi-agent reinforcement learning (MARL) have led to new solutions to the load scheduling problem [6]. First, the residential microgrid with multiple households is naturally modeled as a multi-agent environment where each household is regarded as an agent. Second, without any prior knowledge of the residential microgrid, model-free reinforcement learning (RL) techniques can learn practical policies by interacting with the environment and then perform real-time execution based on the learned policies [7]. Third, the emerging cloud-edge computing structure provides an ideal physical implementation for MARL [8].

Parallel to the design of MARL algorithms, another aspect to be considered is data governance, which is a collection of processes, policies, standards, and metrics that ensure the effective and efficient use of load scheduling data. Although the massive effort has been dedicated to developing the cooperative load scheduling schemes using MARL [9]–[14], the privacy issues are tended to be ignored in these studies. Accessing household information during MARL training may breach user privacy. For example, residents' behaviors can be deduced from the arrival and departure times of household electric vehicles (EVs), and temperature preferences can be inferred from the thermal comfort constraints [15]. Since user data may contain sensitive information, strict data regulations have been established to ensure data governance [16]. Therefore, it is essential to develop a practical MARL framework to address the cooperative load scheduling problem in the cloud-edge environments while complying with data governance.

A. Literature Review

1) Multi-Agent Reinforcement Learning: Several MARL frameworks have been developed to date [17], [18]. A simple framework is to integrate all agents as a single agent with joint state space and action space, where a single-agent RL algorithm is applied [19]. For instance, a *centralized actor-critic* (CAC) framework using prioritized deep deterministic policy gradient (DDPG) is employed to manage all devices of a residential multi-energy system [20]. Although this fully centralized framework theoretically allows cooperative behaviors across individual agents, it fails on simple cooperative MARL problems due to *lazy* agents in practice [21]. Furthermore, the joint action space expands exponentially as the number of agents increases, leading to poor scalability [22]. Additionally, the centralized paradigm requires collecting local observations from all agents during the online execution phase, which imposes high demands on real-time communication. A remedy

This work was supported in part by National Key Research and Development Program of China under Grant No. 2022YFE0140600, and in part by the Shanghai Artificial Intelligence Laboratory.

Z. Qin is with the Automatic Control Laboratory, EPFL, Lausanne 1015, Switzerland. (email: zhaoming.qin@epfl.ch)

N. Dong and Z. Wang are with the Shanghai Artificial Intelligence Laboratory, Shanghai, 200232, China. (emails: {dongnanqing,wangzhefan}@pjlab.org.cn)

D. Liu is with the Department of Automation, Tsinghua University, Beijing, 100084, China. (email: kfliudi@mail.tsinghua.edu.cn)

J. Cao is with the Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China. (email: jcao@tsinghua.edu.cn)

Corresponding authors: N. Dong, D. Liu, and J. Cao.

is *distributed actor-critic* framework [23], in which each agent can access the observations of its neighbors via distributed communications networks.

In this work, we pursue fully decentralized policies depending only on local observations of agents. A straightforward framework to learn decentralized policies is to train the agents independently [17]. For example, an *independent actor-critic* (IAC) framework using the proximal policy optimization (PPO) algorithm is adopted to optimize a multi-household energy management scheme [9], while the independent Q-learning is applied to the demand response programs for different components in residential buildings [24]. From the perspective of a single agent in such a framework, the behaviors and policies of other agents are not observable [18]. Consequently, even if an agent's own policy remains unchanged, the global reward function it receives varies with the policies of other agents. Thus, each agent interacts with a non-stationary environment, making the learning process highly unstable.

To address the non-stationary environment during the learning process of decentralized policies, the framework *decentralized actors with centralized critic* (DACC) is widely adopted by previous MARL approaches [18], [25]–[27]. Its centralized critic has access to all agents' information during learning, ensuring accurate evaluation on the global rewards. While, the decentralized actors, *i.e.*, the decentralized policies, are executed using their corresponding agents' information. Thus, DACC mitigates the challenge of non-stationary environments during learning. Nevertheless, these advantages of DACC come at the expense of agent privacy, since all agents' information must be shared with the centralized critic during learning.

Although few MARL algorithms take privacy issues into account [28], some have the potential to preserve agents' private information during the training phase. For instance, in the multi-actor-attention-critic (MAAC) algorithm [26], the original observations and actions are first encoded into embeddings by local functions. This approach ensures that the central attention accesses only the encoded data. Ye *et al.* [14] leveraged this characteristic to protect consumer privacy in local electricity markets. However, MAAC does not perfectly guarantee privacy, as the high-dimensional embeddings of agents remain vulnerable to privacy attacks. These embeddings, despite being encoded representations, could be exploited by adversaries via advanced machine learning techniques, such as deep learning-based inversion attacks or statistical inference methods. Making matters even worse, in the case that attackers gain access to the detailed parameters or structures of the local embedding functions, the original observations and actions could be (partially) reconstructed, posing a serious threat to agents' sensitive information including position trajectories and behavioral preferences. Finally, it is impractical to deploy MAAC on a large scale in communication-restricted cloud-edge environments. The transmission of high-dimensional information between agents and the cloud significantly increases the communication burden in large-scale systems. Furthermore, the inherent self-attention mechanism incurs noticeable computational costs as the number of agents increases.

2) Data Governance: Data governance is a data management concept focused on ensuring the availability, usability, integrity, and security of the data [29]. In the era of information technology, data privacy has emerged as a prominent topic of ethical and legal discussion [30]. This work specifically addresses data privacy issues within the scope of data governance. Due to the risk of information leakage, data regulations [16] prohibit the data holder from transferring user data out of local devices in any form [31]. Note that this constraint differs from the privacy-persevering techniques widely adopted in the literature, such as differential privacy [32]. While integrating privacy-persevering techniques with MARL can safeguard user privacy in data transmission, it does not necessarily fulfill the requirements of data governance.

3) Edge artificial intelligence: Recent research has introduced various technologies to enhance the efficiency and security of MARL in edge computing environment. Chen et al. explored edge multi-task transfer learning, providing valuable insights into data-driven task allocation methods crucial for optimizing multi-agent environments [33]. Xiong et al. proposed a RL-based framework that intelligently allocates resources across edge devices, with the goal of improving key performance indicators such as latency and energy efficiency [34]. These studies underscore the importance of combining advanced RL techniques with robust data governance frameworks to ensure privacy and efficiency of residential microgrids.

B. Contributions

In this work, we intend to minimize the total operation costs of a residential microgrid within a communication-restricted cloud-edge environment, while effectively preserving local household information. We formulate this cooperative load scheduling problem as a finite-horizon decentralized partial observable Markov decision process (Dec-POMDP). To facilitate collaborative control of distributed demand-side resources and ensure user privacy, we introduce *decentralized actors with distributed critics* (DADC), a novel MARL framework designed with data governance in mind. In the proposed framework, each household operates an individual actor and critic within the edge layer, relying solely on its local information. The local critic networks compute scalar value functions for each household, which are then transmitted to the cloud layer. By restricting communication to scalar values rather than exchanging raw data or model parameters, the framework ensures robust data governance. At the cloud layer, a global value function is estimated using a feed-forward network, which takes as input the concatenation of all individual value functions. This hierarchical learning structure preserves privacy while enabling global optimization. The learning process for both the distributed critics and the cloud-level network is achieved by backpropagating the gradients derived from global temporal-difference (TD) updates, which are computed in the cloud layer based on the global reward signal. This method effectively balances privacy, computational efficiency, and system performance, making it well-suited for decentralized optimization and deployment in resource-constrained edge

environments. The contributions of this paper are summarized as follows.

1. We propose DADC, a novel MARL framework to address the cooperative load scheduling task of a residential microgrid while minimizing user privacy leakage. Unlike existing MARL frameworks used in most load scheduling schemes [9]–[14], [19], [20], DADC ensures that each household only shares an encoded scalar value with the cloud layer during each time step of training phase, efficiently preserving private data.
2. In DADC, the global value function is computed using only the scalar individual values, and the cloud-level network enables linear computational complexity with respect to the number of households. These features facilitate the scalable deployment of DADC in the cloud-edge environments.
3. We empirically evaluate DADC using real-world load data, providing practical insights into the problem formulated in this work. Our results demonstrate that DADC significantly outperform IAC, a seminal baseline under data governance. Furthermore, DADC can achieve comparable performance to DACC, a general MARL framework without privacy-preserving mechanisms, highlighting its superiority in maintaining privacy while achieving efficient load scheduling.

II. Problem Formulation

A. Cloud-Edge Environment

Consider a cloud-edge environment for residential load scheduling. As illustrated in Fig. 1, we assume an isolated microgrid at the edge layer, consisting of a set $\mathcal{D} = \{1, \dots, n\}$ of n households and distributed generators (DGs). The DGs and households communicate bidirectionally with cloud layer to maintain power balance and coordinate the operation of all flexible loads. Without loss of generality, each household is assumed to be equipped with base loads, one EV and one air conditioner (AC). Moreover, each household has one home energy management system (HEMS) that schedules controllable appliances including AC and EV. To ensure data governance, each HEMS can only access public information from the DGs along with its own local information. We consider this load scheduling problem over a horizon T time steps, with each step of duration Δt, *i.e.*, $t \in \mathcal{T} = \{1, \dots, T\}$.

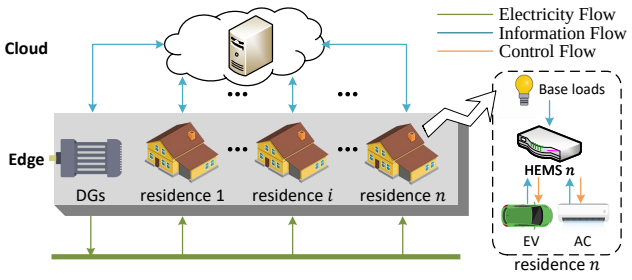

Fig. 1. The cloud-edge environment for residential load scheduling. The DGs supply electricity to households. The HEMSs take as input public information from the DGs and private observations from local households, and generate control signals for local flexible appliances.

B. System Model

The ACs can be dynamically adjusted to maintain thermal comfort of the occupants in the corresponding households. The indoor temperature dynamics for household i are described as follows [11],

$$T^{\text{in}}_{i,t+1} = \mathbf{F}^{\text{AC}}_i \left(T^{\text{in}}_{i,t}, T^{\text{out}}_{i,t}, P^{\text{AC}}_{i,t}, \varrho_{i,t}\right), \tag{1}$$

where $\mathbf{F}^{\text{AC}}_i(\cdot)$ denotes the transition function of indoor temperature with respect to four variables, *i.e.*, current indoor temperature $T^{\text{in}}_{i,t}$, outdoor temperature $T^{\text{out}}_{i,t}$, AC power $P^{\text{AC}}_{i,t}$, and disturbance $\varrho_{i,t}$. Accurately modeling thermal dynamics is typically intractable. Therefore, we assume that the explicit form of the function $\mathbf{F}^{\text{AC}}_i(\cdot)$ is unknown. The working power of ACs can be continuously adjusted within a range,

$$0 \le P^{\text{AC}}_{i,t} \le \overline{P}^{\text{AC}}_i, \tag{2}$$

where $\overline{P}^{\text{AC}}_i$ denotes the maximum working power of the AC in household i. To ensure the thermal comfort of occupants, the following indoor temperature constraint should be satisfied,

$$\underline{T}^{\text{in}}_i \le T^{\text{in}}_{i,t} \le \overline{T}^{\text{in}}_i, \tag{3}$$

where $\underline{T}^{\text{in}}_i$, $\overline{T}^{\text{in}}_i$ denote the lower and upper limits of comfortable temperature in household i, respectively.

The dynamics of EV battery energy are as follows,

$$E^{\text{EV}}_{i,t+1} = \begin{cases} E^{\text{init}}_i, & \text{if } t{+}1{=}t^{\text{a}}_i, \\ E^{\text{EV}}_{i,t} + \eta^{\text{c}}_i P^{\text{EV}}_{i,t} \Delta t, & \text{if } t^{\text{a}}_i{\le}t{<}t^{\text{d}}_i \text{ and } P^{\text{EV}}_{i,t}{\ge}0, \\ E^{\text{EV}}_{i,t} + P^{\text{EV}}_{i,t} \Delta t / \eta^{\text{d}}_i, & \text{if } t^{\text{a}}_i{\le}t{<}t^{\text{d}}_i \text{ and } P^{\text{EV}}_{i,t}{<}0, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

In (4), the variables $E^{\text{EV}}_{i,t}$ and $P^{\text{EV}}_{i,t}$ stand for the battery energy and the charging/discharging power of the EV in household i at time step t, respectively. The parameters E^{init}_i, η^{c}_i, η^{d}_i, t^{a}_i and t^{d}_i are the initial battery energy, the charging and discharging efficiency coefficients, the arrival time and departure time of the EV in household i, respectively. The target EV battery energy should be satisfied at the departure time of the EV,

$$E^{\text{EV}}_{i,t^i_d} \ge E^{\text{targ}}_i, \tag{5}$$

where E^{targ}_i denotes the target battery energy of the EV in household i. Moreover, the charging/discharging power and the battery energy of the EV must be maintained within a range,

$$-\overline{P}^{\text{EV}}_i \le P^{\text{EV}}_{i,t} \le \overline{P}^{\text{EV}}_i, \ \underline{E}^{\text{EV}}_i \le E^{\text{EV}}_{i,t} \le \overline{E}^{\text{EV}}_i, \tag{6}$$

where $\overline{P}^{\text{EV}}_i$, $\underline{E}^{\text{EV}}_i$ and $\overline{E}^{\text{EV}}_i$ represent the maximum charging/discharging power, the minimum and maximum battery energy of the EV in household i.

We assume that DGs have sufficient generation capacity to maintain the power balance of the whole microgrid. Moreover, at each time step t, DGs are automatically adjusted to meet residential electricity needs,

$$P^{\text{DG}}_t = \sum_{i \in \mathcal{D}} \left(P^{\text{BL}}_{i,t} + P^{\text{AC}}_{i,t} + P^{\text{EV}}_{i,t}\right), \tag{7}$$

where $P^{\text{BL}}_{i,t}$ denotes the power of base loads in household i at time step t.

C. Objective Function

The total operation cost of the microgrid can be divided into two parts, *i.e.*, the generation cost of DGs and the adjustment cost of DGs. The former is determined by the output power of DGs. The latter depends on the fluctuation of the output power of DGs because the frequent power adjustment would degrade the service life of DGs. Thus, the total cost at time step t can be presented as follows.

$$C_t = \mathbf{G}_1\left(P_t^{\mathrm{DG}}\right) + \mathbf{G}_2\left(P_t^{\mathrm{DG}} - P_{t-1}^{\mathrm{DG}}\right), \tag{8}$$

where $\mathbf{G}_1(\cdot)$ is the generation cost function of DGs with respect to current output power of DGs [35], [36], and $\mathbf{G}_2(\cdot)$ is the adjustment cost function of DGs with respect to the difference between the output power of DGs at current and last time step. At each time step i, the DGs report the incurred cost to the cloud. It is notable that the functions $\mathbf{G}_1(\cdot)$ and $\mathbf{G}_2(\cdot)$ can be non-linear, thus the individual cost functions specific to households are not available in general.

Based on the above-mentioned models and objective function, a stochastic optimization problem minimizing the long-term microgrid operation cost can be formulated as follows,

$$\begin{aligned} \min_{P_{i,t}^{\mathrm{AC}}, P_{i,t}^{\mathrm{EV}}, i\in\mathcal{D}, t\in\mathcal{T}} \quad & \mathbb{E}\left[\sum_{t\in\mathcal{T}} C_t\right] \\ \text{s.t.} \quad & (1)-(8) \end{aligned} \tag{9}$$

D. Dec-POMDP Formulation

In this subsection, we formulate the cooperative load scheduling problem following Dec-POMDP. The agents in Dec-POMDP are specified as the HEMSs in the microgrid. Since model-free MARL does not rely on the prior knowledge of state transition probability distribution, we focus on three components, the global state and local observations, the actions, and the global reward function.

1) Dec-POMDP: Let $\mathcal{P}(\Omega)$ denote the set of all probability distribution over the space Ω. A finite-horizon Dec-POMDP [22] can be mathematically described by a tuple $\langle \mathcal{D}, \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathbf{P}_s, \mathbf{P}_o, \mathbf{R}, T\rangle$, where

- $\mathcal{D}$ denotes the set of HEMSs.
- $\mathcal{S}$ denotes the space of global states.
- $\mathcal{A} \equiv \times_{i\in\mathcal{D}}\mathcal{A}_i$ denotes the set of joint actions.
- $\mathcal{O} \equiv \times_{i\in\mathcal{D}}\mathcal{O}_i$ denotes the set of joint observations.
- $\mathbf{P}_s : \mathcal{S}\times\mathcal{A}\mapsto\mathcal{P}(\mathcal{S})$ denotes the state transition function.
- $\mathbf{P}_o : \mathcal{S}\times\mathcal{A}\mapsto\mathcal{P}(\mathcal{O})$ denotes the joint observation function.
- $\mathbf{R} : \mathcal{S}\times\mathcal{A}\times\mathcal{S}\mapsto\mathbb{R}$ denotes the global reward function.
- $T \in \mathbb{N}^+$ denotes the horizon.

At every time step t, each HEMS i takes an action $a_{i,t}$ from its individual action space $\mathcal{A}_i$, forming a joint action $\boldsymbol{a}_t$ which leads to a transition to a new state $\boldsymbol{s}_{t+1}\sim\mathbf{P}_s(\boldsymbol{s}_t, \boldsymbol{a}_t)$ and a global reward $r_t = \mathbf{R}(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_{t+1})$. Moreover, the environment emits a joint observation $\mathbf{o}_{t+1} \sim \mathbf{P}_o(\boldsymbol{s}_{t+1}, \boldsymbol{a}_t)$ where each HEMS i only draws its own observation $o_{i,t+1}$.

The goal of the finite-horizon Dec-POMDP is to learn a joint policy $\boldsymbol{\pi} : \mathcal{O}\times\mathcal{A}\mapsto[0, +\infty)$ mapping a joint observation to a probability distribution over actions in *continuous* joint action space, which maximizes the expectation of accumulated global reward $\sum_{t=1}^{T} r_t$. To evaluate the performance of the joint policy $\boldsymbol{\pi}$, we define the joint state value function as

$$V^{\boldsymbol{\pi}}(\boldsymbol{o}_t) := \mathbb{E}_{\boldsymbol{s}_{t+1:T}, \mathbf{o}_{t+1:T}, \boldsymbol{a}_{t:T}\sim\boldsymbol{\pi}}\Big[\sum_{t'=t}^{T} r_{t'}|\boldsymbol{o}_t\Big]. \tag{10}$$

Here, $\mathbb{E}$ denotes the expectation. The subscript of $\mathbb{E}$ enumerates the variables being integrated over, where the global states, joint observations and actions are sampled sequentially from the dynamics model $\mathbf{P}_s$, $\mathbf{P}_o$ and policy $\boldsymbol{\pi}$, respectively. Similarly, the joint state-action value function is defined as

$$Q^{\boldsymbol{\pi}}(\boldsymbol{o}_t, \boldsymbol{a}_t) := \mathbb{E}_{\boldsymbol{s}_{t+1:T}, \mathbf{o}_{t+1:T}, \boldsymbol{a}_{t+1:T}\sim\boldsymbol{\pi}}\Big[\sum_{t'=t}^{T} r_{t'}|\boldsymbol{o}_t, \boldsymbol{a}_t\Big]. \tag{11}$$

The advantage function $A^{\boldsymbol{\pi}}(\mathbf{o}_t, \boldsymbol{a}_t) := Q^{\boldsymbol{\pi}}(\boldsymbol{o}_t, \boldsymbol{a}_t) - V^{\boldsymbol{\pi}}(\boldsymbol{o}_t)$ measures whether the action $\boldsymbol{a}_t$ is better than the default behavior of policy $\boldsymbol{\pi}$.

2) Global state and local observations: In the considered residential load scheduling scenario, the global state $\boldsymbol{s}_t$ incorporates the information owned by all HEMSs and the information from DGs. Since MARL algorithms do not operates over the global state, we omit the mathematical expression of $\boldsymbol{s}_t$. To enable the cooperative scheduling of all households, the power of DGs is viewed as common information and provided to each HEMS. The observation of HEMS $i \in \mathcal{D}$ is

$$o_{i,t} = \left[t, P_t^{\mathrm{DG}}, P_{i,t}^{\mathrm{BL}}, P_{i,t}^{\mathrm{PV}}, T_{i,t}^{\mathrm{out}}, T_{i,t}^{\mathrm{in}}, E_{i,t}^{\mathrm{EV}}, E_i^{\mathrm{targ}}, t_i^{\mathrm{d}}\right], \tag{12}$$

Here, the observation $o_{i,t}$ includes current time step t, which enables the policies to adapt to time-dependent behaviors, such as outdoor temperature and EV arrival/departure time. The last seven components in (12) are local information of household i which should be preserved.

3) Actions: The problem (9) involves two categories of decision variables, namely the working power of ACs and the charging/discharging power of EVs. To facilitate the training of MARL, we unify the continuous action spaces of all decision variables to $[-1, 1]$ by introducing control signals for ACs and EVs. Moreover, by designing the control signals, we rule out the possibility of generating policies that cause the loss of occupant comfort.

The mapping between the working power and control signal of AC of household i is designed as follows,

$$P_{i,t}^{\mathrm{AC}} = \begin{cases} \overline{P}_i^{\mathrm{AC}}, & \text{if } T_{i,t}^{\mathrm{in}} \geq \overline{T}_i^{\mathrm{in}}, \\ 0, & \text{if } T_{i,t}^{\mathrm{in}} \leq \underline{T}_i^{\mathrm{in}}, \\ 0.5\overline{P}_i^{\mathrm{AC}}(u_{i,t}^{\mathrm{AC}} + 1), & \text{otherwise.} \end{cases} \tag{13}$$

Provided that $u_{i,t}^{\mathrm{AC}} \in [-1, 1]$, the working power $P_{i,t}^{\mathrm{AC}}$ is forced to range between $[0, \overline{P}_i^{\mathrm{AC}}]$. Furthermore, the scheme (13) ensures the priority of occupant thermal comfort: AC is forced to run at the maximum power $\overline{P}_i^{\mathrm{AC}}$ when the room temperature exceeds the upper limit of comfortable temperature $\overline{T}_i^{\mathrm{in}}$, and turn off when the room temperature is below the lower limit of comfortable temperature $\underline{T}_i^{\mathrm{in}}$. The power of AC can be adjusted only when the temperature constraint (3) is satisfied.

Considering that the EV charging task (5) should be completed before the departure time, the following inequality must be checked for each time step $t_i^{\mathrm{a}} \leq t < t_i^{\mathrm{d}}$,

$$E_{i,t}^{\mathrm{EV}} + \eta_i^c \overline{P}_i^{\mathrm{EV}} \left(t_i^{\mathrm{d}} - t\right) \Delta t \geq E_i^{\mathrm{targ}}, \tag{14}$$

where the left part indicates the EV battery energy at departure time if the EV is charged at maximum charging power during remaining charging time. Once inequality (14) is not satisfied, the corresponding EV is forced to be charged at maximum power. Therefore, the following EV charging scheme during the charging time is formulated,

$$P_{i,t}^{\mathrm{EV}} = \begin{cases} \overline{P}_i^{\mathrm{EV}}, & \text{if (14) not satisfied,} \\ \max\{0, \overline{P}_i^{\mathrm{EV}} u_{i,t}^{\mathrm{EV}}\}, & \text{else if } E_{i,t}^{\mathrm{EV}} \leq \underline{E}_i^{\mathrm{EV}}, \\ \min\{0, \overline{P}_i^{\mathrm{EV}} u_{i,t}^{\mathrm{EV}}\}, & \text{else if } E_{i,t}^{\mathrm{EV}} \geq \overline{E}_i^{\mathrm{EV}}, \\ \overline{P}_i^{\mathrm{EV}} u_{i,t}^{EV}, & \text{otherwise.} \end{cases} \tag{15}$$

The 2nd and 3rd conditions in (15) guarantees that the battery energy of EV in household i satisfies the constraint (6).

Finally, the individual action of HEMS i at time step t is presented as $a_{i,t} = \left[u_{i,t}^{\mathrm{AC}}, u_{i,t}^{\mathrm{EV}}\right] \in [-1,1]^2, i \in \mathcal{D}$. The joint action formed by individual actions of all HEMSs at time step t is $\boldsymbol{a}_t = [a_{1,t}, \ldots, a_{n,t}] \in [-1,1]^{2n}$.

4) Reward: The load scheduling problem intends to minimize the total operation cost, while the goal of Dec-POMDP is to maximize the accumulated reward. Therefore, we define the immediate global reward taking joint action $\boldsymbol{a}_t$ in state $\boldsymbol{s}_t$ as the negative cost of DGs, i.e., $r_t = -C_t$.

III. MARL Framework in Cloud-Edge Environment under Data Governance

In this section, we introduce DADC, a novel actor-critic framework designed to enable HEMSs to strictly preserve local observations while facilitating efficient collective training. We then elaborate the distributed training process for DADC, utilizing the PPO algorithm.

A. Architecture

DADC employs a structure consisting of decentralized actors and distributed critics. The critics are capable of estimating both the state value function and the action-state value function. For demonstration purposes, Fig. 2 illustrates the structure of DADC, with critics approximating the state value function. The details of this structure are elaborated below.

1) Decentralized Actors: Each HEMS learns a stochastic policy $\pi_i : \mathcal{O}_i \times \mathcal{A}_i \mapsto [0, +\infty)$, parameterized by θ_i, which maps its local observation to a probability distribution over its continuous action space. Note that the policy π_i is conditioned only on the local observation o_i. The joint policy $\boldsymbol{\pi}$ is then constructed from the decentralized policies $\{\pi_i\}_{i=1}^n$:

$$\boldsymbol{\pi}\left(\boldsymbol{a}_t | \mathbf{o}_t\right) := \prod_{i=1}^{n} \pi_i(a_{i,t} | o_{i,t}; \theta_i), \tag{16}$$

where $\boldsymbol{a}_t = (a_{i,t}, ..., a_{i,t})$ and $\boldsymbol{o}_t = (o_{i,t}, ..., o_{i,t})$.

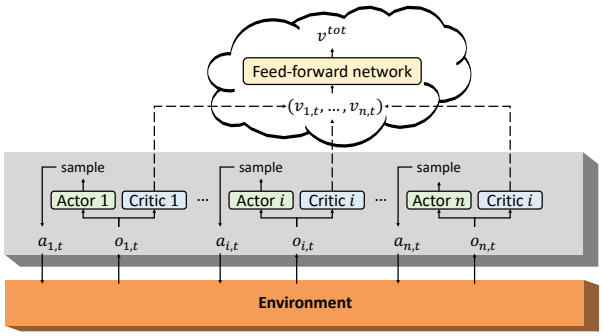

Fig. 2. The framework of DADC in the cloud-edge environment. At the edge layer, the actor network and critic network of each HEMS yield individual policy and a scalar value $v_{i,t}$ only using its local observation $o_{i,t}$, respectively. The individual action $a_{i,t}$ is then sampled according to the generated policy, and the scalar value $v_{i,t}$ is communicated to cloud. At the cloud layer, a learnable feed-forward network maps the concatenation of n scalar values to the global value estimation v^{tot}.

2) Distributed Critics: DACC adopted by existing cooperative multi-agent actor-critic algorithms has a centralized critic to approximate the global value function of the joint policy, which requires the global state including the local observations of all agents, although the decentralized execution is allowed after training. To ensure data governance, the proposed DADC decomposes the approximation of the global value function into two steps,

$$v_{i,t} = V_i(o_{i,t}; \phi_i), i = 1, \ldots, n, \tag{17a}$$

$$V^{\boldsymbol{\pi}}(\mathbf{o}_t) \approx V^{\mathrm{tot}}\left(v_{1,t}, ..., v_{n,t}; \varphi\right). \tag{17b}$$

In (17a), individual critic $V_i(\cdot; \phi_i) : \mathcal{O}_i \to \mathbb{R}$, parameterized by ϕ_i, maps local observation $o_{i,t}$ to a scalar value $v_{i,t}$ at the edge layer. Subsequently, each HEMS transmits $v_{i,t}$ to the cloud. In (17b), a feed-forward network $V^{\mathrm{tot}}(\cdot; \varphi) : \mathbb{R}^n \to \mathbb{R}$, parameterized by φ, maps the concatenation of received n scalar values to the global value estimation at the cloud layer. Put differently, the approximation of the global value function at the cloud layer only requires the collection of the individual value functions, which are scalar values encoded by individual critics of agents at the edge layer. This design brings three key advantages.

- The local observations of agents are preserved strictly since it is intractable to analyze or deduce the original information through a scalar.
- The communication burden between the cloud layer and the edge layer is significantly reduced. For instance, when comparing DADC to MAAC, The local embedding functions in MAAC send transmit vectors of dimension d to the central attention mechanism. Thus, the total communication complexity in a cloud-edge environment using MAAC is $O(nd)$ whereas with DADC it is reduced to is $O(n)$.
- The computational burden at the cloud layer is also reduced by the use of a feed-forward network. The computational complexity is $O(n)$ if the hidden layers of the feed-forward network have fixed number of units, whereas MAAC's complexity is $O(n^2 d)$ due to the self-attention network.

Therefore, the proposed DADC framework facilitates large-scale deployment in the cloud-edge environment.

3) Inner Structure: The individual actors and critics and the feed-forward network are illuminated in Fig. 3. Each agent's policy is represented by a combination of a gate recurrent unit (GRU) and multi-layer perceptrons (MLPs). The GRU module, a gating mechanism in recurrent neural networks, uses the hidden state $h^{\pi}_{i,t-1}$ to retain information from previous time steps, thereby enabling the agent to mitigate the challenges posed by partial observability. Therefore, the integration of a GRU module and MLPs endows the individual actor network with the potential to generate effective policies based on limit observations. Similar to the structure of the individual actor network, the individual critic network also incorporates a GRU module and two MLPs. The feed-forward network at the cloud layer is entirely composed of MLPs.

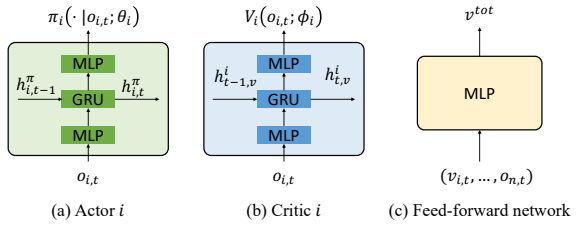

Fig. 3. (a) Individual actor network. This network takes as input the local observation $o_{i,t}$ and the hidden state $h^{\pi}_{i,t-1}$, and generates the probability distribution over the individual action space. (b) Individual critic network. This network takes as input the local observation $o_{i,t}$ and the hidden state $h^{v}_{i,t-1}$ as input, and yields the individual value estimation. (c) Feed-forward network. This network take as input the concatenation of n scalars, and outputs the global value estimation.

B. Distributed Training

DADC can be trained using various RL algorithms in a distributed manner. For demonstration purposes, we employ the PPO algorithm [37] as an example and adapt the single-agent PPO to the multi-agent settings within a cloud-edge environment.

We first recall the single-agent PPO with an actor π and a critic V. Given a policy π parameterized by $\tilde{\theta}$, a batch of samples can be obtained, and the estimate of the advantage function $\hat{A}_t$ can be computed by the general advantage estimation (GAE) method [38]

$$\hat{A}_t := \sum_{t'=t}^{T} \lambda^{t'-t}(-V(\mathbf{o}_{t'};\tilde{\theta}_v) + r_{t'} + V(\mathbf{o}_{t'+1};\tilde{\theta}_v)),$$

where parameter λ is used to control the trade-off between variance and bias of the estimate, and $\tilde{\theta}_v$ is the parameter of the critic V. Then, the actor network is updated by minimizing the loss

$$\mathcal{L}^a(\theta) := \hat{\mathbb{E}}_t[\min(w_t(\theta)\hat{A}_t, \text{clip}(w_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)].$$

Here, the expectation $\hat{\mathbb{E}}_t[\cdot]$ denotes the empirical estimation over a finite batch of samples. The probability ratio $w_t(\theta)$ is defined as $\frac{\pi(\boldsymbol{a}_t|\mathbf{o}_t;\theta)}{\pi(\boldsymbol{a}_t|\mathbf{o}_t;\tilde{\theta})}$. The hyperparameter ϵ limits the change in the probability ratio. The critic network V is updated by minimizing $\mathcal{L}^c(\theta_v) := \hat{\mathbb{E}}_t[(V(\mathbf{o}_t;\theta_v) - V(\mathbf{o}_t;\tilde{\theta}_v) - \hat{A}_t)^2]$.

In the DADC framework, the global value function is approximated by (17). Consequently, the global critic loss in the multi-agent settings is computed as

$$\mathcal{L}^c := \hat{\mathbb{E}}_t\left[(V^{\text{tot}}(v_{1,t},...,v_{n,t};\varphi) - \tilde{v}^{\text{tot}} - \hat{A}_t)^2\right]. \quad (18)$$

where $\tilde{v}^{\text{tot}} := V^{\text{tot}}(v_{1,t},...,v_{n,t};\tilde{\varphi})$. By applying the the chain rule, the gradient of the feed-forward network at the cloud layer is given by $\triangle\varphi = \frac{\partial\mathcal{L}^c}{\partial\varphi}$, while the gradient of the individual critic i at the edge layer is $\triangle\phi_i = \frac{\partial\mathcal{L}^c}{\partial V_i}\frac{\partial V_i}{\partial\phi_i}$. Note that the gradient term $\frac{\partial\mathcal{L}^c}{\partial V_i}$ must be communicated from the cloud to HEMS i. In this manner, each individual critic $V_i(\cdot;\phi_i)$ is trained by backpropagating gradients from the global TD updates, which depends on the joint global reward. In other words, $V_i(\cdot;\phi_i)$ is learned implicitly rather than from any reward specific to HEMS i.

The individual actor loss function is defined as

$$\mathcal{L}^a_i(\theta_i) = \hat{\mathbb{E}}_t\left[\min(w_{i,t}(\theta_i)\hat{A}_t, \text{clip}\left(w_{i,t}(\theta_i), 1-\epsilon, 1+\epsilon\right)\hat{A}_t)\right], \quad (19)$$

where $w_{i,t}(\theta_i) := \frac{\pi_i(a_{i,t}|o_{i,t};\theta_i)}{\pi_i(a_{i,t}|o_{i,t};\tilde{\theta}_i)}$. Note that the individual loss $\mathcal{L}^a_i(\theta_i)$ is calculated locally by HEMS i once the global value function is received. Then, the gradient of individual actor i can be computed as $\triangle\theta_i = \frac{\partial\mathcal{L}^a_i}{\partial\theta_i}$.

The distributed training process in the cloud-edge environment is detailed in Algo. 1. Each training iteration is divided into three primary stages, 1) interaction with the environment, 2) estimation of the global advantage function, and 3) parameter updates. In the algorithm, operations executed at the edge layer are shaded gray, while those at the cloud layer are shaded yellow for clarity.

In the first stage, shown in lines 4-10, agents operates in a fully decentralized manner. At each time step, each HEMS interacts with the environment by independently selecting and executing its own action. The only information uploaded to the cloud layer is the individual value estimation calculated by each HEMS.

The cloud layer exclusively performs the second stage. As shown in line 14, the global value function is estimated using only the scalar value estimates received from HEMSs, without requiring access to their local observations or actions. The estimation of the global advantage function at time step t employs a backward-view TD method, utilizing the advantage function at time step $t+1$.

The third state involves a collaborative update process between the edge and cloud layer. HEMSs first calculate their individual value functions using the updated parameters and transmit these scalar values to the cloud layer. The cloud computes the gradients of the global critic loss with respect to both the feed-forward network parameters and individual value functions. These gradients are used to update the feed-forward network and are distributed back to the respective HEMSs. Finally, each HEMS update its local actor and critic networks using the received gradients and global advantage functions.

Algorithm 1 Distributed Training for DADC with PPO

Initialize θ_i and ϕ_i for each HEMS; initialize φ for feed-forward network.
for episode $= 1$ to episode$_{\max}$ **do**
  % Interact with the environment
  for $t = 1$ to T **do**
    for all HEMSs i **do**
      $\tilde{\theta}_i \leftarrow \theta_i$, $\tilde{\phi}_i \leftarrow \phi_i$
      Sample action $a_{i,t} \sim \pi_i(\cdot|o_{i,t}; \tilde{\theta}_i)$.
      Execute action $a_{i,t}$ and observe o^i_{t+1}.
      $\tilde{p}_{i,t} \leftarrow \pi_i(a_{i,t}|o_{i,t}; \theta_i)$, $\tilde{v}_{i,t} \leftarrow V_i(o_{i,t}; \phi_i)$.
      Upload $\tilde{v}_{i,t}$ to cloud. ▷ Comm.
  % Estimate global advantage function
  $\hat{A}_T \leftarrow 0, \tilde{v}^{\text{tot}}_{T+1} \leftarrow 0, \tilde{\varphi} \leftarrow \varphi$
  for $t = T$ to 1 **do**
    $\tilde{v}^{\text{tot}}_t \leftarrow V^{\text{tot}}(\tilde{v}_{1,t}, ..., \tilde{v}_{n,t}; \tilde{\varphi})$
    $\hat{A}_t \leftarrow \lambda \hat{A}_{t+1} + r_t + \gamma \tilde{v}^{\text{tot}}_{t+1} - \tilde{v}^{\text{tot}}_t$
  Send $\{\hat{A}_t\}_{t=1}^T$ to each HEMS. ▷ Comm.
  % Update parameter
  % Edge layer
  for all HEMSs i **do**
    $\{v_{i,t}\}_{t=1}^T \leftarrow \{V_i(o_{i,t}; \phi_i)\}_{t=1}^T$
    Upload $\{v_{i,t}\}_{t=1}^T$. ▷ Comm.
  $\mathcal{L}^c \leftarrow \sum_{t=1}^T (V^{\text{tot}}(v_{1,t}, ..., v_{n,t}; \varphi) - \tilde{v}^{\text{tot}}_t - \hat{A}_t)^2$
  Update φ with gradient $\partial \mathcal{L}^c / \partial \varphi$.
  Send $\{\partial \mathcal{L}^c / \partial v_{i,t}\}_{t=1}^T$ to HEMS i. ▷ Comm.
  % Edge layer
  for all HEMSs i **do**
    $\triangle \phi_i \leftarrow \sum_{t=1}^T \partial \mathcal{L}^c / \partial v_{i,t} \cdot \partial v_{i,t} / \partial \phi_i$
    Update ϕ_i with gradient $\triangle \phi_i$.
    for $t = 1$ to T **do**
      $w_{i,t} \leftarrow \pi_i(a_{i,t}|o_{i,t}; \theta_i) / \tilde{p}_{i,t}$
    $\mathcal{L}^a_i \leftarrow \sum \min(w_{i,t}\hat{A}_t, \text{clip}(w_{i,t}, 1-\epsilon, 1+\epsilon)\hat{A}_t)$
    $\triangle \theta_i \leftarrow \sum_{t=1}^T \partial \mathcal{L}^a_i / \partial w_{i,t} \cdot \partial w_{i,t} / \partial \theta_i$
    Update θ_i with gradient $\triangle \theta_i$.

IV. Experiments

In this section, we present the simulation experiments and report the empirical results. We begin by detailing the experimental setup of simulations. Next, we compare the proposed DADC with existing actor-critic frameworks using PPO, and analyze the effectiveness of the learned policies in the context of cooperative load scheduling. Finally, we access the scalability of DADC by evaluating its performance across varying numbers of households.

A. Experiment Setup and Implementation

1) Environment: The simulation environment is built upon OpenAI Gym [39]. The dynamics functions, cost functions and important household parameters are provided in the Appendix. We consider the energy management problem over a one-day period, using a time step of 15 minutes, resulting in a time horizon of $T = 96$. Real-world power assumption data and temperature data are employed to model the power of basic loads and outdoor temperature, sourced from Pecan Street Database [40] and NOAA [41], respectively. We first consider a standard scenario consisting of 10 heterogeneous households in the following three subsections. In Sec. IV-E, we investigate the performance of the proposed DADC in large-scale scenarios.

2) Network Architecture: The individual critic networks share the same structure, consisting of three components, as shown in Fig. 2, a fully-connected MLP with two layers of 64 units followed by $tanh$ nonlinearity, a GRU layer with 64 units, and a fully-connected MLP with one hidden layer of 128 units and one ouput layer of 1 units.

To represent the stochastic policy, we use a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ in this work. Therefore, the individual actor networks have one $tanh$ output for the mean μ and another $sigmoid$ output for the variance σ^2. The non-output layers of individual actor networks share the same structure with the individual critic networks.

The MLP of the feed-forward network in the cloud layer is consisted of one hidden layer of 64 units followed by $tanh$ nonlinearity and one output layer of 1 units.

3) Baseline Frameworks: Two baseline frameworks are considered: IAC and DACC. Notably, one of primary goals of this study is to ensure data governance. In contrast to the proposed DADC and IAC, the DACC framework raises non-trivial concerns regarding both privacy and communication costs. Thus, we report the performance of DACC only in Sec. IV-B and Sec. IV-C for quantitative comparison.

Under the **IAC** framework, each HEMS comprises an independent actor and critic, following the same architecture as the individual actor and critic in DADC. The actor and critic of each HEMS are trained using the single-agent PPO algorithm, thereby eliminating the need for communication among agents.

The **DACC** framework maintains decentralized actors for agents and a centralized critic. The decentralized actors share the same network as the individual actors in DADC, while the centralized critic adopts the network shown in Fig. 3 (b), taking the joint observation of all HEMSs as input rather than the local observation of a single HEMS.

4) Shared Hyperparameters: We optimize the actor and critic networks using Adam with the learning rate of 1×10^{-4} and 3×10^{-4}, respectively. The network parameters are updated every 120 environment steps with the batch size of 120. We run 10 parallel environments to improve the training efficiency. The case studies are conducted on a server with an 8-core AMD Ryzen 7 3700X processor and one single GeForce RTX 2080 GPU.

B. Algorithm performance

First, we compare the proposed DADC framework with other actor-critic frameworks on the cooperative load scheduling problem. For a fair comparison, each frameworks is trained with PPO for six times with different random seeds. In PPO, the GAE parameter is set to be 0.95, and the network parameters are updated 3 times per sample [37].

We apply the following evaluation procedure during training: for each trial, training is paused every 1000 episodes, and

10 independent episodes are run with each agent performing decentralized action selection. The cumulative reward for each episode is termed the *episode reward*.

The training curves are shown in Fig. 4. We observe that IAC fails to learn stable policies, resulting in poor performance, arguably due to the non-stationary environments encountered by its independent agents. In contrast, DACC leverage a global critic to facilitate more stable learning of coordinated behaviors across agents. DADC, on the other hand, achieves slightly better performance than DACC. The policies of DADC escape the local minimum of DACC at the price of a sharp performance decline at about 1×10^5 episodes. This implies that DADC has a better exploration capability. Note that DADC preserves local information, unlike DACC. Thus, Fig. 4 demonstrates the superior performance of DADC over other actor-critic frameworks in this cooperative load scheduling task.

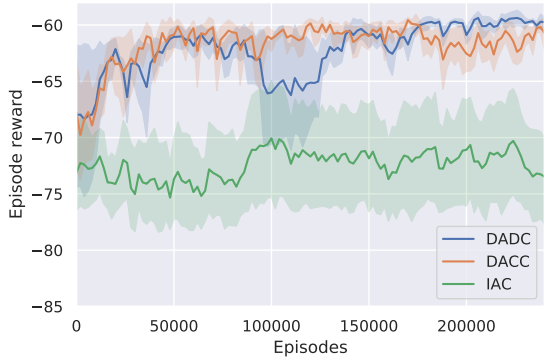

Fig. 4. Training curves of DADC and other frameworks. The solid curves corresponds to the mean and the shaded region to the minimum and maximum episode rewards over the all trials.

C. Effect of Implicit Credit Assignment

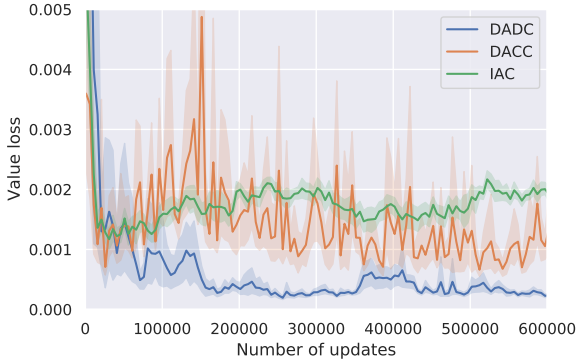

Fig. 5. The value loss for critic networks. DADC achieve the lowest estimation bias for global value function.

As discussed in Section I-A, DACC can implicitly learn credit assignment across agents. To demonstrate this, we plot the value loss for critic networks in Fig. 5. We observe that the independent critic networks in IAC exhibit the highest estimation bias, as the fully decentralized HEMSs in IAC cannot account for the dynamic behaviors of other HEMSs during training.

DACC, with its centralized critic, shows comparatively smaller value loss than IAC. However, the estimate of the global value function remains highly unstable during training. This instability arises because DACC's centralized critic processes observations from all HEMSs, making it slow to adapt to changes in the global reward when any single HEMS adjusts its policy.

In contrast, DADC enables all HEMSs to cooperatively estimate the global value function through distributed critic networks, allowing each individual value function to be learned via end-to-end training. In Fig. 5, DADC achieves a much lower value loss than IAC and DACC, highlighting the effectiveness of implicit credit assignment in DADC. This finding partially explains why DADC performs on par with DACC, despite the cloud receiving considerably less information from each household.

D. Effect of Load Scheduling

We next examine the control effects of DADC on the cooperative load scheduling task. After training, we test the policies that achieved the best evaluation performance during the training phase. The test results, shown in Table I, indicate that DADC reduces the average cost by 11% compared to IAC. Notably, the adjustment cost is reduced by more than 50%.

TABLE I
TEST PERFORMANCE FOR DIFFERENT ACTOR-CRITIC FRAMEWORKS

Metrics	DADC	DACC	IAC
Average Total Cost	58.2 ± 0.9	65.4 ± 1.2	59.5 ± 1.0
Average Generation Cost	55.8 ± 1.0	60.6 ± 1.5	56.7 ± 1.1
Average Adjustment Cost	2.4 ± 0.3	4.9 ± 0.8	2.8 ± 0.4

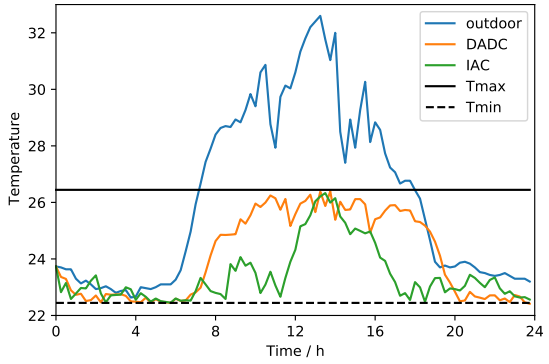

Fig. 6. Indoor temperature. The black solid and dashed lines denote the desirable maximum and minimum indoor temperature, respectively. The orange and green lines denote the indoor temperature curves during one day controlled by decentralized policies with DADC and IAC, respectively.

To present the control effects for ACs, we plot the indoor temperature curves for a single AC over one day in Fig. 6. Both frameworks control the indoor temperature within the specified constraints. However, with DADC, the indoor temperature remains closer to the upper temperature constraint when outdoor temperatures are high, resulting in energy savings and cost reduction compared to IAC. Additionally, the indoor

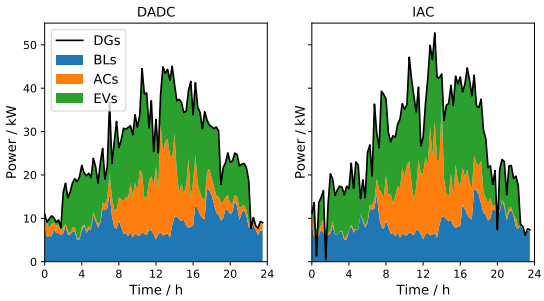

Fig. 7. Load scheduling. The blue, orange and green area denote the power of base load, the total power of ACs and the total charging power, respectively.

temperature curve with DADC is relatively smooth, indicating fewer adjustments to the AC than with IAC.

To demonstrate the overall load scheduling, Fig. 7 displays the base load power, DGs power, total charging/discharging power of EVs, and total working power of ACs. Compared to IAC, the cooperative load scheduling achieved by DADC reveals two salient characteristics. First, the output power adjustments for DGs are relatively stable, resulting to lower adjustment costs for DGs. Second, the peak power of DGs is lower than that with IAC. This indicates that DADC enables HEMSs to learn decentralized policies that allow households to cooperatively schedule load and reduce the global cost. In contrast, IAC fails to learn such cooperative policies due to its fully independent actor-critic structure.

E. Scalability Evaluation

The above subsections presented and discussed simulation results for a scenario with 10 households. In this subsection, we empirically evaluate the effectiveness of the proposed DADC framework as the number of households increases. Table II and Table III report two key metrics for DADC and DACC in scenarios with 10, 100 and 1000 households. The first metric, communication traffic between HEMSs and the cloud, grows linearly with respective to the number of households. The second metric, total processing time at the cloud layer, reflects the precessing burden on the cloud.

Table II and Table III show that DADC requires less than one-fifth of the communication overhead needed by DACC. Moreover, as the number of households increases, the cloud computational efficiency advantage of DADC over DACC becomes more pronounced. Thus, DADC shows significant scalability advantage over DACC in terms of both communication and cloud computational burdens.

TABLE II
SCALABILITY EVALUATION FOR DACC

Number of households	10	100	1000
Communication traffic (MB)	16.9	169	1690
Computation burden (hours)	0.8	3.7	33

V. CONCLUSION

This paper proposes a novel multi-agent actor-critic framework, DADC, to address the cooperative load scheduling problem in a communication-restricted cloud-edge environment. A salient feature of DADC is its two-step approximation of the global value function. First, each HEMS's individual critic network maps its local information into a scalar value, which is subsequently uploaded to the cloud. Second, the cloud estimate the global value function with a feed-forward network that takes these scalar values as inputs.

TABLE III
SCALABILITY EVALUATION FOR DADC

Number of households	10	100	1000
Communication traffic (MB)	3.1	31	310
Computation burden (hours)	0.5	0.82	4.1

This framework brings three significant benefits. First, it enhance user privacy protection. Second, it significantly reduces communication traffic and computational burden on the cloud, thereby improving the training efficiency and scalability. Third, despite the cloud's access to limited information, the decentralized policies learned by HEMSs achieve performance comparable to that of DACC, arguably due to improved implicit credit assignment.

The current DADC framework assumes fixed load types and a fixed number of households during training, which restricts its applicability. Future work will focus on adapting DADC to general scenarios with varying load types and dynamic household participation, enabling broader broader adoption of cooperative residential load scheduling.

APPENDIX

The transition functions and cost functions used for simulation are specified as follows.

$$\mathbf{F}_i^{AC}(T, T^{\text{out}}, P, \varrho) = T + \alpha_i(T^{\text{out}} - T) - \beta_i P + \varrho, \quad (20)$$

where α_i and β_i are the the coefficients associated with the thermal characteristics of corresponding room and AC, and ϱ follows the uniform distribution $\mathcal{U}[-0.1, 0.1]$. The arrival time t_i^{a} is a random variable with probability distribution $\mathcal{U}[\psi_i, \psi_i + \delta_1]$, where ψ_i indicates the parking habit of the occupant in household i, and δ_1 is a shared parameter representing the variance of arrival time. Given the arrival time t_i^{a}, we assume the departure time $t_i^{\text{d}} \sim \mathcal{U}[t_i^{\text{a}} + \delta_2, t_i^{\text{a}} + \delta_3]$, implying that the dwell time of EV i ranges between $[\delta_2, \delta_3]$. The parameters δ_1, δ_2 and δ_3 are set to be $3, 9$ and 12, respectively. Other parameters are randomly sampled according to the range in Table IV. The cost functions of DGs are specified as

TABLE IV
PARAMETER RANGES OF HOUSEHOLDS

Parameters	$\underline{T}_i^{\text{in}}$	$\overline{T}_i^{\text{in}}$	$\overline{P}_i^{\text{AC}}$
Range	$[22, 24]$	$[26, 28]$	$[3, 4]$
Parameters	α_i	β_i	$\overline{P}_i^{\text{EV}}$
Range	$[0.19, 0.21]$	$[0.5, 0.7]$	$[6, 10]$
Parameters	$\overline{E}_i^{\text{EV}}$	η_i^{c}	η_i^{d}
Range	$[40, 60]$	$[0.90, 0.95]$	$[0.90, 0.95]$

$$\begin{aligned}\mathbf{G}_1(P) &= \lambda_1^{\mathrm{DG}} P + \lambda_2^{\mathrm{DG}} P^2, \\ \mathbf{G}_2\left(P_t, P_{t-1}\right) &= \lambda_3^{\mathrm{DG}} |P_t - P_{t-1}|.\end{aligned} \tag{21}$$

where $\lambda_1^{\mathrm{DG}}, \lambda_2^{\mathrm{DG}}, \lambda_3^{\mathrm{DG}}$ denote the cost coefficients of DGs, and are selected as 0.5, 0.0125 and 0.1, respectively.

REFERENCES

[1] "Electric power annual." [Online]. Available: https://www.eia.gov/electricity/annual/

[2] L. Yu, W. Xie, D. Xie, Y. Zou, D. Zhang, Z. Sun, L. Zhang, Y. Zhang, and T. Jiang, "Deep reinforcement learning for smart home energy management," *IEEE IoT-J*, vol. 7, no. 4, pp. 2751–2762, 2020.

[3] Q. Wei, D. Liu, and G. Shi, "A novel dual iterative q-learning method for optimal battery management in smart residential environments," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2509–2518, 2015.

[4] Q. Wei, G. Shi, R. Song, and Y. Liu, "Adaptive dynamic programming-based optimal control scheme for energy storage systems with solar renewable energy," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 7, pp. 5468–5478, 2017.

[5] H. Shuai and H. He, "Online scheduling of a residential microgrid via monte-carlo tree search and a learned model," *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1073–1087, 2021.

[6] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[7] L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, and X. Guan, "A review of deep reinforcement learning for smart building energy management," *IEEE IoT-J*, pp. 1–1, 2021.

[8] C. Xu, S. Liu, C. Zhang, Y. Huang, Z. Lu, and L. Yang, "Multi-agent reinforcement learning based distributed transmission in collaborative cloud-edge systems," *IEEE TVT*, vol. 70, no. 2, pp. 1658–1672, 2021.

[9] C. Zhang, S. R. Kuppannagari, C. Xiong, R. Kannan, and V. K. Prasanna, "A cooperative multi-agent deep reinforcement learning framework for real-time residential load scheduling," in *International Conference on Internet of Things Design and Implementation*, 2019, pp. 59–69.

[10] J. Lee, W. Wang, and D. Niyato, "Demand-side scheduling based on multi-agent deep actor-critic learning for smart grids," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*, 2020, pp. 1–6.

[11] L. Yu, Y. Sun, Z. Xu, C. Shen, D. Yue, T. Jiang, and X. Guan, "Multi-agent deep reinforcement learning for hvac control in commercial buildings," *IEEE Transactions on Smart Grid*, vol. 12, no. 1, pp. 407–419, 2021.

[12] X. Xu, Y. Jia, Y. Xu, Z. Xu, S. Chai, and C. S. Lai, "A multi-agent reinforcement learning-based data-driven method for home energy management," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3201–3211, 2020.

[13] H.-M. Chung, S. Maharjan, Y. Zhang, and F. Eliassen, "Distributed deep reinforcement learning for intelligent load scheduling in residential smart grids," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2752–2763, 2021.

[14] Y. Ye, D. Papadaskalopoulos, Q. Yuan, Y. Tang, and G. Strbac, "Multi-agent deep reinforcement learning for coordinated energy trading and flexibility services provision in local electricity markets," *IEEE Transactions on Smart Grid*, pp. 1–1, 2022.

[15] Z. Qin, D. Liu, H. Hua, and J. Cao, "Privacy preserving load control of residential microgrid via deep reinforcement learning," *IEEE Transactions on Smart Grid*, pp. 1–1, 2021.

[16] European Commission, "General data protection regulation," 2016.

[17] A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," *PloS one*, vol. 12, no. 4, p. e0172395, 2017.

[18] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *AAAI*, vol. 32, no. 1, 2018.

[19] Y. Du, H. Zandi, O. Kotevska, K. Kurte, J. Munk, K. Amasyali, E. Mckee, and F. Li, "Intelligent multi-zone residential hvac control strategy based on deep reinforcement learning," *Applied Energy*, vol. 281, p. 116117, 2021.

[20] Y. Ye, D. Qiu, X. Wu, G. Strbac, and J. Ward, "Model-free real-time autonomous control for a residential multi-energy system using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3068–3082, 2020.

[21] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls *et al.*, "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *AAMAS*, 2018, pp. 2085–2087.

[22] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *ICML*. PMLR, 2018, pp. 4295–4304.

[23] P. Dai, W. Yu, H. Wang, and S. Baldi, "Distributed actor-critic algorithms for multiagent reinforcement learning over directed graphs," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 7210–7221, 2023.

[24] M. Ahrarinouri, M. Rastegar, and A. R. Seifi, "Multiagent reinforcement learning for energy management in residential buildings," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 659–666, 2021.

[25] R. Lowe, Y. WU, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *NIPS*, vol. 30, pp. 6379–6390, 2017.

[26] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *ICML*. PMLR, 2019, pp. 2961–2970.

[27] Y. Wang, B. Han, T. Wang, H. Dong, and C. Zhang, "Dop: Off-policy multi-agent decomposed policy gradients," in *ICLR*, 2020.

[28] Z.-W. Liu, G. Wei, M. Chi, X. Ye, and Y. Li, "Privacy-preserving load scheduling in residential microgrids using multiagent reinforcement learning," *IEEE Journal of Emerging and Selected Topics in Industrial Electronics*, vol. 5, no. 2, pp. 662–669, 2024.

[29] V. Khatri and C. V. Brown, "Designing data governance," *Communications of the ACM*, vol. 53, no. 1, pp. 148–152, 2010.

[30] A. Mehmood, I. Natgunanathan, Y. Xiang, G. Hua, and S. Guo, "Protection of big data privacy," *IEEE Access*, vol. 4, pp. 1821–1834, 2016.

[31] N. Dong, M. Kampffmeyer, I. Voiculescu, and E. Xing, "Federated partially supervised learning with limited decentralized medical images," *IEEE Transactions on Medical Imaging*, 2022.

[32] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 308–318.

[33] Q. Chen, Z. Zheng, C. Hu, D. Wang, and F. Liu, "On-edge multi-task transfer learning: Model and practice with data-driven task allocation," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 6, pp. 1357–1371, 2019.

[34] X. Xiong, K. Zheng, L. Lei, and L. Hou, "Resource allocation based on deep reinforcement learning in iot edge computing," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, pp. 1133–1146, 2020.

[35] Z. Wang, W. Wu, and B. Zhang, "A fully distributed power dispatch method for fast frequency recovery and minimal generation cost in autonomous microgrids," *IEEE Transactions on Smart Grid*, vol. 7, no. 1, pp. 19–31, 2016.

[36] G. Chen, F. L. Lewis, E. N. Feng, and Y. Song, "Distributed optimal active power control of multiple generation systems," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 11, pp. 7079–7090, 2015.

[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[38] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[39] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[40] Pecan Street Database. [Online]. Available: http://www.pecanstreet.org/.

[41] NOAA Data. [Online]. Available: https://www.ncdc.noaa.gov/.

Zhaoming Qin (Graduate Student Member, IEEE) received the B.Sc. degree in Automation from Beihang University, Beijing, China, in 2019, and the M.Sc. degree in control theories and engineering from Tsinghua University, Beijing, China, in 2022, respectively. He is currently pursuing the Ph.D. degree at the Automatic Control Laboratory, EPFL, Lausanne, Switzerland. His research focuses on the intersection of data-driven control, learning and optimization.

Di Liu was born in Henan, China, in 1990. He received the B.S degree in electrical engineering and management in 2013, the M.S degree in electronic and communication engineering in 2015, and the Ph.D. degree in electrical engineering in 2020, all from North China Electric Power University, Beijing, China. From 2020 to 2024, he worked as a postdoctoral researcher at Tsinghua University. He is currently an assistant researcher at the Department of Electrical Engineering, Tsinghua University. His research interests include demand side management, power system stability control, and energy internet.

Zhefan Wang received the Bachelor's degree in computer science and technology from Liaoning University, Liaoning, China, in 2024. She is pursuing a Ph.D. degree in electronic information at Fudan University, Shanghai, China. Her research interests include AI for science and large language models.

Nanqing Dong received the master's degree from the Department of Statistical Science, Cornell University, Ithaca, NY, USA, in 2017, and the Ph.D. degree from the Department of Computer Science, University of Oxford, Oxford, UK, in 2023. He is currently an Associate Professor at the Shanghai Artificial Intelligence Laboratory and a Doctoral Supervisor at the Shanghai Innovation Institute. He visited the Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, USA, from 2017 to 2019. He is a member of IEEE and CAASS, and a special committee member of CCF and CAAI. His research interests include machine learning, computer vision, optimization, and AI for science.

Junwei Cao (Senior Member, IEEE) received the bachelor's and master's degrees in control theories and engineering from Tsinghua University, Beijing, China, in 1998 and 1996, respectively, and the Ph.D. degree in computer science from the University of Warwick, Coventry, U.K., in 2001.

He is currently a Professor of Beijing National Research Center for Information Science and Technology, Tsinghua University. Prior to joining Tsinghua University in 2006, he was a Research Scientist with MIT LIGO Laboratory and NEC Laboratories Europe for about five years. He has published over 400 papers and cited by international scholars for over 120 000 times. He has authored or edited ten books. His research is focused on distributed computing technologies and energy/power applications. He is a Senior Member of the IEEE Computer Society and a member of the ACM and CCF.