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

Zhaoming Qin, 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. In a simplified scenario, centralized training, the most common paradigm for MARL, precludes large-scale deployment of MARL in communication-constrained cloud-edge environments. As a remedy, distributed training shows unparalleled advantages in real-life application scenarios compared with its centralized counterpart. However, distributed training still faces the challenge of system scalability, *e.g.*, the high communication burden 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. The proposed solution 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. In this way, the private data of households is preserved completely and the communication cost is reduced significantly. The simulation experiments demonstrate that the proposed framework can achieve 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 37.8% in the U.S. in 2019 [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) 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. During the training of MARL, the access to the information of households may breach the user privacy. For example, the behaviors of the residents can be captured from the arrival and departure times of the household electric vehicles (EVs). Another case is that the temperature preference of the occupants can be inferred from the thermal comfort constraints [15]. As user's data may contain sensitive information, to ensure data governance, strict data regulations have been established [16]. Therefore, it is essential to develop a practical MARL framework to address the cooperative load scheduling problem in the cloud-edge environment under data governance.

A. Literature Review

1) Multi-Agent Reinforcement Learning: To this end, several MARL frameworks have been developed [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 agents, it fails on simple cooperative MARL problems due to *lazy* agents in practice [21]. Furthermore, the joint action space increases exponentially with the number of agents, resulting in poor scalability [22]. Additionally, the centralized framework needs to collect original observations from agents and distribute actions to them. This relies on high-quality real-time communication and poses a threat to the privacy of the agents.

The above-mentioned constraints necessitate decentralized policies that depend only on the local observations of agents.

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)

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 [23]. From the perspective of a single agent in such a framework, the behaviors and policies of other agents cannot be captured [18]. In this sense, the global reward function for a specific agent is varying with respect to policies of other agents, even though its own policy is unchanged. Thus, each agent interacts with a non-stationary environment where the learning process becomes 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], [24]–[26]. Its centralized critic is available to the information of all agents during learning. While, the decentralized actors, *i.e.*, the decentralized policies, are executed using the information only from their corresponding agents. Thus, DACC mitigates the challenge of non-stationary environments during learning. For example, Chung *et al.* [13] used it to learn the cooperative load scheduling of multiple households. However, the advantages of DACC are at the expense of the privacy of agents, since the information of all agents must be shared with the centralized critic during learning.

Despite few MARL algorithms involved in privacy issues [27], some of them have the potential to preserve the private information of agents at the training phase. For example, when calculating the centralized critic in multi-actor-attention-critic (MAAC) algorithm [25], the original observations and actions are first encoded into embeddings by local embedding functions, such that the central attention network can only access the encoded information. Ye *et al.* [14] took advantage of this characteristic to preserve the privacy of consumers in local electricity markets. Nevertheless, MAAC is intractable to guarantee privacy effectively. The high-dimensional embeddings of agents uploaded to cloud still contain sufficient information which could be analyzed by some privacy attack methods. A potential attack scenario involves adversaries collecting and analyzing a substantial number of local agents' embedding vectors. By leveraging machine learning or statistical methods, they can uncover hidden patterns and correlations within these embeddings, which might enable attackers to deduce partially the original observations or actions, thereby compromising the agents' privacy. Furthermore, if attackers obtain detailed parameters or structures of the local embedding functions utilized in the MAAC algorithm, they could attempt to invert the process and reconstruct the original observations and actions from the encoded embedding vectors. Although this reconstruction process may not be entirely precise, attackers could still extract sensitive information, such as agents' locations, behavioral preferences, and other private data. Moreover, it is impractical to bring MAAC to large-scale deployment in communication-restricted cloud-edge environments. The high-dimensional information transmitted between agents and cloud exacerbate the communication burden in large-scale systems, and the self-attention mechanism of MAAC causes noticeable computational costs with a growing body of agents.

2) Data Governance: Data governance is a data management concept concerning managing the availability, usability, integrity, and security of the data [28]. In the era of information technology, data privacy has become an emerging topic of active ethical and legal discussion [29]. This work specifically focuses on data privacy issues in data governance. Due to the risk of information leakage, data regulations [16] prevent the data holder from transferring the users' data out of local devices in any format [30]. Note, this constraint is different from the privacy-persevering technique widely adopted in the literature, *e.g.* differential privacy [31]. While integrating privacy-persevering techniques with MARL can protect the user privacy in data transmission, it does not necessarily meet the requirements of data governance.

3) Edge artificial intelligence: Recent research has introduced various technologies to improve the efficiency and security of MARL in edge computing environment. Chen et al. explored edge multi task transfer learning and gained a deeper understanding of data-driven task allocation methods that are crucial for optimizing multi-agent environments [32]. These works emphasize the importance of combining advanced RL techniques with robust data governance frameworks to ensure privacy and efficiency of residential microgrids. Xiong et al. applied deep RL to optimizing resource allocation in edge-computing environment [33]. This research proposes a RL-based framework that intelligently allocates resources across various edge devices, aiming to enhance key performance indicators such as latency reduction and energy efficiency. By utilizing the adaptive learning capabilities of RL, this approach effectively addresses the complexity and dynamic nature of multi-agent environments, resulting in improved resource management and system scalability.

B. Contributions

In this work, we intend to minimize the total operation costs of a residential microgrid in a communication-restricted cloud-edge environment while preserving the local information of households effectively. This cooperative load scheduling problem is formulated as a finite-horizon decentralized partial observable Markov decision process (Dec-POMDP). To achieve the collaborative control of distributed resources on the demand side while considering privacy persevering, we propose *decentralized actors with distributed critics* (DADC), a novel MARL framework under data governance. In this framework, each household maintains an individual actor and critic in the edge layer, both of which act only on the local observation of the household. The local critic networks generate scalar individual value functions which are communicated to the cloud layer. In this sense, data governance is ensured as there is no exchange of raw data or model weights between the cloud and edge layers. The global value function is approximated by a feed-forward network taking the concatenation of the individual value functions as input. The distributed critics and the feed-forward network are learnable by backpropagating the

gradients from global temporal-difference (TD) updates, which are dependent on the global reward and computed in the cloud layer. The contributions of this paper can be summarized as follows.

- We propose DADC, a novel MARL framework to address the cooperative load scheduling problem of a residential microgrid while minimizing the leakage of user privacy data. In contrast to the existing MARL frameworks adopted by most load scheduling schemes [9]–[14], [19], [20], each household in DADC only shares an encoded scalar value with the cloud layer at each time step of training phase, preserving its own private data efficiently.
- The calculation of the global value function in DADC only requires the scalar individual value functions from households, and the feed-forward network in DADC leads to a linear computational complexity with respect to the number of households, both of which facilitates the scalable deployment of DADC in the cloud-edge environment.
- We empirically evaluate the performance of DADC with real-world load data and provide an empirical understanding of the problem of interest formulated in this work. Specifically, we show that DADC can outperform IAC, a seminal baseline under data governance, by a large margin. Compared with DACC, a general MARL framework that has no privacy-preserving design to ensure data governance, DADC can achieve comparable performance.

II. Problem Formulation

A. Cloud-Edge Environment

Considering a cloud-edge environment for residential load scheduling. As shown in Fig. 1, we assume that there is an isolated microgrid in the edge layer, consisting of a set $\mathcal{D} = \{1, \ldots, n\}$ of n households and distributed generators (DGs). The DGs and each household communicate with cloud layer bidirectionally, to achieve power balance and coordinate the behavior 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, there exists one home energy management system (HEMS) for each household to schedule the controllable appliances including AC and EV. To ensure data governance, each HEMS can only access the public information from DGs and its own local information. We consider this problem over the horizon T with each time step Δt, *i.e.*, $t \in \mathcal{T} = \{1, \ldots, T\}$.

B. System Model

The ACs can be dynamically adjusted to maintain thermal comfort of the occupants in the corresponding households. The dynamics of indoor temperature in household i can be presented 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 $\varrho_{i,t}$. Normally, it is intractable to obtain accurate thermal dynamics models. Therefore, we assume that the explicit form of function $\mathbf{F}^{\text{AC}}_i(\cdot)$ is unknown. The working power of ACs can be continuously adjusted within a range,

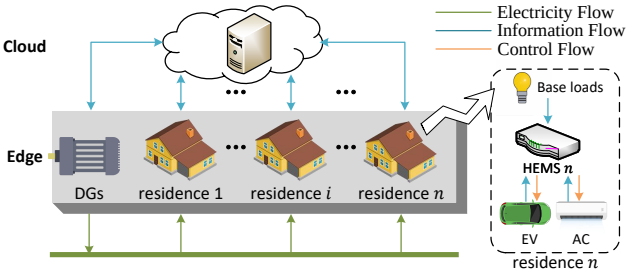

Fig. 1. The cloud-edge environment for residential load scheduling. The DGs supply electricity to households. The HEMSs take the public information from DGs and the private observations from local households as input, and generate control signals for the local load scheduling.

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

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 desirable minimum and maximum indoor temperature in household i, respectively.

It is assumed that EVs can be charged or discharged during parking time. The dynamics of EV battery energy in household i can be represented 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 $P^{\text{EV}}_{i,t}$ are 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 DGs have sufficient generation capacity to maintain the power balance in the whole microgrid. Moreover, at each time step t, DGs are automatically adjusted to meet residential eletricity needs,

$$P_t^{\mathrm{DG}} = \sum_{i \in \mathcal{D}} \left(P_{i,t}^{\mathrm{BL}} + P_{i,t}^{\mathrm{AC}} + P_{i,t}^{\mathrm{EV}} \right), \tag{7}$$

where $P_{i,t}^{\mathrm{BL}}$ 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 [34], [35], 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 generally available.

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

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

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 the model-free MARL does not rely on the state transition probability distribution, we focus on three components, *i.e.* 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 set 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 accumulated global reward $\sum_{t=1}^{T} r_t$. We assume that the joint policy is stochastic. The notation of joint policy would degenerate into $\boldsymbol{\pi} : \mathcal{O} \mapsto \mathcal{A}$ if the policy is deterministic. To evaluate the performance of the joint policy $\boldsymbol{\pi}$, we define the joint state value function as

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

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

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

The advantage function, widely used by policy-based RL, is represented as

$$A^{\boldsymbol{\pi}}(\mathbf{o}_t, \boldsymbol{a}_t) = Q^{\boldsymbol{\pi}}(\boldsymbol{o}_t, \boldsymbol{a}_t) - V^{\boldsymbol{\pi}}(\boldsymbol{o}_t), \tag{12}$$

which measures whether the action $\boldsymbol{a}_t$ is better than the default behaviour 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 defined as

$$o_{i,t} = \left[t, P_t^{\mathrm{DG}}, P_{i,t}^{\mathrm{BL}}, P_{i,t}^{\mathrm{PV}}, T_{i,t}^{\mathrm{out}}, T_{i,t}^{\mathrm{in}}, E_{i,t}^{\mathrm{EV}}, E_i^{\mathrm{targ}}, t_i^{\mathrm{d}} \right], \tag{13}$$

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 (13) are local information of household i which should be preserved.

3) Actions: We unify the continuous action spaces to $[-1, 1]$ by introducing control signals for ACs and EVs to facilitate the training of MARL. To guarantee the thermal comfort of the occupants and satisfy the indoor temperature constraint (3), the following scheme is designed,

$$P_{i,t}^{\mathrm{AC}} = \begin{cases} \overline{P}_i^{\mathrm{AC}}, & \text{if } T_{i,t}^{\mathrm{in}} \geq \overline{T}_i^{\mathrm{in}}, \\ 0, & \text{if } T_{i,t}^{\mathrm{in}} \leq \underline{T}_i^{\mathrm{in}}, \\ 0.5 \overline{P}_i^{\mathrm{AC}} (u_{i,t}^{\mathrm{AC}} + 1), & \text{otherwise.} \end{cases} \tag{14}$$

Under this scheme, ACs would take corresponding mode to ameliorate the thermal condition: run at the maximum power if current indoor temperature is above desirable maximum temperature; turn-off if current indoor temperature is below the desirable minimum temperature.

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

$$E_{i,t}^{\mathrm{EV}} + \eta_i^c \overline{P}_i^{\mathrm{EV}} \left(t_i^{\mathrm{d}} - t\right) \Delta t \geq E_i^{\mathrm{targ}}, \tag{15}$$

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 (15) is not satisfied, the EV must be charged at maximum power. Therefore, the following EV charging scheme during the charging time is formulated,

$$P_{i,t}^{\mathrm{EV}} = \begin{cases} \overline{P}_i^{\mathrm{EV}}, & \text{if (15) not satisfied,} \\ \max\{0, \overline{P}_i^{\mathrm{EV}} u_{i,t}^{\mathrm{EV}}\}, & \text{else if } E_{i,t}^{\mathrm{EV}} \leq \underline{E}_i^{\mathrm{EV}}, \\ \min\{0, \overline{P}_i^{\mathrm{EV}} u_{i,t}^{\mathrm{EV}}\}, & \text{else if } E_{i,t}^{\mathrm{EV}} \geq \overline{E}_i^{\mathrm{EV}}, \\ \overline{P}_i^{\mathrm{EV}} u_{i,t}^{EV}, & \text{otherwise.} \end{cases} \tag{16}$$

The 2nd and 3rd conditions in (16) are designed to ensure that the battery energy of EV in household i satisfies the constraint (6).

In this sense, the individual action of HEMS i at time step t is presented as

$$a_{i,t} = \left[u_{i,t}^{\mathrm{AC}}, u_{i,t}^{\mathrm{EV}}\right] \in [-1, 1]^2, i \in \mathcal{D}. \tag{17}$$

And the joint action formed by individual actions of all HEMSs at time step t is

$$\boldsymbol{a}_t = [a_{1,t}, \ldots, a_{n,t}] \in [-1, 1]^{2n}. \tag{18}$$

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,

$$r_t = -C_t. \tag{19}$$

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

As stated in Sec. I-A, confronted with the formulated large-scale load scheduling problem in the cloud-edge environment, the existing actor-critic algorithms fail to obtain cooperative policies effectively. In this section, we first propose DADC, a novel actor-critic algorithm which allows HEMSs to preserve their local observations strictly and perform collective training efficiently. Then, we elaborate the distributed training for DADC with PPO algorithm and adaptation to increasing agents.

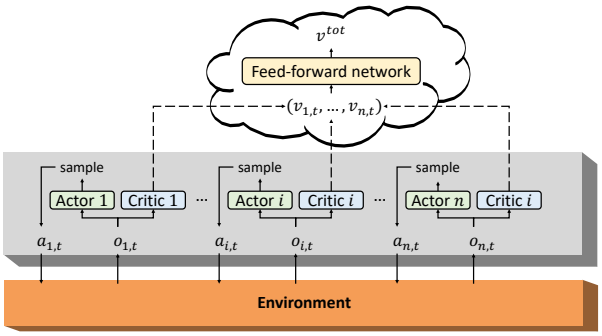

Fig. 2. The framework of DADC in the cloud-edge environment. In the edge layer, the actor network and critic network of each HEMS yield individual policy and a scalar value $v_{i,t}$ only using its local observation $o_{i,t}$, respectively; then individual action $a_{i,t}$ is sampled according to the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

A. Architecture

DADC maintains the structure of decentralized actors and distributed critics. The critics can be used to estimate both the state value function and action-state value function. For demonstration purposes, we intuitively present the structure of DADC with the critics approximating the state value function in Fig. 2. We elaborate this structure as follows.

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

$$\boldsymbol{\pi}\left(\boldsymbol{a}_t | \mathbf{o}_t\right) \triangleq \prod_{i=1}^{n} \pi_i(a_{i,t} | o_{i,t}; \theta_i), \tag{20}$$

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

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,

$$v_{i,t} = V_i(o_{i,t}; \phi_i), i = 1, \ldots, n, \tag{21a}$$

$$V^{\boldsymbol{\pi}}(\mathbf{o}_t) \approx V^{\mathrm{tot}}\left(v_{1,t}, ..., v_{n,t}; \varphi\right). \tag{21b}$$

In (21a), 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 $v_{i,t}$ in the edge layer. Subsequently, each HEMS transmits $v_{i,t}$ to the cloud. In (21b), 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 in the cloud layer.

In this way, the approximation of the global value function in the cloud layer only requires to collect the individual value functions which are scalar values encoded by individual critics of agents in the edge layer. This brings three advantages. First,

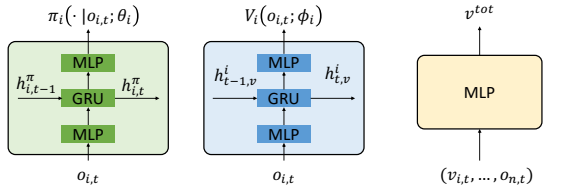

Fig. 3. (a) Individual actor network. This network take the local observation $o_{i,t}$ and the hidden state at last time step $h^{\pi}_{i,t-1}$ as input, and generates the probability distribution over the individual action space. (b) Individual critic network. This network take the local observation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

the local observations of agents are preserved strictly since it is intractable to analyze or deduce the original information through a scalar. Second, the communication burden between cloud layer and edge layer is significantly reduced. We compare DADC with MAAC for example. The local embedding functions of agents in MAAC send vectors with dimension d to the central attention mechanism. The total communication complexity of cloud-edge environment with MAAC is $O(nd)$ while that with DADC is $O(n)$. Third, the computational burden in the cloud layer is also reduced by the feed-forward network. The computational complexity is $O(n)$ if the hidden layers of the feed-forward network have fixed number of units, while the MAAC is $O(n^2d)$ 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. The policy of each agent is represented as the combination of gate recurrent unit (GRU) and multi-layer perceptrons (MLPs). The GRU module, a gating mechanism in recurrent neural networks (RNNs), use the hidden state $h^{\pi}_{i,t-1}$ to memorize the information before the step t, which enables the agent to alleviate the severe problem of partial observability. Therefore, with the integration of a GRU module and MLPs, the individual actor network has the potential to generate a good policy given limit observation. Similar to the inner structure of individual actor network, the individual critic network is also concatenated by a GRU module and two MLPs. The feed-forward network is composed of MLP entirely.

B. Distributed Training

DADC can be trained by many RL algorithms in a distributed manner. For demonstration purposes, we take the PPO algorithm [36] as an example and adapt the single-agent PPO to the multi-agent settings and the 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 [37]

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

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 the loss

$$\mathcal{L}^c(\theta_v) = \hat{\mathbb{E}}_t[(V(\mathbf{o}_t;\theta_v) - V(\mathbf{o}_t;\tilde{\theta}_v) - \hat{A}_t)^2].$$

In the DADC framework, the global value function is approximated by (21). Thus, the global critic loss in multi-agent settings is computed as

$$\mathcal{L}^c = \hat{\mathbb{E}}_t\left[(V^{\text{tot}}(v_{1,t},...,v_{n,t};\varphi) - \tilde{v}^{\text{tot}} - \hat{A}_t)^2\right]. \tag{22}$$

where $\tilde{v}^{\text{tot}} = V^{\text{tot}}(v_{1,t},...,v_{n,t};\tilde{\varphi})$. According the the chain rule, the gradient of the feed-forward network in the cloud layer is computed as $\triangle\varphi = \frac{\partial\mathcal{L}^c}{\partial\varphi}$, while the gradient of individual critic i in 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 $\frac{\partial\mathcal{L}^c}{\partial V_i}$ should be distributed to HEMS i from the cloud. In this way, each individual critic $V_i(\cdot;\phi_i)$ is learned by backpropagating the gradients from global TD updates, which is dependent on the joint global reward. Put differently, $V_i(\cdot;\phi_i)$ is learned implicitly rather than from any reward specific to HEMS i. The individual actor loss function is computed as

$$\mathcal{L}^a_i(\theta_i) = \hat{\mathbb{E}}_t\left[\min(w_{i,t}(\theta_i)\hat{A}_t, \text{clip}\left(w_{i,t}(\theta_i), 1-\epsilon, 1+\epsilon\right)\hat{A}_t)\right], \tag{23}$$

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 is calculated locally by corresponding agent 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. Overall, the training during one iteration can be divided into three parts, namely, interacting with the environment, estimating the global advantage function, and performing parameter updates. The gray region indicates the operation in the edge layer, while the yellow region indicates the operation in the cloud layer.

The first part, shown in lines 4-10 of Algo. 1, is conducted by agents in a fully decentralized manner. At each time step, each HEMS interacts with the environment by selecting and executing its own action independently. The only information uploaded to the cloud layer is the individual value estimation calculated by each HEMS.

The second part is performed by the cloud layer. As shown in line 14, the estimation of global value function only relies on

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 $\{\hat{A}_t\}^T_{t=1}$ to each HEMS. ▷ Comm.
  % Update parameter
  % Edge layer
  for all HEMSs i **do**
    $\{v_{i,t}\}^T_{t=1} \leftarrow \{V_i(o_{i,t}; \phi_i)\}^T_{t=1}$
    Upload $\{v_{i,t}\}^T_{t=1}$. ▷ Comm.
  $\mathcal{L}^c \leftarrow \sum^T_{t=1}(V^{\text{tot}}(v_{1,t}, ..., v_{n,t}; \varphi) - \tilde{v}^{\text{tot}}_t - \hat{A}_t)^2$
  Update φ with gradient $\partial \mathcal{L}^c / \partial \varphi$.
  Send $\{\partial \mathcal{L}^c / \partial v_{i,t}\}^T_{t=1}$ to HEMS i. ▷ Comm.
  % Edge layer
  for all HEMSs i **do**
    $\triangle \phi_i \leftarrow \sum^T_{t=1} \partial \mathcal{L}^c / \partial v_{i,t} \cdot \partial v_{i,t} / \partial \phi_i$
    Update ϕ_i with gradient $\triangle \phi_i$.
    for $t = 1$ to T **do**
      $w_{i,t} \leftarrow \pi_i(a_{i,t}|o_{i,t}; \theta_i) / \tilde{p}_{i,t}$
    $\mathcal{L}^a_i \leftarrow \sum \min(w_{i,t}\hat{A}_t, \text{clip}(w_{i,t}, 1-\epsilon, 1+\epsilon)\hat{A}_t)$
    $\triangle \theta_i \leftarrow \sum^T_{t=1} \partial \mathcal{L}^a_i / \partial w_{i,t} \cdot \partial w_{i,t} / \partial \theta_i$
    Update θ_i with gradient $\triangle \theta_i$.

the scalar values uploaded by HEMSs, rather than the global states including the observations of all HEMSs. The estimation of global advantage function at time step t uses the advantage function at time step $t+1$, which is oriented backward in time.

The third part is executed alternately in edge layer and cloud layer. HEMSs in the edge layer first calculate their own individual value functions given the new parameters and upload them to cloud. the gradients of the global critic loss with respect to the parameters and individual value functions are used to update the feed-forward network and distributed to corresponding HEMSs, respectively. The local actor and critic networks of agents are updated using the received gradients and global advantage functions.

IV. Experiments

In this section, we conduct the simulation experiments and report the empirical results. We first present the simulation experiment setup. Then, we compare the proposed DADC with existing actor-critic frameworks using PPO and analyze the control effects of learned policies in the context of cooperative load scheduling. Finally, we evaluate the scalability of the proposed DADC with varying numbers of households.

A. Experiment Setup and Implementation

1) Environment: The simulation environment is built upon OpenAI Gym [38]. The dynamics functions, cost functions and important parameters of households are provided in Appendix. We consider the energy management problem during one day. We use a time step of 15 minutes, such that the time horizon T is 96. The real-world power data and temperature data are employed to model the power of basic loads and outdoor temperature, which are available from Pecan Street Database [39] and NOAA [40]. We first consider a standard scenario consisting of 10 heterogeneous households in subsequent 3 subsections. In Sec. IV-E, we investigate the performance of the proposed DADC in more 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, including IAC and DACC. It is worth mentioning that one of primary goals of this study is to ensure data governance. In contrast to the proposed DADC and IAC, DACC can cause non-trivial concerns over both privacy and communication cost in practice. Thus, we only report the performance of DACC in Sec. IV-B and Sec. IV-C for quantitative comparison.

Under **IAC**, each HEMS is composed of an independent actor and critic, following the same architecture with the individual actor and critic of DADC. The actor and critic of each HEMS are trained with single-agent PPO algorithm, dispensing with the communication among agents.

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

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 with other actor-critic frameworks on the formulated cooperative load scheduling problem. For a fair comparison, we train all frameworks 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 during one sample [36].

In order to evaluate the performance during training, we adopt the following evaluation procedure: for each trial of a framework, we pause the training every 1000 episodes and run 10 independent episodes with each agent performing decentralized action selection. The sum of reward during an episode is referred to as the *episode reward*.

The training curves are reported in Fig. 4. We can observe that IAC fails to learn stable policies and result in poor performance. This highly unstable training can be attributed to the non-stationary environments observed by the independent agents of IAC. In contrast, owing to the global critic, DACC is able to learn the coordinated behaviours across all agents more stably. Compared to DACC, DADC can learn policies with slightly better performance. 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 the local information of agents while DACC fails. Therefore, Fig. 4 shows the superior performance of DADC over other actor-critic frameworks.

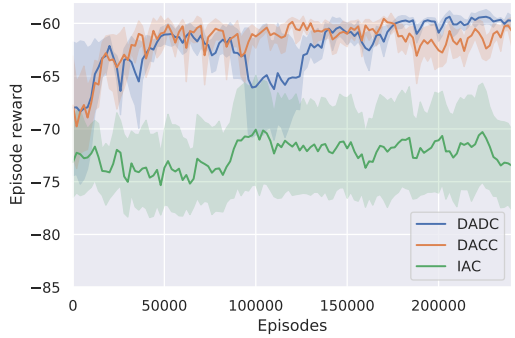

Fig. 4. Training curves of DADC and other frameworks. The solid curves corresponds to the mean and the shaded region to the minimum and maximum episode rewards over the all trials.

C. Effect of Implicit Credit Assignment

As stated in Section I-A, DACC is able to implicitly learn credit assignment across agents. To demonstrate this point, we plot the value loss for critic networks in Fig. 5. We can observe the independent critic networks of IAC have the maximum bias of estimate, because the fully decentralized HEMS in IAC cannot capture the other HEMSs changing their behaviour during training.

Due to the centralized critic, DACC has relatively smaller value loss than IAC. However, the estimate for global value function is highly unstable during training. This can be explained as follows. The centralized critic of DACC takes the observations of all HEMSs as input, thus it cannot rapidly adapt the change of global reward when one specific HEMS changes the behaviour.

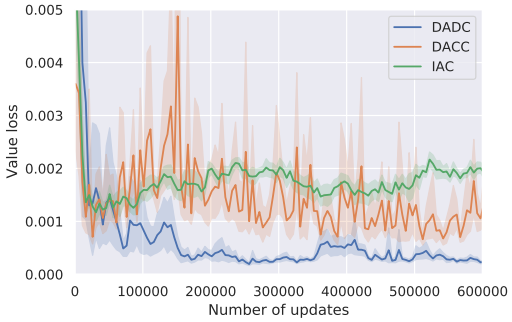

Fig. 5. The value loss for critic networks. DADC achieve the lowest estimation bias for global value function.

In contrast, all HEMSs cooperatively estimate the global value function with distributed critic networks in DADC. The individual value function can be learned with end-to-end training. In Fig. 5, we can observe that DADC has much smaller value loss than IAC and DACC. This demonstrates the effect of implicit credit assignment in DADC and partially explains why DADC shows parallel performance with DACC even though the cloud only receives much less information from households.

D. Effect of Load Scheduling

Next, we specify the control effects of DADC on the cooperative load scheduling problem. After training, we test the policies with best evaluation performance during training. The test results shown in Table I demonstrate that the average cost of DADC is reduced by 11% than that of IAC. Especially, 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

To present the control effects for ACs, we plot the indoor temperature curves associated with one AC during one day in Fig. 6. We can observe that both two framework can control the indoor temperature within the constraint. However, the indoor temperature with DADC is closer to the upper temperature constraint when outdoor temperature is high, which can save energy and reduce cost compared with IAC. Moreover, we can see that the indoor temperature curve with DADC is relatively smooth, indicating fewer adjustments to the AC compared with IAC.

To demonstrate the overall load scheduling, the power of base load, the power of DGs, the total charging/discharging power of EVs, and the total working power of ACs are

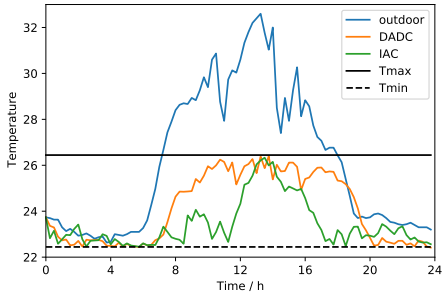

Fig. 6. Indoor temperature. The black solid and dashed lines denote the desirable maximum and minimum indoor temperature, respectively. The orange and green lines denote the indoor temperature curves during one day controlled by decentralized policies with DADC and IAC, respectively.

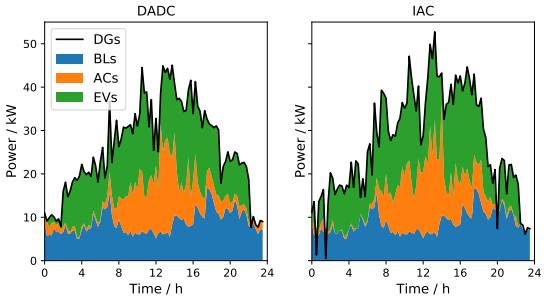

Fig. 7. Load scheduling. The blue, orange and green area denote the power of base load, the total power of ACs and the total charging power, respectively.

shown in Fig. 7. Compared with IAC, the cooperative load scheduling with DADC present two salient points. First, the adjustments for the output power of DGs is relatively flat, resulting to less adjustment costs of DGs. Second, the peak power of DGs is smaller than that with IAC. In this sense, DADC allows HEMSs to learn decentralized policies such that households can cooperatively schedule load and reduce the global cost. While, IAC fails to learn such policies due to the fully independent actor-critic structure.

E. Scalability Evaluation

The above subsections present and discuss simulation results of the scenario consisting 10 households. In this subsection, we empirically the effectiveness of the proposed DADC framework with increasing number of households. In Table II and Table III, we report two key metrics of DADC and DACC in scenarios with 10, 100 and 1000 households. The first metric denotes the communication traffic between HEMSs and the cloud, which grows linearly with respective to the number of households.The second metric is the total processing time in the cloud layer, representing the communication burden of the cloud. It can be concluded from Table II and Table III that the communication overhead required by DADC is less than one-fifth of that required by DACC. Moreover, as the number of households increases, the cloud computational efficiency advantage of DADC over DACC becomes more pronounced. Therefore, DADC shows sigficant scalability advantage over DACC in terms of communication burden and cloud computational burden.

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

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

V. CONCLUSION

In this paper, a novel multi-agent actor-critic framework, DADC, is proposed to address the cooperative load scheduling problem in a communication-restricted cloud-edge environment. A salient feature of DADC is the two-step approximation of global value function. First, the individual critic network of each HEMS maps its own local information into a scalar value, which is subsequently uploaded to the cloud. Second, the cloud estimate global value function with a feed-forward network taking the collection of scalar values as input. This framework brings three significant benefits. Firstly, users' privacy information is better protected. Secondly, the communication traffic and the computation burden in the cloud are reduced significantly, effectively enhancing the training efficiency and scalability. Thirdly, even though the cloud accesses much less information, the decentralized policies of HEMSs exhibit comparable performance with that of DACC, arguably due to better implicit credit assignment.

The DADC framework proposed in this paper necessitates fixed types of loads of each household and a fixed number of households during training, which limits its application. Therefore, in future work, it is interesting to adapt DADC for general scenarios with varying types of flexible loads and varying households, so that more loads and households can participate in cooperative residential load scheduling.

APPENDIX

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

$$\mathbf{F}_i^{AC}(T, T^{\text{out}}, P, \varrho) = T + a_i(T^{\text{out}} - T) - b_i P + \varrho, \quad (24)$$

where a_i and b_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 parameters are randomly sampled according to the range in Table IV.

TABLE IV
PARAMETER RANGES OF HOUSEHOLDS

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

The cost functions of DGs are specified as

$$\begin{aligned} \mathbf{G}_1(P) &= \lambda_1^{\mathrm{DG}} P + \lambda_2^{\mathrm{DG}} P^2, \\ \mathbf{G}_2\left(P_t, P_{t-1}\right) &= \lambda_3^{\mathrm{DG}} |P_t - P_{t-1}|. \end{aligned} \tag{25}$$

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 2019. [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 TSG*, 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 (SmartGridComm)*, 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 TSG*, 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 TSG*, 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 TII*, 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 TSG*, 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 TSG*, 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 TSG*, 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] M. Ahrarinouri, M. Rastegar, and A. R. Seifi, "Multiagent reinforcement learning for energy management in residential buildings," *IEEE TII*, vol. 17, no. 1, pp. 659–666, 2021.

[24] 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.

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

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

[27] 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.

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

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

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

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

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

[37] 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.

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

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

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