Optimal Scheduling in IoT-Driven Smart Isolated Microgrids Based on Deep Reinforcement Learning

Jiaju Qi, Lei Lei, *Senior Member, IEEE*, Kan Zheng, *Senior Member, IEEE*, Simon X. Yang, *Senior Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

***Abstract*—In this article, we investigate the scheduling issue of diesel generators (DGs) in an Internet of Things (IoT)-Driven isolated microgrid (MG) by deep reinforcement learning (DRL). The renewable energy is fully exploited under the uncertainty of renewable generation and load demand. The DRL agent learns an optimal policy from history renewable and load data of previous days, where the policy can generate real-time decisions based on observations of past renewable and load data of previous hours collected by connected sensors. The goal is to reduce operating cost on the premise of ensuring supply–demand balance. In specific, a novel finite-horizon partial observable Markov decision process (POMDP) model is conceived considering the spinning reserve. In order to overcome the challenge of discrete-continuous hybrid action space due to the binary DG switching decision and continuous energy dispatch (ED) decision, a DRL algorithm, namely, the hybrid action finite-horizon RDPG (HAFH-RDPG), is proposed. HAFH-RDPG seamlessly integrates two classical DRL algorithms, i.e., deep Q-network (DQN) and recurrent deterministic policy gradient (RDPG), based on a finite-horizon dynamic programming (DP) framework. Extensive experiments are performed with real-world data in an IoT-driven MG to evaluate the capability of the proposed algorithm in handling the uncertainty due to interhour and interday power fluctuation and to compare its performance with those of the benchmark algorithms.**

***Index Terms*—Deep reinforcement learning (DRL), energy management, microgrid (MG).**

NOMENCLATURE

System Model

δ_t	Surplus of generating capacity.
η_{ch},η_{dis}	Charging/discharging efficiencies of the battery.
DG_d	dth DG.
a_d, b_d, c_d	Quadratic/monomial/constant coefficient of DG power generation cost curve.
$B_t^{\text{DG}_d}$	ON/OFF status of DG_d.
C_R, C_S, C_{SR}	Fixed running/start-up/spinning reserve cost coefficient.
$c_t^{\text{DG}_d}$	DG power generation cost.
c_t^{OP}	The operating cost
$c_t^{\text{S\_DG}_d}$, $c_t^{\text{R\_DG}_d}$, $c_t^{\text{SR\_DG}_d}$	Start-up/running/spinning reserve cost.
D,d	Number/index of DGs.
E_t	SoC of the battery.
$E_{\min}$, $E_{\max}$	Minimum/maximum energy level of the battery.
$P^E_{\text{ch\_lim}}$, $P^E_{\text{dis\_lim}}$	Charging/discharging power limits of the battery.
$P^E_{\max}$	Maximum charging or discharging power of the battery.
$P^{\text{DG}}_{\min}$, $P^{\text{DG}}_{\max}$	Minimum/maximum power limits.
$P_t^{\text{DG}_d\_S}$	Set-point for the DG output power.
$P_t^{\text{DG}_d}$	Output power of DG_d.
P_t^{EL}	Equivalent load.
P_t^{ERR}	Unbalanced power after charging or discharging battery.
P_t^E	Charging or discharging power of the battery
P_t^L	Load consumption power.
P_t^{PV}	Output power of PV.
P_t^{UB}	Unbalanced power after DG output power correction.
T,t	Number/index of time steps.
u_t	Charging or discharging status of the battery.
$U_t^{\text{DG}_d}$	Binary switching decision of DG_d.

POMDP Model

α,β	Learning rate for updating θ or ω.
γ	Discount factor.
$\hat{\mathcal{A}}^{\text{SW}}$	Simplified switching action space.
$\hat{k}^*$, $a^{\text{ED}*}_{\hat{k}^*}$	Optimal switching and ED action derived by the proposed algorithm.
$\mathcal{A}$	Action space.
$\mathcal{A}^{\text{SW}}$, $\mathcal{A}^{\text{ED}}$	Switching and ED action space.
$\mathcal{A}^{\text{ED}}_k$	ED action space corresponding to the switching action k.
$\mathcal{H}$	History space.

Manuscript received 13 December 2022; revised 22 March 2023; accepted 11 April 2023. Date of publication 24 April 2023; date of current version 7 September 2023. This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada under Discovery Grant and in part by the University of Guelph through CARE-AI Seed Fund. *(Corresponding author: Lei Lei.)*

Jiaju Qi, Lei Lei, and Simon X. Yang are with the School of Engineering, University of Guelph, Guelph, ON N1G 2W1, Canada (e-mail: jiaju@uoguelph.ca; leil@uoguelph.ca; syang@uoguelph.ca).

Kan Zheng is with the College of Electrical Engineering and Computer Sciences, Ningbo University, Ningbo 315211, China (e-mail: zhengkan@nbu.edu.cn).

Xuemin Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: sshen@uwaterloo.ca).

Digital Object Identifier 10.1109/JIOT.2023.3267625

μ_k,λ_k	Actor and critic networks for switching action k.
π^*	Optimal policy.
τ	Length of history time window .
θ, ω	Weights of μ_k and λ_k.
a_k^{ED}	Action in $\mathcal{A}_k^{\mathrm{ED}}$.
a_k^{SW}	kth action in $\mathcal{A}^{\mathrm{SW}}$.
A_t	Action at time step t.
A_t^{SW}, A_t^{ED}	Switching and ED action at time step t.
C_1, C_2	Weights that indicate the relative importance of unserved power situation and lost power situation.
c_t^{US}	Cost of aggregated unserved or lost active power.
H_t	History at time step t.
k	Index of switching actions.
k^*,$a_{k^*}^{\mathrm{ED}*}$	Optimal switching and ED action.
m	Simplified switching action.
m^*	Optimal simplified switching action.
O_t	Observation at time step t.
$r(S_t, A_t)$	Reward.
R^π	Cumulative reward.
S_t	System state at time step t.
$\mathcal{A}_m^{\mathrm{SW}}$	Subset within $\mathcal{A}^{\mathrm{SW}}$ that corresponds to m.
$a_k^{\mathrm{ED}*}$	Optimal ED action of $\mathcal{A}_k^{\mathrm{ED}}$.

I. Introduction

MICROGRIDS (MGs) are small-scale low- or medium-voltage distribution networks comprising various on-site generators. Nowadays, Internet of Things (IoT) plays an essential role in MGs [1]. In a smart MG driven by the IoT technologies, the system operator (SO) is deployed in the edge or cloud server and connects with networked sensors embedded in various MG components [2]. The real-time data collected by these sensors are transmitted to the SO through the IoT communication networks to reflect the status of the MG system [3]. By fully analyzing the sensory data, the SO makes intelligent energy management decisions to drive the smart MG devices, such as diesel generators (DGs) [4].

MGs can operate either interconnected or isolated from the main distribution grid [1]. Isolated MGs are deployed in many remote areas all over the world [5], which often rely on a base load generation source, such as reciprocating DGs of heavy fuel oil. As DGs incur high costs of electricity in isolated MGs due to the cost of fuel transportation and delivery, renewable energy sources (RES), such as wind and solar, are a clean complementary power source that greatly reduces the operating costs [6]. In this article, we focus on the optimal scheduling problem in smart isolated MGs, which aims at minimizing the operating cost by fully exploiting the RES power, while satisfying the critical requirement to maintain a balance between power generation and consumption. The main challenge stems from the variability and uncertainty in RES generation and load demand, making it difficult to guarantee energy balance with efficient operations of DGs.

Compared with connected MGs, isolated MGs are less flexible in dealing with the above challenge as they do not have power support from the main grid and usually lack advanced control hardware for demand-side management. Fortunately, the energy storage elements in MGs, such as electrochemical battery, can charge or discharge to compensate for short-term unbalance, and thus help level off the power fluctuations to some extent. Moreover, the battery can act as an additional power source to minimize the operating cost of DGs. As the maximum amount of energy that can be charged or discharged at a certain point of time is limited by the State of Charge (SoC) of the battery, which is, in turn, determined by the previous charging/discharging behavior, optimal scheduling becomes a sequential decision problem for a dynamic system where earlier decisions influence future available choices.

In this article, we leverage deep reinforcement learning (DRL) to learn an optimal scheduling policy from history RES and load data of previous days where the policy can generate real-time decisions based on observations of past RES and load data of previous hours. The main advantage of such an approach is that it does not require prediction models and therefore is not susceptible to prediction errors.

A. Related Works

Energy management for MGs includes several typical functions, i.e., optimal scheduling (such as energy storage management, energy dispatch (ED), unit commitment (UC) [7], etc.), demand response [8], and energy trading [9]. Existing works in this area can be classified into two broad categories, depending on whether the MG is connected [10], [11] or isolated [12], [13]. Our study falls into optimal scheduling in isolated MGs, which has been an extensive study in the literature. To determine the daily energy scheduling operation, various mathematical optimization models have been proposed, where the objectives are normally to minimize the operating cost considering various constraints, such as operational, energy balance, and reserve constraints. Different problem formulations have been studied, such as mixed-integer linear programming (MILP) [14], [15], [16], mixed-integer quadratic programming (MIQP) [17], [18], and mixed-integer and integer-free second-order cone programming (SOCP) [19], [20]. In order to address the challenge of uncertainty in RES generation and load demand, several approaches, such as stochastic programming (SP) [21], model predictive control (MPC) [12], [18], [22], robust optimization [13], [20], and chance-constraint programming (CCP) [23], [24], have been adopted. All of the above methods require forecast models or probabilistic models for the unknown data, where robustness to model inaccuracy is limited.

To avoid the above limitations, intelligent energy management policies can be derived by leveraging reinforcement learning (RL) techniques. Compared with the non-RL methods, RL algorithms can learn the optimal control policies by trial and error based on real-world data and do not require rigorous mathematical models for RES and load [25]. In other words, RL methods can directly learn optimal policies from history data, without the need of prior modeling or parameter identification. DRL is a promising branch of RL, where the powerful deep neural networks (DNNs) make it possible to learn and fit complex patterns better. Moreover, it is easy for DRL methods to solve large-scale optimization problems

in real-time after training the DNNs, since only the forward propagation in the networks is involved.

In recent years, DRL has been adopted for energy management of MGs with the aim for learning optimal policies of demand response [26], [27], [28], energy trading [29], [30], [31], as well as optimal scheduling in both connected MGs [32], [33], [34] and isolated MGs [35], [36]. Ji et al. [32] adopted a classical model-free value-based DRL algorithm, namely, deep Q-network (DQN) [37] to achieve real-time scheduling of a connected MG, which did not require an explicit model or a predictor for unknown data. The real-time dispatch of DGs and battery as well as power transactions with the main grid are optimized based on this approach. Considering a similar online scheduling problem for a connected MG, Shuai and He [33] adopted a model-based DRL algorithm, namely, MuZero, to perform online optimization of MG under uncertainties, which combined a Monte Carlo tree search method with the neural network model. Du and Li [34] considered a scenario where multiple MGs are connected to the main grid, where each MG adopts a model-free Monte Carlo DRL algorithm to learn its retail pricing policy. While the above works focus on connected MGs, DRL-based solutions for isolated MGs have received less attention. François-Lavet et al. [35] presented a value-based DRL algorithm to efficiently operate the storage devices in an isolated MG. However, only three simple discrete actions are considered, i.e., charge, discharge, and idle. Lei et al. [36] proposed new DRL algorithms to solve the ED problem in an isolated MG. The algorithms combined classical actor–critic algorithms, namely, deep deterministic policy gradient (DDPG) [38] and recurrent deterministic policy gradient (RDPG) [39] with dynamic programming (DP) approach and could effectively address the instability problem, finite-horizon setting, and partial observable problem. However, the startup and shutdown of DGs are not considered, which results in inefficient operation.

Based on the review of related works, some important research gaps in the field are identified as follows.

1) In non-RL methods, spinning reserve is usually considered a deterministic [12], [17], [18] or probabilistic constraint [23], [24] to guarantee energy balance under power fluctuations. Deterministic constraint generally leads to inefficient DG operation [23], while the probabilistic constraint is vulnerable to model or parameter deviations.
2) When applying DRL for optimal scheduling of MGs, one important challenge is the discrete-continuous hybrid action space induced by the binary DG switching decision and the continuous ED decision. Classical DRL algorithms are not suitable to handle the hybrid action space. Previous works in DRL usually approximate the hybrid space by discretization [32], [33], which suffers from performance loss.

B. Contributions

The main contributions are summarized as follows.

1) A novel partial observable Markov decision process (POMDP) model is conceived to learn the optimal scheduling policy that can provide sufficient spinning reserve capacity to maintain energy balance while minimizing the operating cost in the IoT-driven isolated MG. Instead of considering spinning reserve as constraints, a penalty for energy imbalance is included in the reward function, and the recalculation of new set-points during real-time operation is captured in the system transition function, so that the DRL agent can learn to set the optimal amount of spinning reserve.
2) A novel DRL algorithm, i.e., hybrid action finite-horizon RDPG (HAFH-RDPG), is proposed. The integration of actor–critic RDPG and value-based DQN characteristics enables our algorithm to have great advantages in dealing with the discrete-continuous hybrid action space. Moreover, the DRL algorithm is embedded within the finite-horizon value iteration framework to improve the convergence stability and performance.
3) The capability of DRL in handling uncertainty is evaluated at two time scales of power fluctuation: a) interhour and b) interday. The capability to deal with interhour fluctuation is assessed by comparing the performance of the proposed algorithm in POMDP and Markov decision process (MDP) environments, respectively. The capability to deal with interday fluctuation is assessed by comparing the performance of the HAFH-RDPG algorithm when it is trained using same-day and previous-days data, respectively. Moreover, we analyze whether the learned policy can make efficient use of the battery in achieving energy shifting and providing sufficient spinning reserve with minimum cost.

The remainder of this article is organized as follows. The IoT-driven isolated MG system model is introduced in Section II. The POMDP model is developed in Section III. The design of the HAFH-RDPG algorithm is presented in Section IV. In Section V, experiments are conducted to verify the efficiency of the HAFH-RDPG algorithm. Finally, the conclusion is drawn in Section VI.

II. System Model

As shown in Fig. 1, the IoT-driven smart isolated MG system considered in this article can be divided into several parts, i.e., the controllable generation units (i.e., DGs), uncontrollable generation units [i.e., photovoltaic cell (PV)], an energy storage device (i.e., battery storage), loads, and an SO (i.e., DRL-based intelligent energy management system) to perform energy scheduling. To reflect the status of the smart MG system, sensors are deployed to collect real-time data on the load power consumption, the output power generated by the PV panels, and the SoC of the battery storage. The sensory data is delivered to the SO in the edge or cloud server, where the DRL agent of the SO processes the data to make intelligent decisions on optimal scheduling of DGs. The information of sensory data and control decisions is transmitted between the SO in edge/cloud server and the smart devices in MG through the IoT communication networks.

The intraday operation of our MG system model is divided into T time steps, indexed by $\{1, \ldots, T\}$. The interval of each time step is denoted as Δt.

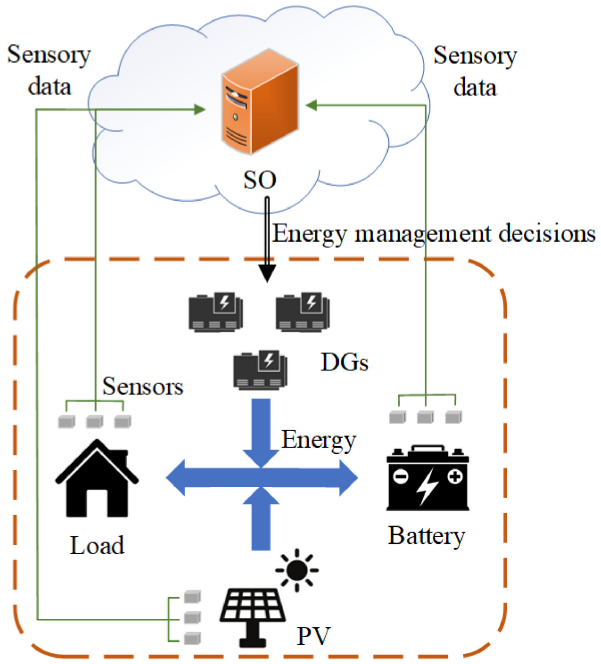

Fig. 1. Schematic of the IoT-driven smart isolated MG system.

A. Diesel Generator Model

Consider there are D DGs in the MG, i.e., $\mathrm{DG}_1, \ldots, \mathrm{DG}_D$, where $D > 1$. Let $B_t^{\mathrm{DG}_d}$ denote the ON/OFF status of DG_d $\forall d \in \{1, \ldots, D\}$, *at the beginning* of time step t, where 1 stands for ON and 0 for OFF.

Let $U_t^{\mathrm{DG}_d}$ denote the binary switching decision of DG_d at time step t. After $U_t^{\mathrm{DG}_d}$ is determined at time step t, the ON/OFF status of DG_d will be set accordingly for the rest of the time step. Note that the ON/OFF status $B_t^{\mathrm{DG}_d}$ at the beginning of time step t is the same as the switching decision $U_{t-1}^{\mathrm{DG}_d}$ at the previous time step $t-1$. Therefore, we have

$$U_{t-1}^{\mathrm{DG}_d} = B_t^{\mathrm{DG}_d}. \tag{1}$$

At time step t, let $P_t^{\mathrm{DG}_d}$ denote the output power of DG_d $\forall d \in \{1, \ldots, D\}$. When a DG is switched on, its output power can be any real number between the minimum and maximum power limits, i.e., $P_{\min}^{\mathrm{DG}}$ and $P_{\max}^{\mathrm{DG}}$. Thus, we have

$$\begin{aligned} P_t^{\mathrm{DG}_d} &= 0, && \text{if } U_t^{\mathrm{DG}_d} = 0 \\ P_{\min}^{\mathrm{DG}} \le P_t^{\mathrm{DG}_d} &\le P_{\max}^{\mathrm{DG}} && \text{if } U_t^{\mathrm{DG}_d} = 1. \end{aligned} \tag{2}$$

B. Battery Model

Let E_t be SoC of the battery at time step t, which is constrained by the minimum and maximum energy levels as

$$E_{\min} \le E_t \le E_{\max}. \tag{3}$$

The SoC of battery E_{t+1} at time step $t+1$ can be calculated based on the SoC state E_t at time step t as

$$E_{t+1} = E_t + \eta_{ch} u_t P_t^E \Delta t - (1 - u_t) P_t^E \Delta t / \eta_{\mathrm{dis}} \tag{4}$$

where η_{ch} and η_{dis} denote the charging and discharging efficiencies of the battery, respectively. P_t^E indicates the charging or discharging power of the battery, and u_t denotes charging or discharging status, which is 0 if the battery is discharging and 1 otherwise.

Let $P_{\max}^E$ be the maximum charging or discharging power of the battery. The charging and discharging power limits of the battery can be calculated separately as

$$P_{\mathrm{ch\_lim}}^E = \min\left(P_{\max}^E, (E_{\max} - E_t)/(\eta_{ch} \Delta t)\right) \tag{5}$$

and

$$P_{\mathrm{dis\_lim}}^E = \min\left(P_{\max}^E, \eta_{\mathrm{dis}}(E_t - E_{\min})/\Delta t\right). \tag{6}$$

C. Energy Balance and Spinning Reserve

At each time step t, the total amount of energy dispatched by the DGs, PV, and battery should meet the load demand whenever possible, i.e.,

$$\sum_{d=1}^{D} P_t^{\mathrm{DG}_d} + P_t^{\mathrm{PV}} + \eta_{ch} u_t P_t^E - (1 - u_t) P_t^E / \eta_{\mathrm{dis}} = P_t^L \tag{7}$$

where P_t^{PV} is the output power of PV, and P_t^L is the load consumption at time step t.

At the beginning of each time step t, a set-point for the DG output power $P_t^{\mathrm{DG}_d\_S}$ is determined by the DRL agent. During real-time operation within the time step, as the actual PV generation P_t^{PV} and load demand P_t^L are observed, corrective actions must be taken to ensure the energy balance. For an isolated MG without power support from the main grid, the spinning reserve is an important resource to compensate for power shortages due to uncertainty in PV generation and load demand. Both the DGs and battery can provide spinning reserve, where the battery is given a priority due to its lower cost and faster response than the DGs. Similarly, the battery is leveraged to cope with power oversupply before adjusting DG output power.

1) Charging/Discharging of Battery: In order to maintain energy balance in (7), the battery needs to follow load, where the charging/discharging power P_t^E and charging/discharging status u_t can be determined as follows. First, a variable δ_t is defined to represent the surplus of generating capacity at time step t assuming the DG output power is set to $P_t^{\mathrm{DG}_d\_S}$, i.e.,

$$\delta_t = \sum_{d=1}^{D} P_t^{\mathrm{DG}_d\_S} + P_t^{\mathrm{PV}} - P_t^L. \tag{8}$$

When a surplus is generated, i.e., $\delta_t > 0$, the excess power is stored in the battery and the battery is in the charging status. On the contrary, when the power generation of PV and DGs is insufficient, i.e., $\delta_t < 0$, the battery is in the discharging status to provide the lacked power. In this way, u_t can be determined based on δ_t as

$$u_t = \begin{cases} 0, & \text{if } \delta_t < 0 \\ 1, & \text{otherwise} \end{cases} \tag{9}$$

and P_t^E can be derived based on δ_t as

$$P_t^E = \begin{cases} \min\left(-\delta_t, P_{\mathrm{dis\_lim}}^E\right), & \text{if } \delta_t < 0 \\ \min\left(\delta_t, P_{\mathrm{ch\_lim}}^E\right), & \text{otherwise.} \end{cases} \tag{10}$$

To elaborate on (10), when $\delta_t < -P^E_{\text{dis\_lim}}$, even discharging the battery cannot provide enough power to meet the load demand. When $\delta_t > P^E_{\text{ch\_lim}}$, the excessive power is beyond the charging capability of the battery. In the above cases, we will need to adjust the output power of DGs. For ease of notation, we denote the unbalanced power after charging/discharging battery as

$$P_t^{\text{ERR}} = \begin{cases} \delta_t + P^E_{\text{dis\_lim}}, & \text{if } \delta_t < -P^E_{\text{dis\_lim}} \\ \delta_t - P^E_{\text{ch\_lim}}, & \text{if } \delta_t > P^E_{\text{ch\_lim}} \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

2) Adjustment of DG Output Power: To maintain energy balance in (7), the total output power of DGs is adjusted subject to the power limits as follows:

$$\sum_{d=1}^{D} P_t^{\text{DG}_d} = \begin{cases} \max\left\{\sum_{d=1}^{D} U_t^{\text{DG}_d} P^{\text{DG}}_{\min}, P_t^{\text{Adjusted}}\right\} & \text{if } P_t^{\text{ERR}} > 0 \\ \min\left\{\sum_{d=1}^{D} U_t^{\text{DG}_d} P^{\text{DG}}_{\max}, P_t^{\text{Adjusted}}\right\} & \text{if } P_t^{\text{ERR}} < 0 \end{cases} \tag{12}$$

where

$$P_t^{\text{Adjusted}} = \sum_{d=1}^{D} P_t^{\text{DG}_d\_S} - P_t^{\text{ERR}}. \tag{13}$$

We define the unbalanced power after DG output power correction as

$$P_t^{\text{UB}} = \sum_{d=1}^{D} P_t^{\text{DG}_d} + P_t^{\text{PV}} + \eta_{ch} u_t P_t^E - (1-u_t) P_t^E / \eta_{\text{dis}} - P_t^L. \tag{14}$$

When $P_t^{\text{UB}} = 0$, energy balance in (7) is achieved. If $P_t^{\text{UB}} > 0$, the power generation is larger than demand, and the excessive power will go to load bank, which will be lost. Large P_t^{UB} beyond the maximum capacity of load bank will cause the undesirable reverse power flow in the MG. If $P_t^{\text{UB}} < 0$, the generation is smaller than demand, which will result in some of the loads unserved.

D. Operating Cost

The operating cost includes four parts. For any DG_d $\forall d \in \{1, \ldots, D\}$, the DG power generation cost $c_t^{\text{DG}_d}$ can be derived by the conventional quadratic cost function in (15). The start-up cost $c_t^{\text{S\_DG}_d}$ in (16) and the running cost $c_t^{\text{R\_DG}_d}$ in (17) are penalty costs associated with turning on DG_d and the ON status of DG_d, respectively. Finally, the spinning reserve cost $c_t^{\text{SR\_DG}_d}$ in (18) depends on the spinning reserve capacity, i.e., the amount of unused capacity, provided by DG_d

$$c_t^{\text{DG}_d} = \left[a_d \left(P_t^{\text{DG}_d}\right)^2 + b_d P_t^{\text{DG}_d} + c_d\right] \Delta t \quad \forall d \in \{1, \ldots, D\} \tag{15}$$

$$c_t^{\text{S\_DG}_d} = C_S U_t^{\text{DG}_d} \left(1 - B_t^{\text{DG}_d}\right) \Delta t \quad \forall d \in \{1, \ldots, D\} \tag{16}$$

$$c_t^{\text{R\_DG}_d} = C_R U_t^{\text{DG}_d} \Delta t \quad \forall d \in \{1, \ldots, D\} \tag{17}$$

$$c_t^{\text{SR\_DG}_d} = C_{\text{SR}} \left(U_t^{\text{DG}_d} P^{\text{DG}}_{\max} - P_t^{\text{DG}_d}\right) \Delta t. \tag{18}$$

Note that in (15)–(18), a_d, b_d, c_d, C_S, C_R, and C_{SR} are the coefficients, where a_d, b_d, and c_d are fitted to fuel consumption curves for DGs; while C_S, C_R, and C_{SR} are weights that indicate the relative importance of the start-up cost, the running cost, and the spinning reserve cost, respectively.

Therefore, the operating cost c_t^{OP} is the sum of the above costs for all the DGs, i.e.,

$$c_t^{\text{OP}} = \sum_{d=1}^{D} \left(U_t^{\text{DG}_d} c_t^{\text{DG}_d} + c_t^{\text{S\_DG}_d} + c_t^{\text{R\_DG}_d} + c_t^{\text{SR\_DG}_d}\right). \tag{19}$$

Note that (12) only provides the total DG output power after adjustment. The output power of individual DGs can be derived by minimizing the operating cost c_t^{OP} in (19) after the total output power is determined.

III. POMDP Model

In this section, we formulate a finite-horizon POMDP model based on which the optimal scheduling decisions using DRL can be made. The objective of our POMDP model is to minimize the operating cost and ensure the supply–demand balance, where the uncertainty of both fluctuating loads and stochastic generation of PV are taken into account.

A. State and Observation

Let $S_t = (P_t^{\text{EL}}, E_t, \{B_t^{\text{DG}_d}\}_{d=1}^D)$ be the system state at time step t $\forall t \in \{1, \ldots, T\}$, where $P_t^{\text{EL}} = P_t^L - P_t^{\text{PV}}$ is the equivalent load. Due to the uncertainties of load consumption and PV generation, the agent is unable to observe P_t^{EL} at the beginning of time step t, but receives observation $O_t = (P_{t-1}^{\text{EL}}, E_t, \{B_t^{\text{DG}_d}\}_{d=1}^D)$ instead. Note that $P_{t-1}^{\text{EL}} = P_{t-1}^L - P_{t-1}^{\text{PV}}$ is the equivalent load at time step $t-1$.

B. Action

Let $A_t = (A_t^{\text{SW}}, A_t^{\text{ED}})$ be the action at time step $t \in \{1, \ldots, T\}$, where $A_t^{\text{SW}} = \{U_t^{\text{DG}_d}\}_{d=1}^D$ denotes the DG switching action, which is a vector of *discrete* variables. $A_t^{\text{ED}} = \{P_t^{\text{DG}_d\_S}\}_{d=1}^D$ represents the ED action that determines the DG output power set-point, which is a vector of *continuous* variables.

The action space $\mathcal{A} = \mathcal{A}^{\text{SW}} \times \mathcal{A}^{\text{ED}}$. As $\mathcal{A}^{\text{SW}}$ is D permutations of 0 and 1, its size is 2^D. Let a_k^{SW} denote the kth action in the switching action space $\mathcal{A}^{\text{SW}}$, where $k \in \{1, \ldots, 2^D\}$ is the index of the switching action. Since there is a one-to-one correspondence between a_k^{SW} and k, we substitute k for a_k^{SW} for convenience in the rest of this article.

According to (2), $\mathcal{A}^{\text{ED}}$ is related to the switching action k. Therefore, let $\mathcal{A}_k^{\text{ED}}$ denote the ED action space corresponding to the chosen switching action k. Let a_k^{ED} denote an action belonging to the ED action space $\mathcal{A}_k^{\text{ED}}$, i.e., $a_k^{\text{ED}} \in \mathcal{A}_k^{\text{ED}}$.

C. History

As only the observation O_t is available instead of state S_t, the agent is provided with history H_t to derive the action A_t. The history is tailored for the optimal scheduling of isolated MG and defined as $H_t = (P_{t-\tau}^{\text{EL}},$

$P^{\mathrm{EL}}_{t-\tau+1}, \ldots, P^{\mathrm{EL}}_{t-1}, E_t, \{B^{\mathrm{DG}_d}_t\}^{D}_{d=1})$ $\forall t \in \{1, \ldots, T\}$, where τ is the length of the time window to look into the past. Note that H_t includes history data of equivalent load statistics $\{P^{\mathrm{EL}}_{t'}\}^{t-1}_{t'=t-\tau}$ to implicitly predict the equivalent load state P^{EL}_t at time step t. Let $\mathcal{H}$ denote the history space.

D. Reward Function

The optimization objective in the IoT-driven MG system is to minimize the operating cost within the considered time horizon, i.e., one day while guaranteeing energy balance. Therefore, we define the reward function as

$$r(S_t, A_t) = -\left(c^{\mathrm{OP}}_t + c^{\mathrm{US}}_t\right) \tag{20}$$

where c^{OP}_t is the operating cost in (19). c^{US}_t represents the cost of aggregated unserved or lost active power, i.e.,

$$c^{\mathrm{US}}_t = \begin{cases} C_1 P^{\mathrm{UB}}_t \Delta t, & \text{if } P^{\mathrm{UB}}_t > 0 \\ -C_2 P^{\mathrm{UB}}_t \Delta t, & \text{if } P^{\mathrm{UB}}_t < 0 \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

where C_1 and C_2 are weights that indicate the relative importance of unserved power situation and lost power situation, and P^{UB}_t can be derived according to (14).

E. Transition Probability

The state transition probability is derived as

$$\begin{aligned} \Pr(S_{t+1}|S_t, A_t) = \Pr\left(P^L_{t+1}|P^L_t\right)\Pr\left(P^{\mathrm{PV}}_{t+1}|P^{\mathrm{PV}}_t\right) \\ \Pr\left(E_{t+1}|E_t, P^L_t, P^{\mathrm{PV}}_t, A^{\mathrm{ED}}_t\right)\Pr\left(B_{t+1}|A^{\mathrm{SW}}_t\right) \end{aligned} \tag{22}$$

where the transition probabilities of load demands $\Pr(P^L_{t+1}|P^L_t)$ and PV power outputs $\Pr(P^{\mathrm{PV}}_{t+1}|P^{\mathrm{PV}}_t)$ are not available, but samples of the trajectory can be obtained from real-world data. The transition probability of SoC $\Pr(E_{t+1}|E_t, P^L_t, P^{\mathrm{PV}}_t, A^{\mathrm{ED}}_t)$ can be calculated through (4) and (8)–(10). The last transition probability $\Pr(B_{t+1}|A^{\mathrm{SW}}_t)$ is for the switching state, which can be easily inferred from (1).

Similarly, the history transition probability is derived as

$$\begin{aligned} \Pr(H_{t+1}|H_t, A_t) = \Pr\left(P^L_t|P^L_{t-\tau}, \ldots, P^L_{t-1}\right) \\ \Pr\left(P^{\mathrm{PV}}_t|P^{\mathrm{PV}}_{t-\tau}, \ldots, P^{\mathrm{PV}}_{t-1}\right)\Pr\left(B_{t+1}|A^{\mathrm{SW}}_t\right) \\ \Pr\left(E_{t+1}|E_t, P^L_t, P^{\mathrm{PV}}_t, A^{\mathrm{ED}}_t\right) \end{aligned} \tag{23}$$

where the transition probabilities of load demands $\Pr(P^L_t|P^L_{t-\tau}, \ldots, P^L_{t-1})$ and PV power outputs $\Pr(P^{\mathrm{PV}}_t|P^{\mathrm{PV}}_{t-\tau}, \ldots, P^{\mathrm{PV}}_{t-1})$ can be obtained by the samples of the trajectory from real-world data. The transition probability of SoC $\Pr(E_{t+1}|E_t, P^L_t, P^{\mathrm{PV}}_t, A^{\mathrm{ED}}_t)$ and the transition probability of switching state $\Pr(B_{t+1}|A^{\mathrm{SW}}_t)$ are the same as the transition probabilities in (22).

F. Optimization Objective

We define the cumulative reward R^π as the sum of rewards over the finite horizon under a given policy π, i.e.,

$$R^\pi = \sum_{t=1}^{T} r(S_t, A_t). \tag{24}$$

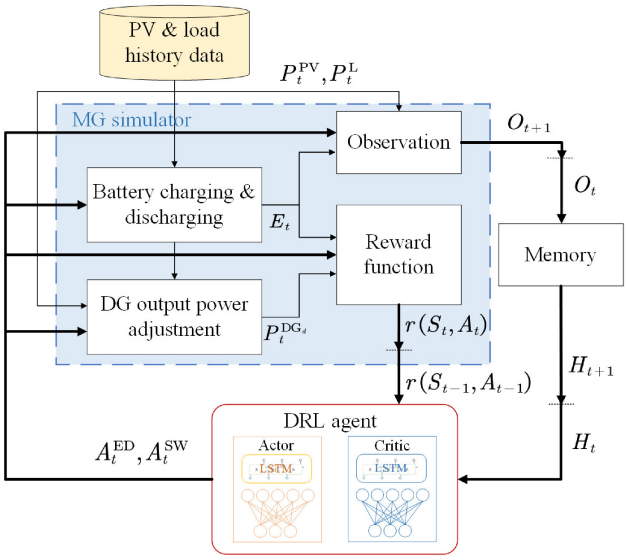

Fig. 2. Schematic description of the learning process of the DRL agent.

Our objective is to derive the optimal policy π^* to maximize the expected cumulative reward, i.e.,

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[R^\pi\right]. \tag{25}$$

IV. DRL Solution

In this section, we present a DRL algorithm to solve the POMDP model formulated in Section III. A schematic description of the learning process is provided in Fig. 2. The DRL agent is trained using history PV and load data and can be used for real-time optimal scheduling after training.

A. Dealing With the Hybrid Action Space

Since the action A_t in our POMDP model includes both continuous and discrete variables, the action space $\mathcal{A}$ is a discrete-continuous hybrid space, while most of the existing DRL algorithms require either a discrete space or a continuous space. Inspired by the P-DQN algorithm in [40], we break the optimization problem down into two steps and design a new algorithm to solve this problem.

First, the optimization objective in (25) is equivalent to finding the policy π^* that can maximize the action value $Q(H_t, k, a^{\mathrm{ED}}_k)$, i.e.,

$$\pi^*(H_t) = \left(k^*, a^{\mathrm{ED}*}_{k^*}\right) = \arg\max_{k\in\{1,\ldots,2^D\}, a^{\mathrm{ED}}_k \in \mathcal{A}^{\mathrm{ED}}_k} Q\left(H_t, k, a^{\mathrm{ED}}_k\right) \tag{26}$$

where k^* and $a^{\mathrm{ED}*}_{k^*}$ denote the optimal switching action and ED action, respectively.

For the discrete switching action k, there are 2^D choices. For any given switching action $k \in \{1, \ldots, 2^D\}$, we are able to select the best ED action $a^{\mathrm{ED}*}_k$ from the ED action space $\mathcal{A}^{\mathrm{ED}}_k$ by solving

$$a^{\mathrm{ED}*}_k = \arg\max_{a^{\mathrm{ED}}_k \in \mathcal{A}^{\mathrm{ED}}_k} Q\left(H_t, k, a^{\mathrm{ED}}_k\right) \quad \forall k \in \left\{1, \ldots, 2^D\right\}. \tag{27}$$

However, as a_k^{ED} is a continuous variable, (27) cannot be directly solved by classical value-based algorithms, such as Q-learning and DQN [40], unless the action space is discretized, which will result in performance loss due to discretization error. In contrast, the actor–critic algorithms can efficiently handle the continuous action space. Therefore, our algorithm adopts the actor–critic framework to calculate the optimal ED action $a_k^{\mathrm{ED}*}$.

Specifically, for a given switching action $k \quad \forall k \in \{1, \ldots, 2^D\}$, an actor network $\mu_k(\cdot;\theta) : \mathcal{H} \to \mathcal{A}_k^{\mathrm{ED}}$ is introduced to approximate the optimal ED action $a_k^{\mathrm{ED}*}$ from the continuous space $\mathcal{A}_k^{\mathrm{ED}}$ such that

$$a_k^{\mathrm{ED}*} \approx \mu_k(H_t;\theta), \tag{28}$$

where θ is the weights of μ_k. Correspondingly, a critic network $\lambda_k(\cdot; w)$ is applied to evaluate the ED action $a_k^{\mathrm{ED}*}$ and approximate the action value $Q(H_t, k, a_k^{\mathrm{ED}*})$, i.e.,

$$Q\left(H_t, k, a_k^{\mathrm{ED}*}\right) \approx \lambda_k\left(H_t, a_k^{\mathrm{ED}*}; w\right) \tag{29}$$

where w denotes the weights of λ_k.

Once the optimal ED action $a_k^{\mathrm{ED}*}$ for all the switching action $k \quad \forall k \in \{1, \ldots, 2^D\}$, are derived, we could choose the optimal switching action $\hat{k}^*$ from the 2^D-dimensional discrete switching action space. The above action value $Q(H_t, k, a_k^{\mathrm{ED}*})$ given by the critic network λ_k can be used to evaluate the switching action k when the corresponding optimal ED action $a_k^{\mathrm{ED}*}$ is applied. We choose the switching action k with the largest action value as the optimal solution, i.e.,

$$\hat{k}^* = \underset{k\in\{1,\ldots,2^D\}}{\arg\max}\ Q\left(H_t, k, a_k^{\mathrm{ED}*}\right). \tag{30}$$

As shown in Fig. 3, the above two-step framework is summarized as follows.

1) For each switching actions $k \in \{1, \ldots, 2^D\}$, select the corresponding optimal ED action $a_k^{\mathrm{ED}*}$ by (27) in the continuous action space.
2) Derive the optimal switching action $\hat{k}^*$ in the discrete action space according to (30).

In the following, we present Theorem 1 that states the optimal policy for the objective in (26) can be derived by the proposed two-step framework.

Theorem 1: For any history H_t, the action $(\hat{k}^*, a_{\hat{k}^*}^{\mathrm{ED}*})$ derived by the two-step framework is as good as the optimal action $\pi^*(H_t)$ for the objective in (26), i.e.,

$$Q\left(H_t, \hat{k}^*, a_{\hat{k}^*}^{\mathrm{ED}*}\right) = Q\left(H_t, k^*, a_{k^*}^{\mathrm{ED}*}\right). \tag{31}$$

The proof of Theorem 1 is given in Appendix A.

In order to train the actor and critic networks, we focus on the action value function $Q(H_t, A_t) = Q(H_t, A_t^{\mathrm{SW}}, A_t^{\mathrm{ED}})$ based on the Bellman equation as

$$\begin{aligned} Q\left(H_t, A_t^{\mathrm{SW}}, A_t^{\mathrm{ED}}\right) &= \mathop{\mathbb{E}}_{S_t|H_t}\left[r\left(S_t, A_t^{\mathrm{SW}}, A_t^{\mathrm{ED}}\right)\right] \\ &+ \mathop{\mathbb{E}}_{H_{t+1}|H_t, A_t^{\mathrm{SW}}, A_t^{\mathrm{ED}}}\left[\gamma \cdot \max_{k\in\{1,\ldots,2^D\}} \max_{a_k^{\mathrm{ED}}\in\mathcal{A}_k^{\mathrm{ED}}} Q\left(H_{t+1}, k, a_k^{\mathrm{ED}}\right)\right] \end{aligned} \tag{32}$$

where γ is the discount factor that satisfies $0 < \gamma \leq 1$.

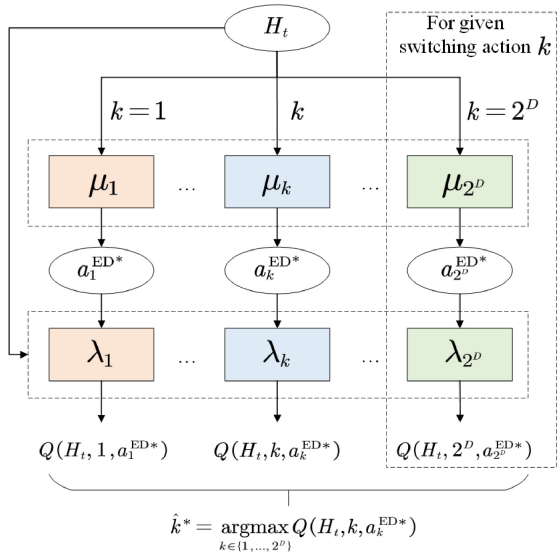

Fig. 3. Neural network framework of HAFH-RDPG.

Thus, the loss function $L_k(w)$ of the critic network $\lambda_k \quad \forall k \in \{1, \ldots, 2^D\}$ can be calculated as the mean-square error, i.e.,

$$L_k(w) = \mathop{\mathbb{E}}_{H_t\sim\rho_k^{\mu}, a_k^{\mathrm{ED}}\sim\mu_k}\left[\left(y_{t,k} - \lambda_k\left(H_t, a_k^{\mathrm{ED}}; w\right)\right)^2\right] \tag{33}$$

where ρ_k^{μ} denotes the discounted state visitation distribution following policy μ_k. $y_{t,k}$ denotes the target for each critic network λ_k at time step t, which is defined based on the Bellman equation in (32) as

$$y_{t,k} \approx r_t + \gamma \max_{k\in\{1,\ldots,2^D\}} \lambda_k(H_{t+1}, \mu_k(H_{t+1};\theta); w). \tag{34}$$

By minimizing the loss function $L_k(w)$ via stochastic gradient descent, the weight w of λ_k can be updated at each time step by

$$w \leftarrow w + 2\beta\left(y_{t,k} - \lambda_k\left(H_t, a_k^{\mathrm{ED}}; w\right)\right)\nabla_w \lambda_k\left(H_t, a_k^{\mathrm{ED}}; w\right) \tag{35}$$

where β is the learning rate for updating w.

The loss function of actor network $\mu_k \quad \forall k \in \{1, \ldots, 2^D\}$ can be defined as

$$L_k(\theta) \approx -\mathop{\mathbb{E}}_{H_0\sim\rho^0}[\lambda_k(H_0, \mu_k(H_t;\theta); w)] \tag{36}$$

where ρ_0 is the distribution of the initial history H_0.

We use stochastic gradient decent to sample deterministic policy gradient [41], and the weight θ of μ_k is updated by

$$\theta \leftarrow \theta - \alpha\nabla_\theta \mu_k(H_t;\theta)\nabla_{a_k^{\mathrm{ED}}}\lambda_k\left(H_t, a_k^{\mathrm{ED}}; w\right)|_{a_k^{\mathrm{ED}}=\mu_k(H_t;\theta)} \tag{37}$$

where α is the learning rate for updating θ.

B. Designing the Framework

As we focus on energy management within one day with T time steps, our task corresponds to a finite-horizon POMDP model. There are a few popular DRL algorithms for solving POMDP, such as deep recurrent Q-learning (DRQN) [42] and RDPG. The basic ideas of DRQN and RDPG are to add recurrency to DQN and DDPG algorithms, respectively, by

replacing the first fully connected layer with a recurrent long short-term memory (LSTM) layer. However, the above DRL algorithms are developed for an infinite-horizon setting, while the value function, i.e., critic, and the policy, i.e., actor, are normally dependent on the time steps for the finite-horizon case. Therefore, we propose a novel DRL algorithm named HAFH-RDPG, which adopts a framework that is a combination of DP and DRL, where RDPG with a fixed target is embedded within the framework of finite-horizon value iteration.

In HAFH-RDPG, the finite-horizon value iteration starts from the time step T, and uses backward induction to iteratively derive the value function and optimal policy for each time step $t \in \{T, T-1, \ldots, 1\}$, until it reaches the first time step $t = 1$. In each time step, the RDPG algorithm is used to solve a simple one-step POMDP where the target actor and critic networks, i.e., λ_t' and μ_t', are fixed to be the trained actor and critic networks of the next time step, i.e., λ_{t+1} and μ_{t+1}, which greatly increases stability and performance.

Algorithm 1 describes the pseudocode of HAFH-RDPG. The calculation of the hybrid actions and the method to update networks are presented by two functions separately, i.e., Algorithms 2 and 3. In the Action Calculation function, we introduce multiple actor networks to calculate the optimal ED actions for every switching action. According to (30), we choose the optimal switching action whose critic network has the largest output value and the corresponding ED action. In the Networks Updating function, in order to update the critic networks, the minibatch of transitions $\mathcal{B}$ are further divided into nonoverlapping subsets according to the switching actions. The critic network λ_k is updated by a minibatch of transitions $\mathcal{J}$ in which the switching action of each transition is k. After that, the actor networks can be updated based on $\mathcal{B}$ using the sampled policy gradient.

The proposed HAFH-RDPG algorithm works for MG systems with an arbitrary number of DGs. When the number of DGs changes, we only need to add or remove actor–critic pairs correspondingly and the training process will be in the same way as before.

C. Reducing the Algorithm Complexity

Since the proposed method is a DRL-based algorithm, it updates the network parameters instead of the Q values of all state-action pairs in every iteration. The computational complexity of training each pair of actor–critic networks in HAFH-RDPG is the same as that of the classical RDPG algorithm, which is associated with the size of neural networks instead of the state and action spaces. In specific, the per-iteration computational complexity is mainly dominated by the matrix multiplication in the LSTM layer and fully connected layers. However, note that the HAFH-RDPG algorithm requires $|\mathcal{A}^{\mathrm{SW}}| = 2^D$ pairs of actor and critic networks, thus the computational complexity is $O(2^D)$. As the number of DGs D increases, the number of actor and critic networks and thus the computational complexity of HAFH-RDPG increases exponentially.

In order to reduce the computational complexity, we map the action space to a new space with reduced dimensionality.

Algorithm 1 HAFH-RDPG Algorithm

Randomly initialize actor networks $\mu_k(h;\theta_k)$ and critic networks $\lambda_k(h,a;w_k)$ with weights $\theta_k = \theta_k^0$ and $w_k = w_k^0$, respectively, where $\forall k \in \{1,\ldots,2^D\}$. Initialize target networks $\mu_k'(h;\theta_k')$ and $\lambda_k'(h,a;w_k')$ with $\theta_k' \leftarrow \theta_k$ and $w_k' \leftarrow w_k$;
for $t = T, \ldots, 1$ **do**
  Initialize replay buffer $\mathcal{R}$
  Initialize a random process $\mathcal{N}$ for action exploration
  for episode $e = 1, \ldots, M$ **do**
    Receive history $H_t^{(e)}$
    $k_t^{(e)}, A_t^{\mathrm{ED}(e)}$=ACTION($H_t^{(e)}$,$\mu_k(h;\theta_k)$,$\lambda_k(h,a;w_k)$)
    Execute action and observe reward $r_t^{(e)}$ and next observation $O_{t+1}^{(e)}$
    Store transition $\left(H_t^{(e)}, k_t^{(e)}, A_t^{\mathrm{ED}(e)}, r_t^{(e)}, O_{t+1}^{(e)}\right)$ into $\mathcal{R}$
    Sample a random minibatch of N transitions $\mathcal{B} = \left(H_t^{(i)}, k_t^{(i)}, A_t^{\mathrm{ED}(i)}, r_t^{(i)}, O_{t+1}^{(i)}\right)$ from $\mathcal{R}$
    if $t = T$ **then**
      Set the target $y_T^{(i)} = r_T^{(i)}$
    else
      Construct $H_{t+1}^{(i)}$
      Set the target

$$y_t^{(i)} = r_t^{(i)} + \gamma \cdot \max_{k\in\{1,\ldots,2^D\}} \lambda_k'\left(H_{t+1}^{(i)}, \mu_k'\left(H_{t+1}^{(i)};\theta_k\right);w_k\right)$$

    end if
    UPDATE($\mathcal{B}$,$y_t^{(i)}$,$\mu_k(h;\theta_k)$,$\lambda_k(h,a;w_k)$)
  end for
  Update the target networks: $w_k' \leftarrow w_k$, $\theta_k' \leftarrow \theta_k$
  Save weight of the actor networks: $\theta_{k,t} \leftarrow \theta_k$
  Reset weight of the actor and critic networks to initial value: $\theta_k \leftarrow \theta_k^0$, $w_k \leftarrow w_k^0$
end for

Algorithm 2 Action Calculation

function ACTION($H_t^{(e)}$, $\mu_k(h;\theta_k)$, $\lambda_k(h,a;w_k)$)
  for switching action $k = 1, \ldots, 2^D$ **do**
    Calculate ED action $a_k^{\mathrm{ED}*(e)} = \mu_k\left(H_t^{(e)};\theta_k\right)$
  end for
  Select switching action $k_t^{(e)} = \arg\max_{k\in\{1,\ldots,2^D\}} \lambda_k\left(H_t^{(e)}, a_k^{\mathrm{ED}*(e)};w_k\right)$
  Select ED action $A_t^{\mathrm{ED}(e)} = a_{k_t^{(e)}}^{\mathrm{ED}*(e)}$
  Adjust switching action $k_t^{(e)}$ based on ϵ-greedy policy
  Adjust ED action $A_t^{\mathrm{ED}(e)}$ with exploration noise
  return $k_t^{(e)}$ and $A_t^{\mathrm{ED}(e)}$
end function

For simplicity of explanation, we consider that the parameters of all D DGs are identical in the following. When the parameters of only a subset of DGs are identical, our method can be applied to the subset of DGs instead of all the DGs. The new switching action space $\hat{\mathcal{A}}^{\mathrm{SW}}$ is defined as

Algorithm 3 Networks Updating

1: **function** UPDATE($\mathcal{B}$, $y_t^{(i)}$, $\mu_k(h;\theta_k)$, $\lambda_k(h,a;w_k)$)
2: **for** $k = 1, \ldots, 2^D$ **do**
3: Sample the minibatch of transitions $\mathcal{J} = \left(H_t^{(j)}, k_t^{(j)}, A_t^{\mathrm{ED}(j)}, r_t^{(j)}, O_{t+1}^{(j)} \mid k_t^{(j)} = k\right)$ from $\mathcal{B}$ and corresponding targets $y_t^{(j)}$
4: Update the critic networks by minimizing the loss:

$$L_k = \frac{1}{|\mathcal{J}|}\sum_{j\in\mathcal{J}}\left(y_t^{(j)} - \lambda_k\left(H_t^{(j)}, A_t^{\mathrm{ED}(j)}; w_k\right)\right)^2,$$

$$w_k \leftarrow w_k + \beta\nabla_{w_k} L_k$$

5: **end for**
6: Update the actor networks using the sampled policy gradient:

$$\nabla_{\theta_k} J_k \approx \frac{1}{|\mathcal{B}|}\sum_i \Big[\nabla_a \lambda_k(h,a;w_k)|_{h=H_t^{(i)}, a=\mu_k\left(H_t^{(i)};\theta_k\right)} \nabla_{\theta_k}\mu_k\left(H_t^{(i)};\theta_k\right)\Big], \theta_k \leftarrow \theta_k + \alpha\nabla_{\theta_k} J_k.$$

7: **end function**

$\hat{\mathcal{A}}^{\mathrm{SW}} = \{0, 1, \ldots, D\}$, where a simplified switching action in the new space $m \in \hat{\mathcal{A}}^{\mathrm{SW}}$ corresponds to the number of DGs that are ON. Compared with $\mathcal{A}^{\mathrm{SW}}$, the size of $\hat{\mathcal{A}}^{\mathrm{SW}}$ is reduced from 2^D to $D+1$, which can, in turn, reduce the number of neural networks and the computation complexity from $O(2^D)$ to $O(D)$.

Note that there is a one-to-many relationship between m and k. Specifically, each m corresponds to C_D^m possible switching actions k. When m DGs should be ON, the corresponding switching actions k form a subset $\mathcal{A}_m^{\mathrm{SW}}$ within $\mathcal{A}^{\mathrm{SW}}$. In other words, $\mathcal{A}^{\mathrm{SW}}$ can be divided into $D+1$ nonoverlapping subsets, i.e.,

$$\bigcup_{m=0}^{D} \mathcal{A}_m^{\mathrm{SW}} = \mathcal{A}^{\mathrm{SW}}. \tag{38}$$

In the following, we introduce a strategy that maps a simplified switching action $m \in \hat{\mathcal{A}}^{SW}$ to a unique switching action $k_m \in \mathcal{A}_m^{\mathrm{SW}}$.

DG Selection Strategy: First, a priority value is assigned for each DG_d $\forall d \in \{1, 2, \ldots, D\}$, so that DG_1 has the highest priority and DG_D has the lowest. If m DGs should be ON at time step t where $\forall m \in \{0, 1, \ldots, D\}$, the switching action under this strategy k_m should be

$$k_m = \left(\left\{U_t^{\mathrm{DG}_d}\right\}_{d=1}^{m} = 1, \left\{U_t^{\mathrm{DG}_d}\right\}_{d=m+1}^{D} = 0\right). \tag{39}$$

With the above strategy, we can first map each m to k_m, and then choose the optimal m^* as

$$m^* = \arg\max_{m\in\hat{\mathcal{A}}^{\mathrm{SW}}} Q\left(H_t, k_m, a_{k_m}^{\mathrm{ED}*}\right). \tag{40}$$

Specifically, the HAFH-RDPG algorithm given in Algorithms 1–3 can be used to train the actor and critic networks by replacing the switching action k with the simplified switching action m. However, in the eighth line of Algorithms 1, $m_t^{(e)}$ should first be mapped to $k_t^{(e)}$ by the DG selection strategy before its execution to derive the reward and next state. Since each m corresponds to a unique k_m, the reward $r(S_t, m, a_m^{\mathrm{ED}*})$ and transition probability $\Pr(H_{t+1}|H_t, m, a_m^{\mathrm{ED}*})$ can be derived by replacing m with k_m in the expressions. Finally, the optimal simplified switching action m^* is mapped to the switching action k_{m^*}.

We now provide Theorem 2 which elaborates that the two switching actions, i.e., k^* derived by (30) and k_{m^*} with m^* derived by (40), achieve the same performance.

Theorem 2: Suppose that all the DG parameters are identical, the POMDPs with action space $\hat{\mathcal{A}}^{\mathrm{SW}}$ and $\mathcal{A}^{\mathrm{SW}}$ have the same optimal performance given that the above DG selection strategy is used to determine the m DGs that are ON, i.e.,

$$\max_{m\in\hat{\mathcal{A}}^{\mathrm{SW}}} Q\left(H_t, k_m, a_{k_m}^{\mathrm{ED}*}\right) = \max_{k\in\mathcal{A}^{\mathrm{SW}}} Q\left(H_t, k, a_k^{\mathrm{ED}*}\right). \tag{41}$$

TABLE I
PARAMETER CONFIGURATION IN THE MG SYSTEM MODEL

Notations	Values	Description
D	3	The number of DGs
$P_{\min}^{\mathrm{DG}}$ / $P_{\max}^{\mathrm{DG}}$	60kW / 300kW	The minimum/maximum power constraint of a DG
$P_{\max}^{\mathrm{E}}$	200kW	The power constraint of the battery
$E_{\min}$ / $E_{\max}$	24kWh / 600kWh	The minimum / maximum storage constraint of the battery
$\eta_{\mathrm{ch}}/\eta_{\mathrm{dis}}$	0.98	The charging/discharging efficiency
$a_d/b_d/c_d$	0.0000381 / 0.1887 / 3.8571	The quadratic / monomial / constant coefficient of cost curve
C_{R} / C_{S} / C_{SR}	20 / 10 / 0.25	The fixed running / start-up / spinning reserve cost coefficient

V. NUMERICAL ANALYSIS

In order to evaluate the effectiveness of the proposed algorithm, we perform experiments based on real-world data in an IoT-driven MG and discuss about the simulation results. All the experiments are performed on a Linux server, where the DRL algorithms are implemented in Tensorflow 1.14 using Python.

A. Experimental Setup

The MG system simulated in our experiments comprises three DGs, a PV panel, and a battery. The setting of three DGs is commonly adopted for experiments on isolated MGs [12], [23], [24]. The parameters of the system model are listed in Table I. Note that different values of coefficients C_R, C_S, and C_{SR} can result in different optimal policies, since varying these coefficients will change the weights of the DG power generation cost $c_t^{\mathrm{DG}_d}$, the start-up cost $c_t^{\mathrm{S\_DG}_d}$, running cost $c_t^{\mathrm{R\_DG}_d}$, and the spinning reserve cost $c_t^{\mathrm{SR\_DG}_d}$ in the operating cost c_t^{OP} according to (15)–(19). Based on our setting, the DG generation cost occupies the largest proportion in the operating cost, followed by the spinning reserve cost, the running cost, and the start-up cost. Meanwhile, all the above costs are in the same order of magnitude.

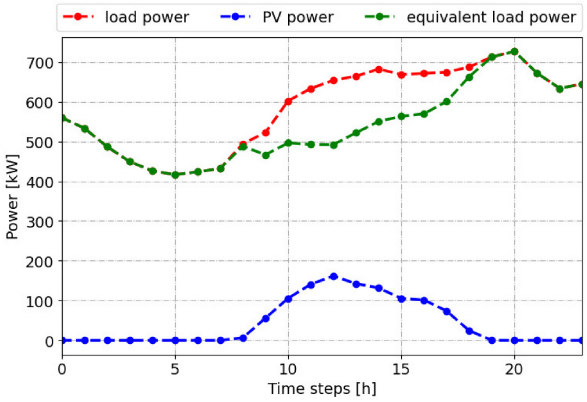

Fig. 4. Trajectories of the PV power, load power, and equivalent load power of one typical day in the experimental data.

1) Experimental Data: We use PV and load statistics in a real IoT-driven isolated MG as the experimental data. These data are collected by the sensors per hour. Therefore, the duration of a time step is set to 1 h, i.e., $\Delta t = 1$ h. Each data at a time step include two values, representing the average load power and average PV power within the corresponding hour. Fig. 4 shows the trajectories of the PV power, load power, and equivalent load power of one typical day in the experimental data. By observing the fluctuations over time in the curve, we find that the PV power is nonnegative between 8:00 and 18:00 with a peak value of 180 kW at around 12:00; while the load power drops to the lowest value of 400 kW between 4:00 and 6:00 and reaches a peak value of 700 kW between 18:00 and 21:00 We regard one-day data as an episode, where the total number of time steps is $T = 24$. In order to evaluate the capability of the proposed algorithm in dealing with interday power fluctuation, two types of data sets are designed for the experiment.

1) *Type A:* The data of one single day randomly sampled from the data set are used as both the training and test set.
2) *Type B:* The data of N continuous days randomly sampled from the data set are used as the training set, while the test set comprises data of the next $((N+1)$th) day.

Type A can provide a benchmark scenario, where there is no uncertainty due to interday power fluctuation. Meanwhile, Type B is used to evaluate the performance of the proposed algorithm in a practical scenario. By comparing the performance of the proposed algorithm under the two data sets, we can evaluate whether the algorithm is able to learn a good policy from history data, which is generalized enough to work well for the next day with unknown PV generation and load demand.

In addition, for both Types A and B, we randomly select five sets of data on different days and train the DRL algorithms in five runs. By comparing the performance of the proposed algorithm under different runs, the stability of the proposed algorithm can be evaluated.

2) DRL Environment: In order to evaluate the capability of the proposed algorithm in dealing with interhour power fluctuation, two types of DRL environments are implemented in the experiments.

1) *MDP:* The state information, including PV and load data of the current time step, is available at the beginning of a time step for making decisions.
2) *POMDP:* Only the PV and load data of the previous τ time steps are available at the beginning of a time step, which is regarded as the history data for making decisions.

The MDP environment provides a benchmark scenario, where there is no uncertainty due to interhour power fluctuation. Meanwhile, the POMDP environment captures interhour power fluctuation in a practical scenario. Comparing the performance of the proposed algorithm under the two environments can prove whether the proposed algorithm is able to deal with partial observability problem and works well when PV and load data for the coming hour is unknown.

3) Benchmark Algorithms: Several benchmark algorithms are selected for performance comparison with the proposed algorithm.

1) *Myopic Algorithm:* The action A_t is directly optimized by minimizing the reward function $r_t(S_t, A_t)$ in (20) without foreseeing the impact on future rewards at each time step $t \in \{1, 2, \ldots, T\}$.
2) *Discrete Dynamic Programming (DDP) [12]:* The finite-horizon value iteration algorithm in DP, where the states and actions are discretized. DDP guarantees to find the global optimum for the MDP after discretization, although performance loss could be incurred for the original MDP depending on the discretization granularity.
3) *Hybrid Action Finite-Horizon DDPG (HAFH-DDPG):* The version of HAFH-RDPG for MDP environment, where the LSTM layer is replaced by a fully connected layer in actor and critic networks.
4) *DRQN [42]:* A popular value-based DRL algorithm for solving POMDP problem. To deploy DRQN, the action space needs to be discretized at proper granularity.

The hyperparameters of the DRL algorithms are listed in Table II.

TABLE II
HYPERPARAMETERS OF THE DRL ALGORITHMS

Parameters	Values		
	HAFH-DDPG	**HAFH-RDPG**	**DRQN**
Actor size	256,300,100	128,128,64	\
Critic(Q) size	400,300,100	128,128,64	256,300,100
Actor learning rate	5e-6	5e-6	\
Critic(Q) learning rate	5e-6	5e-6	1e-4
History window	\	4	4
Batch size	128		64
Training episodes	15000		
Reward scale	2e-3		

B. Experimental Results

1) Performance for Test Set: Table III summarizes the individual, average, maximum performance, as well as the

TABLE III
INDIVIDUAL, AVERAGE, MAXIMUM PERFORMANCE, AS WELL AS THE STANDARD ERROR OF THE PROPOSED AND BENCHMARK ALGORITHMS ACROSS FIVE RUNS. IN EACH RUN, WE EXECUTED 100 EPISODES OF 24 TIME STEPS. THE INDIVIDUAL PERFORMANCE IS THE AVERAGE CUMULATIVE REWARD OVER 100 TEST EPISODES

Environments	Algorithms	Dataset	Performance							
			Run 1	Run 2	Run 3	Run 4	Run 5	Max	Average	Std Error
MDP	Myopic	Type A	-12.8175	-13.5276	-14.9506	-16.8130	-13.2446	-12.8175	-14.2707	1.6311
	DDP		-11.2421	-12.1325	-12.5870	-15.1273	-11.7430	-11.2421	-12.5664	1.5148
	HAFH-DDPG		-11.1819	-11.8509	-12.4679	-14.9620	-11.5876	-11.1819	-12.4101	1.5011
POMDP	DRQN		-12.0895	-12.9763	-14.1435	-15.8547	-12.4533	-12.0895	-13.5035	1.5262
	HAFH-RDPG		-11.2208	-12.2243	-12.7963	-15.1054	-11.6984	-11.2208	-12.6090	1.5142
		Type B, $N = 7$	-11.4238	-12.4536	-13.0564	-15.3693	-11.9409	-11.4238	-12.8488	1.5336
		Type B, $N = 14$	-11.4021	-12.3575	-13.3536	-15.2703	-12.0685	-11.4021	-12.8904	1.5044
		Type B, $N = 21$	-11.6264	-12.5459	-13.5580	-15.5341	-12.3246	-11.6264	-13.1178	1.5176

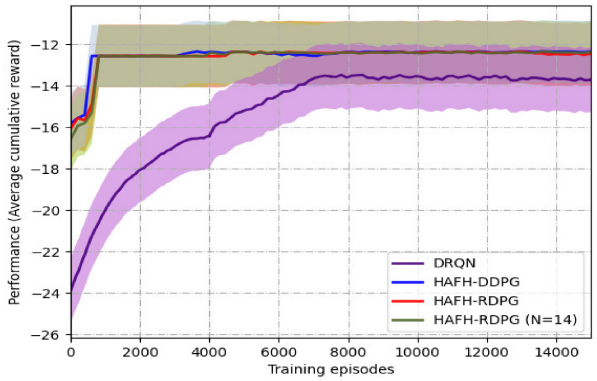

Fig. 5. Average performance curves of three DRL algorithms across five runs in the training process. The ordinate corresponds to the performance, i.e., the average cumulative reward over ten test episodes. The shaded areas indicate the standard errors of the three algorithms.

standard error on five runs for all the algorithms. In each run, we executed 100 complete episodes of 24 time steps for each algorithm based on the test set, and obtain the individual performance by averaging the cumulative rewards over the 100 episodes, where the cumulative reward of one episode is given by (24). Therefore, the performance corresponds to the average cumulative reward over 100 test episodes, which reflects the optimization objective defined in (25). The myopic, DDP, and HAFH-DDPG algorithms are implemented in the MDP environment, while the HAFH-RDPG and DRQN algorithms are implemented in the POMDP environment. Moreover, dataset Type A is used for all the benchmark algorithms, while both dataset Types A and B are used for HAFH-RDPG algorithms.

2) Convergence Properties: Fig. 5 shows the average performance curves of the DRL algorithms across five runs, which are obtained by evaluating the policies periodically during training without noise. In specific, during the training process, we ran ten test episodes based on the current model parameters every 100 training episodes in order to accurately represent the training results. The abscissa is the number of training episodes and the ordinate is the performance of DRL algorithms, which denotes the average cumulative reward over ten test episodes. The shaded areas indicate the standard errors of the algorithms across the five runs. Moreover, for HAFH-DDPG and HAFH-RDPG algorithms, we only show the performance curves of the first time step $t = 1$, as the performance curves for all T time steps are similar.

3) Energy Dispatch Results: Fig. 6 illustrates the energy scheduling decisions and corresponding costs of different algorithms over one day based on the test set of run 1. Due to space limitations, only the results for myopic, DDP, HAFH-DDPG, and HAFH-RDPG (Type A) are shown. In Fig. 6(a), (c), (e), and (g), three curves are displayed corresponding to the equivalent load trajectory, generation power trajectory, and battery trajectory, respectively, in each figure. The generation power value at time step t is derived by $\sum_{d=1}^{D} P_t^{\mathrm{DG}_d} - P_t^E$, i.e., the total power of DGs after charging/discharging of the battery. Moreover, the charge or discharge power of the battery and the power generated by each of the three DGs are also shown through the bar chart. In Fig. 6(b), (d), (f), and (h), the sum cost trajectory is displayed in each figure, where the sum cost at time step t is the negative reward $-r(S_t, A_t)$. Moreover, we use the bar chart to visualize the individual values of different costs in detail. Finally, Fig. (7) displays the sum cost trajectories of the different algorithms in the same figure to clearly show the sum cost comparisons.

C. Analysis and Discussion

1) Capability in Handling Uncertainty Due to Interhour Power Fluctuation: We focus on the performance of myopic, DDP, HAFH-DDPG, and HAFH-RDPG algorithms when data set Type A is used. Note that the benchmark algorithms are all implemented in the MDP environment, where the PV and load data of the current time step are assumed to be available, while the proposed HAFH-RDPG algorithm is implemented in the POMDP environment, where the PV and load data of the past four time steps are used to determine the actions. It can be observed in Table III that HAFH-DDPG has the largest maximum performance, followed by HAFH-RDPG, DDP, and myopic algorithms. Moreover, HAFH-DDPG also achieves the best average performance across five runs among all the algorithms, while DDP and HAFH-RDPG (for Type A) have the second and third best average performance. The myopic algorithm has the worst average performance on five runs. Finally, HAFH-DDPG achieves the lowest standard error, followed by HAFH-RDPG, DDP, and myopic algorithms.

For each run, the individual performance of HAFH-DDPG is consistently better than those of the other algorithms. As HAFH-DDPG is the version of HAFH-RDPG in the MDP environment, its high performance indicates that our proposed algorithm works well when there is no uncertainty due to interhour power

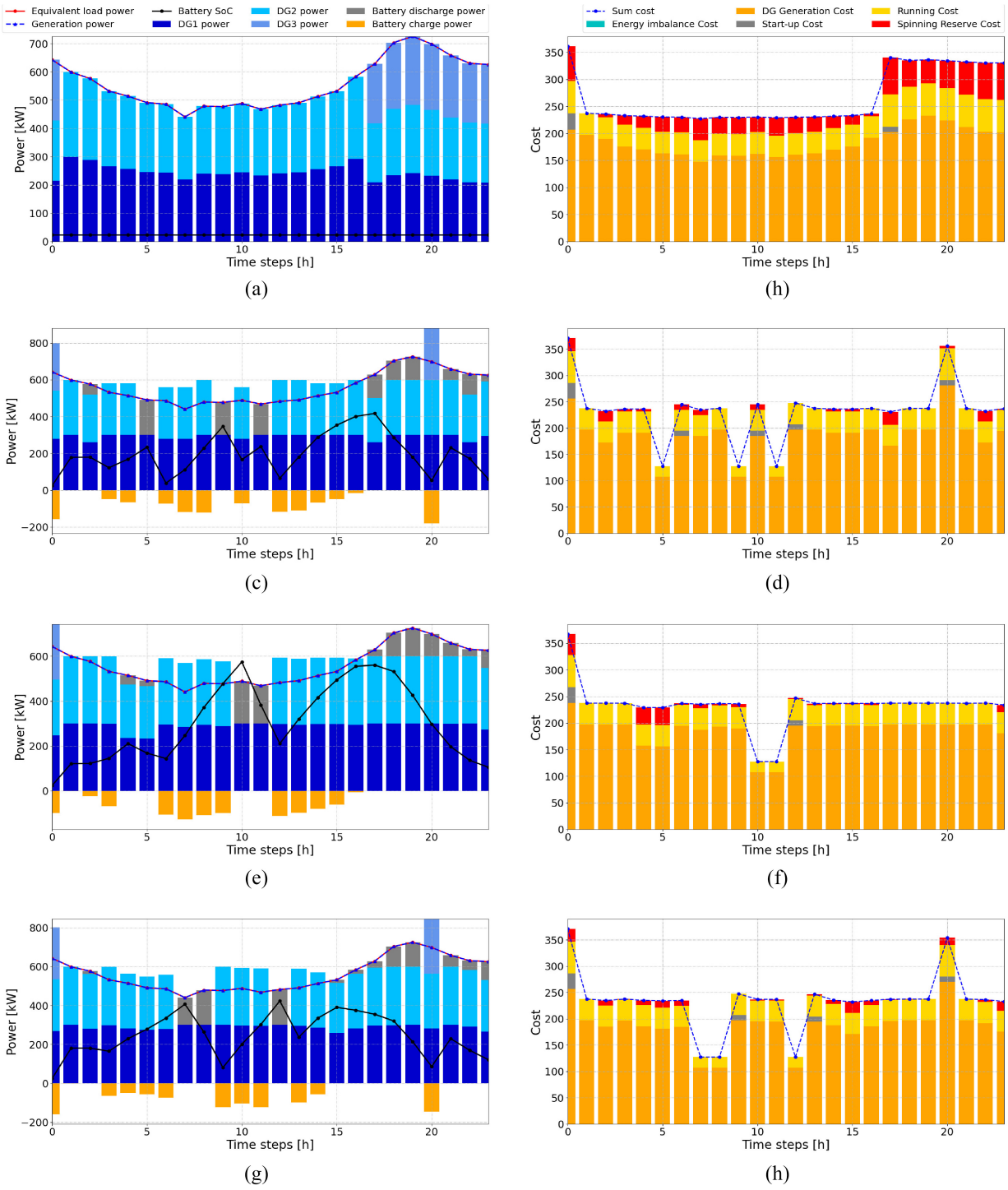

Fig. 6. Equivalent load, generation power, battery, and sum cost trajectories of various algorithms for a specific test episode on run 1. The charge or discharge power of the battery, the power generated by each of the three DGs, and the individual costs are shown through the bar chart. (a) Energy scheduling of myopic. (b) Costs of myopic. (c) Energy scheduling of DDP. (d) Costs of DDP. (e) Energy scheduling of HAFH-DDPG. (f) Costs of HAFH-DDPG. (g) Energy scheduling of HAFH-RDPG. (h) Costs of HAFH-RDPG.

fluctuation, and thus no performance degradation due to partial observation. Moreover, although HAFH-RDPG is applied in the POMDP environment, it outperforms DDP in the MDP environment on run 1, run 4, and run 5, which demonstrates that our proposed algorithm can exploit history data to deal with the uncertainty due to interhour power fluctuation and make efficient decisions. Although the performance of DDP is comparable to that of HAFH-RDPG, it is trained in the MDP environment ignoring the partial observable problem due to the unavailable PV generation and load demand data of the next hour. When prediction errors for the next-hour data occur in realistic scenarios, it can be expected that the performance of DDP will degrade and become worse than that of HAFH-RDPG. More importantly, the computational

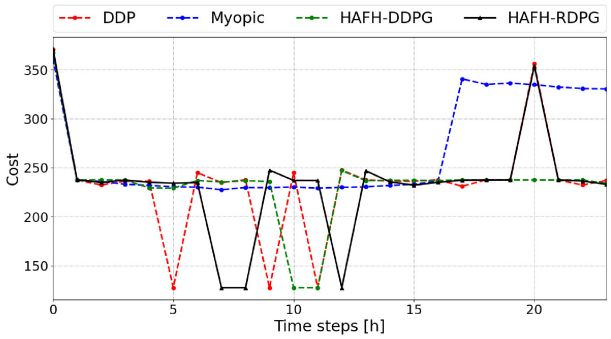

Fig. 7. Sum cost trajectories for a specific test episode to compare the performance of different algorithms.

complexity of DDP is much higher than that of HAFH-RDPG. Although theoretically, DDP can obtain the optimal solution when the discretization granularity of state and action space is infinitely small, it is not a practical option as the computation complexity $O(|\mathcal{S}|^2 * |\mathcal{A}|)$ depends on the state and action space dimensions. In order to achieve the presented performance in our experiment, we train DDP for 100 iterations, and the optimal value function is updated by over 2.7 billion times in each iteration. Overall, the running time of DDP is approximately 11 times that of HAFH-RDPG. Finally, the standard errors across the five runs are mainly due to the differences in PV and load data on different days instead of the instability of DRL algorithms. This is corroborated by the fact that the standard error of the myopic algorithm is higher than those of the DRL algorithms. It can be observed that our proposed algorithm performs stably in both MDP and POMDP environments, as the HAFH-DDPG and HAFH-RDPG algorithms achieve the first and second lowest standard errors.

Focusing on the convergence properties in Fig. 5, it can be observed that HAFH-DDPG has the fastest convergence speed and converges at approximately 600 episodes. Despite being in the POMDP environment, HAFH-RDPG also converges fast at approximately 800 episodes. Moreover, the shaded areas of HAFH-DDPG and HAFH-RDPG are similar and have a large overlap, which indicates that the stability of the proposed algorithms is comparable in both MDP and POMDP environments.

For the ED results in Fig. 6, we find that the curves of load and generation power match exactly for all the algorithms, which means the basic energy balance requirement can be met. A closer examination shows that the generation power and battery trajectories of DDP [Fig. 6(c)], HAFH-DDPG [Fig. 6(e)], and HAFH-RDPG [Fig. 6(g)] are very similar. These algorithms adopt similar strategies which try to keep the DGs that are ON generating very high power, which can largely reduce spinning reserve cost. The surplus power is charged to the battery so that one or more DGs can be turned off to reduce running cost when the battery has enough power to meet the load demand. As a result, two DGs are ON at most time steps, and sometimes only one ON DG is needed during the morning and noon hours. In addition, it can be observed visually that only two DGs are used in HAFH-DDPG except for the first time step as shown in Fig. 6(e), which leads to the lowest costs as shown in Fig. 7. This conclusion shows that the MDP version of the proposed algorithm achieves the best results, which is also verified by Table III. Most importantly, although the PV and load power of the current time step cannot be obtained by HAFH-RDPG, Fig. 6(g) shows that this algorithm still outperforms DDP in energy scheduling on run 1. Compared with DDP, HAFH-RDPG only incurs the start-up cost three times at 9:00, 13:00, and 20:00, which reduces the loss of frequently changing switching states. Finally, compared with the above policies, the myopic algorithm is unable to take advantage of the battery to charge in advance for the evening peak and for energy shifting. Therefore, it has to turn on three DGs after 17:00, which leads to great spinning reserve cost and DG generation cost as shown in Fig. 7.

2) Capability in Handling Uncertainty Due to Interday Power Fluctuation: We compare the performance of HAFH-RDPG when both dataset Types A and B are used, respectively. For Type B, we set the number of past days for training, i.e., N, to be 7, 14, and 21, respectively, to prevent the agent from overfitting to the statistics of one particular day. Table III shows that the best average performance for Type B ($N = 7$) is only 1.90% lower than that for Type A and the best maximum performance for Type B ($N = 14$) is only 1.62% lower than that for Type A, demonstrating that the proposed algorithm has the capability in handling uncertainty due to interday power fluctuation. Comparing the performance for Type B when different numbers of past days N are used for training, it can be observed that the individual performance for $N = 21$ on each run is always the worst. The individual performance for $N = 7$ is better than those for $N = 14$ on run 1, run 2, and run 4, while the opposite is true on run 3 and run 5. Overall, the average performance for $N = 21$ is 2.09% and 1.76% lower than that for $N = 7$ and $N = 14$, respectively. This is because when N is smaller, the training data are more recent, and there is a higher probability that the statistics of PV and load data are more similar to those of the test data. Meanwhile, in the rare case, when the test data is an outliner, whose statistical properties are much different from those of data in the recent past, increasing N can make the training data more general and thus improve the performance of the trained policy on the test data. Therefore, the setting for N should strike a good tradeoff, and both $N = 7$ and $N = 14$ are appropriate in this case.

Fig. 5 shows that the convergence speed and stability of HAFH-RDPG are very similar when trained using dataset Type A or B, which is as expected. In addition, notice that DDP requires predictive models to handle the uncertainty due to interday power fluctuation, while HAFH-RDPG directly learns a policy from the history data. The resultant robustness to the prediction model error is also an advantage of the proposed algorithm over DDP.

3) Capability in Dealing With Hybrid Action Space: We compare the performance of DRQN and HAFH-RDPG, where both algorithms work in the POMDP environment. However, DRQN uses discretization to deal with the hybrid action space, while HAFH-RDPG uses the proposed method. Table III shows that HAFH-RDPG always has better individual performance on each run than DRQN. The maximum performance of HAFH-RDPG is 7.74% larger than that of DRQN, and the average performance of HAFH-RDPG is

7.09% larger than that of DRQN, demonstrating the superior capability of HAFH-RDPG in dealing the hybrid action space. The standard error of DRQN is 0.79% larger than that of HAFH-RDPG, which indicates that the proposed algorithm is more stable than DRQN.

Focusing on the convergence properties, Fig. 5 shows that HAFH-RDPG converges much faster and with smaller fluctuation than DRQN. The former converges at approximately 800 training episodes, while the latter converges at approximately 7500 episodes. This is because HAFH-RDPG iteratively trains to solve the one-period POMDP problem for each time step using RDPG with a fixed target, which makes our proposed algorithm much easier to converge and more stable.

VI. Conclusion

In this article, we have studied the energy scheduling issue with low operation cost in IoT-driven isolated smart MG systems by fully exploiting the renewable energy. The DRL approach has been adopted to handle the uncertainty of PV generation and load demand. A finite-horizon POMDP model has been developed considering the spinning reserve. The HAFH-RDPG algorithm has been proposed, which overcomes the challenge of learning optimal policy with discrete-continuous hybrid action space. Finally, experiments have been performed to demonstrate that the proposed algorithm can efficiently tackle with the uncertainty due to interhour and interday power fluctuation and can achieve better performance than the other benchmark algorithms. In the future, this work will be further extended by optimizing the ED of network PV panels jointly with the scheduling of DGs, which can help to minimize reverse power flow in the isolated MGs. Due to the distributed deployment of residential PV, multiagent DRL and federated DRL can be leveraged.

Appendix A
Proof for Theorem 1

The optimal policy for the objective in (26) achieves the optimal Q value, i.e.,

$$Q\left(H_t, k^*, a_{k^*}^{\mathrm{ED}*}\right) = \max_{k\in\{1,\ldots,2^D\}, a_k^{\mathrm{ED}}\in\mathcal{A}_k^{\mathrm{ED}}} Q\left(H_t, k, a_k^{\mathrm{ED}}\right). \quad (42)$$

Combining (27) and (30), the policy derived by the two-step framework also achieves the optimal Q value, i.e.,

$$\begin{aligned} Q\left(H_t, \hat{k}^*, a_{\hat{k}^*}^{\mathrm{ED}*}\right) &= \max_{k\in\{1,\ldots,2^D\}} \max_{a_k^{\mathrm{ED}}\in\mathcal{A}_k^{\mathrm{ED}}} Q\left(H_t, k, a_k^{\mathrm{ED}}\right), \\ &= \max_{k\in\{1,\ldots,2^D\}, a_k^{\mathrm{ED}}\in\mathcal{A}_k^{\mathrm{ED}}} Q\left(H_t, k, a_k^{\mathrm{ED}}\right) \\ &= Q\left(H_t, k^*, a_{k^*}^{\mathrm{ED}*}\right). \end{aligned} \quad (43)$$

■

Appendix B
Proof for Theorem 2

Our proof is divided into two steps. The first step is to prove that both k^* and k_{m^*} can achieve the maximum immediate reward at any time step t. According to the definitions in (21), (15), (17) and (18), it is obvious that DGs have translatable symmetry for the supply–demand balance cost, power generation cost, running cost, and spinning reserve cost. Therefore, the sum of these costs is the same for any $k \in \mathcal{A}_m^{\mathrm{SW}}$, i.e.,

$$\begin{aligned} &c_{t,k_1}^{\mathrm{US}} + \sum_{d=1}^{D}\left(c_{t,k_1}^{\mathrm{DG}_d} + c_{t,k_1}^{\mathrm{R\_DG}_d} + c_{t,k_1}^{\mathrm{SR\_DG}_d}\right) \\ &\quad = c_{t,k_2}^{\mathrm{US}} + \sum_{d=1}^{D}\left(c_{t,k_2}^{\mathrm{DG}_d} + c_{t,k_2}^{\mathrm{R\_DG}_d} + c_{t,k_2}^{\mathrm{SR\_DG}_d}\right), \\ &\quad \forall k_1, k_2 \in \mathcal{A}_m^{\mathrm{SW}}. \end{aligned} \quad (44)$$

In addition, the DG selection strategy can ensure that the maximum number of ON DGs at time step t is selected to be ON at time step $t+1$. In other words, the strategy minimizes the number of DGs whose $B_t^{\mathrm{DG}_d} = 0$ and $U_t^{\mathrm{DG}_d} = 1$. According to (1) and (16), the DG selection strategy in (39) leads to the switching action k_m in $\mathcal{A}_m^{\mathrm{SW}}$ that minimizes the start-up cost, i.e.,

$$k_m = \arg\min_{k\in\mathcal{A}_m^{\mathrm{SW}}}\left(\sum_{d=1}^{D} c_{t,k}^{\mathrm{S\_DG}_d}\right). \quad (45)$$

Therefore, according to the reward definition in (20), k_m derived from the strategy maximizes the reward according to (44) and (45), i.e.,

$$k_m = \arg\max_{k\in\mathcal{A}_m^{\mathrm{SW}}}\left(r\left(S_t, k, a_k^{\mathrm{ED}*}\right)\right). \quad (46)$$

Thus, we can derive

$$\begin{aligned} \max_{k\in\mathcal{A}^{\mathrm{SW}}}\left(r\left(S_t, k, a_k^{\mathrm{ED}*}\right)\right) &\overset{(a)}{=} \max_{m\in\hat{\mathcal{A}}^{\mathrm{SW}}}\max_{k\in\mathcal{A}_m^{\mathrm{SW}}}\left(r\left(S_t, k, a_k^{\mathrm{ED}*}\right)\right) \\ &\overset{(b)}{=} \max_{m\in\hat{\mathcal{A}}^{\mathrm{SW}}}\left(r\left(S_t, k_m, a_{k_m}^{\mathrm{ED}*}\right)\right) \end{aligned} \quad (47)$$

where (a) is due to (38); and (b) follows from (46).

In the second step, we prove (41) using induction by the Bellman equation (32). For the last time step T, we have

$$Q_T\left(H_T, k, a_k^{\mathrm{ED}*}\right) = \mathop{\mathbb{E}}_{S_T|H_T}\left[r\left(S_T, k, a_k^{\mathrm{ED}*}\right)\right] \quad \forall k \in \mathcal{A}^{\mathrm{SW}}. \quad (48)$$

Thus, combining (47) and (48), we have

$$\max_{m\in\hat{\mathcal{A}}^{\mathrm{SW}}} Q_T\left(H_T, k_m, a_{k_m}^{\mathrm{ED}*}\right) = \max_{k\in\mathcal{A}^{\mathrm{SW}}} Q_T\left(H_T, k, a_k^{\mathrm{ED}*}\right). \quad (49)$$

Then, we assume that

$$\begin{aligned} &\max_{m'\in\hat{\mathcal{A}}^{\mathrm{SW}}} Q_{t+1}\left(H_{t+1}, k_{m'}, a_{k_{m'}}^{\mathrm{ED}*}\right) \\ &\quad = \max_{k'\in\mathcal{A}^{\mathrm{SW}}} Q_{t+1}\left(H_{t+1}, k', a_{k'}^{\mathrm{ED}*}\right) \quad \forall H_{t+1} \in \mathcal{H}. \end{aligned} \quad (50)$$

Next, we can prove

$$\begin{aligned} &\sum_{H_{t+1}\in\mathcal{H}} \Pr\left(H_{t+1}|H_t, k_1, a_{k_1}^{\mathrm{ED}*}\right)\cdot \max_{k'\in\mathcal{A}^{\mathrm{SW}}} Q_t\left(H_{t+1}, k', a_{k'}^{\mathrm{ED}*}\right) \\ &\quad = \sum_{H_{t+1}\in\mathcal{H}} \Pr\left(H_{t+1}|H_t, k_2, a_{k_2}^{\mathrm{ED}*}\right)\cdot \max_{k'\in\mathcal{A}^{\mathrm{SW}}} Q_t\left(H_{t+1}, k', a_{k'}^{\mathrm{ED}*}\right), \\ &\quad \forall k_1, k_2 \in \mathcal{A}_m^{\mathrm{SW}}. \end{aligned} \quad (51)$$

This is because the history transition probability $\Pr(H_{t+1}|H_t, A_t)$ consists of four parts according to (23), where the transition probabilities of load demands and PV output powers, i.e., $\Pr(P^L_t|P^L_{t-\tau}, \ldots, P^L_{t-1})$ and $\Pr(P^{\rm PV}_t|P^{\rm PV}_{t-\tau}, \ldots, P^{\rm PV}_{t-1})$, do not depend on the switching action k. Moreover, all the switching actions $\forall k \in \mathcal{A}^{\rm SW}_m$ lead to the same SoC state E_{t+1} according to (4) and (8)–(10). Finally, the next switching state B_{t+1} is different for $\forall k \in \mathcal{A}^{\rm SW}_m$, but the number of ON DGs at the next time step $t+1$ is the same, i.e., m. Therefore, (51) is proved.

Finally, we can prove (41) at each time step t based on mathematical induction by considering the following Bellman equations in (32), i.e.,

$$\begin{aligned}
&\max_{m\in\hat{\mathcal{A}}^{\rm SW}} Q_t\left(H_t, k_m, a^{\rm ED*}_{k_m}\right)\\
&\overset{(a)}{=} \max_{m\in\hat{\mathcal{A}}^{\rm SW}} \left\{ \mathop{\mathbb{E}}_{S_t|H_t}\left[r\left(S_t, k_m, a^{\rm ED*}_{k_m}\right)\right] \right.\\
&\quad \left. + \mathop{\mathbb{E}}_{H_{t+1}|H_t,k_m,a^{\rm ED*}_{k_m}}\left[\gamma \cdot \max_{m'\in\hat{\mathcal{A}}^{\rm SW}} Q_t\left(H_{t+1}, k_{m'}, a^{\rm ED*}_{k_{m'}}\right)\right]\right\}\\
&\overset{(b)}{=} \max_{m\in\hat{\mathcal{A}}^{\rm SW}} \left\{ \mathop{\mathbb{E}}_{S_t|H_t}\left[r\left(S_t, k_m, a^{\rm ED*}_{k_m}\right)\right] \right.\\
&\quad \left. + \mathop{\mathbb{E}}_{H_{t+1}|H_t,k_m,a^{\rm ED*}_{k_m}}\left[\gamma \cdot \max_{k'\in\mathcal{A}^{\rm SW}} Q_t\left(H_{t+1}, k', a^{\rm ED*}_{k'}\right)\right]\right\}\\
&\overset{(c)}{=} \max_{m\in\hat{\mathcal{A}}^{\rm SW}} \max_{k\in\mathcal{A}^{\rm SW}_m} \left\{ \mathop{\mathbb{E}}_{S_t|H_t}\left[r\left(S_t, k, a^{\rm ED*}_{k}\right)\right] \right.\\
&\quad \left. + \mathop{\mathbb{E}}_{H_{t+1}|H_t,k,a^{\rm ED*}_{k}}\left[\gamma \cdot \max_{k'\in\mathcal{A}^{\rm SW}} Q_t\left(H_{t+1}, k', a^{\rm ED*}_{k'}\right)\right]\right\}\\
&\overset{(d)}{=} \max_{k\in\mathcal{A}^{\rm SW}} \left\{ \mathop{\mathbb{E}}_{S_t|H_t}\left[r\left(S_t, k, a^{\rm ED*}_{k}\right)\right] \right.\\
&\quad \left. + \mathop{\mathbb{E}}_{H_{t+1}|H_t,k,a^{\rm ED*}_{k}}\left[\gamma \cdot \max_{k'\in\mathcal{A}^{\rm SW}} Q_t\left(H_{t+1}, k', a^{\rm ED*}_{k'}\right)\right]\right\}\\
&\overset{(e)}{=} \max_{k\in\mathcal{A}^{\rm SW}} Q_t\left(H_t, k, a^{\rm ED*}_{k}\right) \qquad (52)
\end{aligned}$$

where (a) and (e) are due to the definition of the Bellman equation; (b) follows by (50); (c) is due to (51) and the definition of expectation; and (d) follows by (38). Therefore, (41) in Theorem 1 is proved according to (52). ■

REFERENCES

[1] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, "Deep reinforcement learning for autonomous Internet of Things: Model, applications and challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1722–1760, 3rd Quart., 2020.

[2] M. S. Munir, S. F. Abedin, N. H. Tran, and C. S. Hong, "When edge computing meets microgrid: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7360–7374, Oct. 2019.

[3] H. Zhou, A. Aral, I. Brandić, and M. Erol-Kantarci, "Multiagent Bayesian deep reinforcement learning for microgrid energy management under communication failures," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 11685–11698, Jul. 2022.

[4] R. M. González, F. D. Wattjes, M. Gibescu, W. Vermeiden, J. G. Slootweg, and W. L. Kling, "Applied Internet of Things architecture to unlock the value of smart microgrids," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 5326–5336, Dec. 2018.

[5] O. M. Butt, M. Zulqarnain, and T. M. Butt, "Recent advancement in smart grid technology: Future prospects in the electrical power network," *Ain Shams Eng. J.*, vol. 12, no. 1, pp. 687–695, 2021.

[6] G. Bedi, G. K. Venayagamoorthy, R. Singh, R. R. Brooks, and K.-C. Wang, "Review of Internet of Things (IoT) in electric power and energy systems," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 847–870, Apr. 2018.

[7] M. Azaroual, M. Ouassaid, and M. Maaroufi, "Optimal control for energy dispatch of a smart grid tied PV-wind-battery hybrid power system," in *Proc. 3rd Int. Conf. Intell. Comput. Data Sci. (ICDS)*, 2019, pp. 1–7.

[8] H. Çimen, N. Çetinkaya, J. C. Vasquez, and J. M. Guerrero, "A microgrid energy management system based on non-intrusive load monitoring via multitask learning," *IEEE Trans. Smart Grid*, vol. 12, no. 2, pp. 977–987, Mar. 2021.

[9] G. van Leeuwen, T. AlSkaif, M. Gibescu, and W. van Sark, "An integrated blockchain-based energy management platform with bilateral trading for microgrid communities," *Appl. Energy*, vol. 263, Apr. 2020, Art. no. 114613. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306261920301252

[10] D. Arcos-Aviles et al., "A review of fuzzy-based residential grid-connected microgrid energy management strategies for grid power profile smoothing," in *Energy Sustainability in Built and Urban Environments*. Singapore: Springer, 2019, pp. 165–199.

[11] X. Chen, W. Dong, and Q. Yang, "Robust optimal capacity planning of grid-connected microgrid considering energy management under multi-dimensional uncertainties," *Appl. Energy*, vol. 323, Oct. 2022, Art. no. 119642.

[12] J. Sachs and O. Sawodny, "A two-stage model predictive control strategy for economic diesel-PV-battery island microgrid operation in rural areas," *IEEE Trans. Sustain. Energy*, vol. 7, no. 3, pp. 903–913, Jul. 2016.

[13] J. D. Lara, D. E. Olivares, and C. A. Cañizares, "Robust energy management of isolated microgrids," *IEEE Syst. J.*, vol. 13, no. 1, pp. 680–691, Mar. 2019.

[14] A. C. Luna, N. L. Diaz, M. Graells, J. C. Vasquez, and J. M. Guerrero, "Mixed-integer-linear-programming-based energy management system for hybrid PV-wind-battery microgrids: Modeling, design, and experimental verification," *IEEE Trans. Power Electron.*, vol. 32, no. 4, pp. 2769–2783, Apr. 2017.

[15] L. Moretti, S. Polimeni, L. Meraldi, P. Raboni, S. Leva, and G. Manzolini, "Assessing the impact of a two-layer predictive dispatch algorithm on design and operation of off-grid hybrid microgrids," *Renew. Energy*, vol. 143, pp. 1439–1453, Dec. 2019.

[16] F. Conte, F. D'Agostino, P. Pongiglione, M. Saviozzi, and F. Silvestro, "Mixed-integer algorithm for optimal dispatch of integrated PV-storage systems," *IEEE Trans. Ind. Appl.*, vol. 55, no. 1, pp. 238–247, Jan./Feb. 2019.

[17] B. V. Solanki, K. Bhattacharya, and C. A. Cañizares, "A sustainable energy management system for isolated microgrids," *IEEE Trans. Sustain. Energy*, vol. 8, no. 4, pp. 1507–1517, Oct. 2017.

[18] B. V. Solanki, C. A. Cañizares, and K. Bhattacharya, "Practical energy management systems for isolated microgrids," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 4762–4775, Sep. 2019.

[19] M. F. Zia, E. Elbouchikhi, M. Benbouzid, and J. M. Guerrero, "Energy management system for an islanded microgrid with convex relaxation," *IEEE Trans. Ind. Appl.*, vol. 55, no. 6, pp. 7175–7185, Nov./Dec. 2019.

[20] J. S. Giraldo, J. A. Castrillon, J. C. López, M. J. Rider, and C. A. Castro, "Microgrids energy management using robust convex programming," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 4520–4530, Jul. 2019.

[21] D. E. Olivares, J. D. Lara, C. A. Cañizares, and M. Kazerani, "Stochastic-predictive energy management system for isolated microgrids," *IEEE Trans. Smart Grid*, vol. 6, no. 6, pp. 2681–2693, Nov. 2015.

[22] W. Violante, C. A. Cañizares, M. A. Trovato, and G. Forte, "An energy management system for isolated microgrids with thermal energy resources," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 2880–2891, Jul. 2020.

[23] Y. Li, Z. Yang, G. Li, D. Zhao, and W. Tian, "Optimal scheduling of an isolated microgrid with battery storage considering load and renewable generation uncertainties," *IEEE Trans. Ind. Electron.*, vol. 66, no. 2, pp. 1565–1575, Feb. 2019.

[24] Y. Li, R. Wang, and Z. Yang, "Optimal scheduling of isolated microgrids using automated reinforcement learning-based multi-period forecasting," *IEEE Trans. Sustain. Energy*, vol. 13, no. 1, pp. 159–169, Jan. 2022.

[25] B. O'Donoghue, I. Osband, R. Munos, and V. Mnih, "The uncertainty Bellman equation and exploration," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3836–3845.

[26] C. Hu, Z. Cai, and Y. Zhang, "A multi-agent deep reinforcement learning approach for temporally coordinated demand response in microgrids," *CSEE J. Power Energy Syst.*, early access, Aug. 18, 2022, doi: 10.17775/CSEEJPES.2021.05090.
[27] E. Mocanu et al., "On-line building energy optimization using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3698–3708, Jul. 2019.
[28] F. Ruelens, B. J. Claessens, S. Vandael, S. Iacovella, P. Vingerhoets, and R. Belmans, "Demand response of a heterogeneous cluster of electric water heaters using batch reinforcement learning," in *Proc. Power Syst. Comput. Conf.*, 2014, pp. 1–7.
[29] X. Xiao, C. Dai, Y. Li, C. Zhou, and L. Xiao, "Energy trading game for microgrids using reinforcement learning," in *Proc. Int. Conf. Game Theory Netw.*, 2017, pp. 131–140.
[30] B.-G. Kim, Y. Zhang, M. Van Der Schaar, and J.-W. Lee, "Dynamic pricing for smart grid with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2014, pp. 640–645.
[31] L. Xiao, X. Xiao, C. Dai, M. Pengy, L. Wang, and H. V. Poor, "Reinforcement learning-based energy trading for microgrids," 2018, *arXiv:1801.06285*.
[32] Y. Ji, J. Wang, J. Xu, X. Fang, and H. Zhang, "Real-time energy management of a microgrid using deep reinforcement learning," *Energies*, vol. 12, no. 12, p. 2291, 2019. [Online]. Available: https://www.mdpi.com/1996-1073/12/12/2291
[33] H. Shuai and H. He, "Online scheduling of a residential microgrid via Monte-Carlo tree search and a learned model," *IEEE Trans. Smart Grid*, vol. 12, no. 2, pp. 1073–1087, Mar. 2021.
[34] Y. Du and F. Li, "Intelligent multi-microgrid energy management based on deep neural network and model-free reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 2, pp. 1066–1076, Mar. 2020.
[35] V. François-Lavet, D. Taralla, D. Ernst, and R. Fonteneau, "Deep reinforcement learning solutions for energy microgrids management," in *Proc. Eur. Workshop Reinforcement Learn. (EWRL)*, 2016, pp. 1–7.
[36] L. Lei, Y. Tan, G. Dahlenburg, W. Xiang, and K. Zheng, "Dynamic energy dispatch based on deep reinforcement learning in IoT-driven smart isolated microgrids," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 7938–7953, May 2021.
[37] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[38] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[39] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," 2015, *arXiv:1512.04455*.
[40] J. Xiong et al., "Parametrized deep Q-networks learning: Reinforcement learning with discrete-continuous hybrid action space," 2018, *arXiv:1810.06394*.
[41] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.
[42] M. J. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," in *Proc. AAAI Fall Symp. Ser.*, 2015, pp. 29–37.

Jiaju Qi received the B.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2017. He is currently pursuing the Ph.D. degree in engineering with the University of Guelph, Guelph, ON, Canada.

His research interests include machine learning, Internet of Things, reinforcement learning, smart grid, and electric vehicle.

Lei Lei (Senior Member, IEEE) received the B.S. and Ph.D. degrees in telecommunications engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2001 and 2006, respectively.

She is currently an Associate Professor with the College of Engineering and Physical Sciences, University of Guelph, Guelph, ON, Canada. Her research interests mainly lie in machine learning/deep reinforcement learning, Internet of Things/Internet of Vehicles, mobile-edge computing, and smart grid.

Kan Zheng (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 1996, 2000, and 2005, respectively.

He is currently a Full Professor with Ningbo University, Ningbo, Zhejiang, China. He has rich experiences in research and standardization of new emerging technologies. He has authored over 200 journal articles and conference papers in the field of wireless communications, vehicular networks, IoT, and security.

Prof. Zheng holds editorial board positions with several journals. He has also served in the organizing/TPC committees for more than ten conferences.

Simon X. Yang (Senior Member, IEEE) received the B.Sc. degree in engineering physics from Beijing University, Beijing, China, in 1987, the first M.Sc. degree in biophysics from the Chinese Academy of Sciences, Beijing, in 1990, the second M.Sc. degree in electrical engineering from the University of Houston, Houston, TX, USA, in 1996, and the Ph.D. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 1999.

He is currently a Professor and the Head of the Advanced Robotics and Intelligent Systems Laboratory, University of Guelph, Guelph, ON, Canada. His research interests include robotics, intelligent systems, control systems, sensors and multisensor fusion, wireless sensor networks, intelligent communication, intelligent transportation, machine learning, fuzzy systems, and computational neuroscience.

Prof. Yang has been very active in professional activities. He serves as the Editor-in-Chief for *Intelligence & Robotics* and *International Journal of Robotics and Automation* and an Associate Editor for IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS OF ARTIFICIAL INTELLIGENCE, and several other journals. He has involved in the organization of many international conferences.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular networks.

Dr. Shen received the Canadian Award for Telecommunications Research from the Canadian Society of Information Theory in 2021, the R.A. Fessenden Award in 2019 from IEEE, Canada, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society (ComSoc), and the Technical Recognition Award from Wireless Communications Technical Committee in 2019 and AHSN Technical Committee in 2013. He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom'16, IEEE Infocom'14, IEEE VTC'10 Fall, and IEEE Globecom'07, and the Chair for the IEEE ComSoc Technical Committee on Wireless Communications. He is the President of the IEEE ComSoc. He was the Vice President for Technical & Educational Activities, the Vice President for Publications, a Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, and a member of IEEE Fellow Selection Committee of the ComSoc. He served as the Editor-in-Chief for the IEEE INTERNET OF THINGS JOURNAL, IEEE NETWORK, and *IET Communications*. He is a Registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.