Reward Shaping-based Actor-Critic Deep Reinforcement Learning for Residential Energy Management

Renzhi Lu, *Member, IEEE,* Zhenyu Jiang, Huaming Wu, *Senior Member, IEEE,* Yuemin Ding*, *Senior Member, IEEE,* Dong Wang, *Senior Member, IEEE,* and Hai-Tao Zhang, *Senior Member, IEEE*

Abstract—**Residential energy consumption continues to climb steadily, requiring intelligent energy management strategies to reduce power system pressures and residential electricity bills. However, it is challenging to design such strategies due to the random nature of electricity pricing, appliance demand and user behavior. This paper presents a novel reward shaping (RS)-based actor-critic deep reinforcement learning (ACDRL) algorithm to manage the residential energy consumption profile with limited information about the uncertain factors. Specifically, the interaction between the energy management center and various residential loads is modeled as a Markov decision process that provides a fundamental mathematical framework to represent the decision making in situations where outcomes are partially random and partially influenced by the decision maker control signals, in which the key elements containing the agent, environment, state, action, and reward are carefully designed, and the electricity price is considered as a stochastic variable. An RS-ACDRL algorithm is then developed, incorporating both the actor and critic network and an RS mechanism, to learn the optimal energy consumption schedules. Several case studies involving real-world data are conducted to evaluate the performance of the proposed algorithm. Numerical results demonstrate that the proposed algorithm outperforms state-of-the-art RL methods in terms of learning speed, solution optimality and cost reduction.**

Index Terms—**Demand response, residential energy management, deep reinforcement learning, deep deterministic policy gradient, reward shaping.**

I. Introduction

With growing populations, ever-increasing living standards, and more appliances now being used in homes, the energy demand of the residential sector has grown considerably in these days, and is expected to increase even further [1]. Thus, an energy management program is required to allow residential users to make informed changes concerning their energy consumption and help the power grid reshape load patterns and reduce peak demand, thereby eliminating potential electricity supply disruptions and the need for backup power plants [2]. A residential energy management center (REMC) in a home can use autonomous load scheduling plans offered by energy management programs with real-time electricity pricing to manage the load energy consumption, reduce user electricity costs, and enhance power grid reliability [3].

To date, extensive efforts have been conducted to the optimal residential energy management. For example, [4] presented a demand response program for household energy management to reduce daily payments, in which the optimization problem was formulated as a mixed integer linear programming model. [5] developed an event-triggered multi-stage secondary control approach for microgrid management to overcome the power sharing error and frequency deviation problems. [6] proposed a model predictive control method to schedule air conditioners in the residential electricity network, aiming to shave peak demand and firm photovoltaic generation capacity. [7] created a delay-tolerant predictive power compensation control strategy for voltage regulation

This work was supported in part by the National Key R&D Program of China under Grant 2021ZD0201300, in part by the National Natural Science Foundation of China under Grant 62003143, in part by the Science and Technology Project of State Grid Corporation of China under Grant 5100-202199557A-0-5-ZN, in part by the Fundamental Research Funds for the Central Universities under Grant HUST 2020kfyXJJS084, Grant 2022SMECP03, in part by the State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources under Grant LAPS21006, in part by the Key Laboratory of Industrial Internet of Things and Networked Control under Grant 2020FF02, and in part by the Open Fund of Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering at Wuhan University of Science and Technology under Grant MTMEOF2021B04. (Corresponding author: Yuemin Ding.)

Renzhi Lu is with the Key Laboratory of Image Processing and Intelligent Control, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China, with the State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources, North China Electric Power University, Beijing 102206, China, with the Key Laboratory of Smart Manufacturing in Energy Chemical Process, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China, with the Key Laboratory of Industrial Internet of Things and Networked Control, Chongqing University of Posts and Telecommunications, Chongqing 400065, China, and with the Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology, Wuhan 430081, China (e-mail: rzlu@hust.edu.cn)).

Zhenyu Jiang is with the Key Laboratory of Image Processing and Intelligent Control, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: m202173166@hust.edu.cn).

Huaming Wu is with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: whming@tju.edu.cn).

Yuemin Ding is with the Department of Electrical and Electronic Engineering, University of Navarra, San Sebastian 20018, Spain (e-mail: yuemin.ding1986@gmail.com).

Dong Wang is with the Key Laboratory of Intelligent Control and Optimization for Industrial Equipment of Ministry of Education, Dalian University of Technology, Dalian 116024, China, and also with the School of Control Science and Engineering, Dalian University of Technology, Dalian 116024, China (e-mail: dwang@dlut.edu.cn).

Hai-Tao Zhang is with the Key Laboratory of Image Processing and Intelligent Control, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China, and also with the State Key Lab of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: zht@mail.hust.edu.cn).

in a high penetration level of photovoltaic energy systems. Most recently, a predictive voltage hierarchical controller is adopted in [8] to enhance the voltage and energy sharing of distributed generations for microgrids. However, designing an efficient energy management program in a residential environment involves two challenges [9]. First, affected by the user commuting behavior and living activity, the REMC has uncertainties in operational interval and time slots of residential appliances, which makes it difficult for an REMC to produce energy management schedules efficiently in response to varying electricity prices. Second, to effectively control residential appliances, precise appliance parameters and models would be predetermined by relevant experts to estimate their operational dynamics and energy demand characteristics. Nevertheless, expert experience is not always available in homes.

Driven by the fast evolution of artificial intelligence, reinforcement learning (RL)-based approaches that avoid the necessity of an accurate mathematical model have recently induced increasing interest in the power grid domain for creating optimal energy dispatch schedules [10]. To be specific, RL comprises a smart agent that gradually learns the optimal control policy via utilizing Markov decision process (MDP) experiences obtained from repeated interactions with the stochastic environment, without prior system knowledge [11]. This provides the agent with powerful capacities to learn the optimal performance behavior when it is difficult or costly to model the system explicitly and accurately. Furthermore, once the learning procedure of RL is accomplished, the acquired policy can be directly deployed for end-to-end real-time decision-making to produce optimal actions instantly for the given system states.

The great success of RL has inspired the development of RL-based approaches for residential energy management. For example, [12] proposed a device-based RL approach for various loads energy management in residential and small commercial building sectors. [13] created a day-ahead energy consumption schedule of thermostatically controlled loads using a batch RL method. [14] designed an RL-based real-time demand response scheme for residential customers to equilibrize energy demand and supply. [15] developed a multi-agent RL structure for building energy management to minimize electricity costs and the induced user dissatisfaction. However, these works employing RL in energy management problems all utilized the traditional Q-learning algorithm and its variants. This kind of RL algorithm calculates the action-value functions for all possible state-action pairs based on a look-up table; thus both state spaces and action spaces are discretized, which is not appropriate for most applications with massive states and actions in the real-world, given that the Q-table becomes greatly large.

To address such dimensionality challenges, a new promising approach, referred to as deep Q-network (DQN), which incorporates RL with deep neural network (DNN) to exploit its powerful function approximation capabilities, has been witnessed recently. A DQN is an extension of Q-learning that employing a DNN to approximate the Q-value so as to handle the continuous multi-dimensional state space problems. For example, [16] explored the benefits of utilizing DQN to accomplish online strategies optimization in residential energy management systems, based on highly dimensional observed data. [17] adopted a distributed DQN to manage household energy consumption profiles, considering the randomness in electricity price and appliance demand. [18] developed a DQN model that shifts residential load from peak to off-peak slots, with the aim of minimizing user electricity costs and reducing grid peak load synchronously. [19] proposed a federated DQN for multiple homes energy management, taking into account the dynamically varying operation conditions of appliances. Although the abovementioned works have demonstrated high-quality DQN performance in continuous state spaces energy management problems, the DNN employed is commonly learned to generate discrete Q-value estimates instead of continuous actions, which severely prevents its advantages from dealing with multi-dimensional continuous action space energy management issues.

To resolve this problem, another RL algorithm of deterministic deep policy gradient (DDPG) approach is further proposed. Rather than approximating the action-value function in DQN, it calculates the possibility of choosing an action at a particular state directly to deal with the continuous action space situations. [20] presented a commercial building load control method based on DDPG to minimize electricity costs and thermal discomfort without information about the building thermal dynamic model. [21] combined the DDPG with prioritized experience replay strategy for residential multi-energy systems to schedule different devices for reducing energy costs, considering the uncertainties in both demand and supply sides. [22] employed a DPPG and federated learning to incentivize residential users to reduce electricity demand during peak time slots while accounting for the indeterminacy in privacy concerns and power flow constraints. [23] proposed a privacy-preserving residential microgrids load scheduling algorithm using a DDPG, without knowing a prior information about customer behavior or model dynamics. Despite the great promise of utilizing RL approaches to solve highly complex energy management problems in dynamic residential environments, challenges remain to be solved. One such pitfall is that discovering the state-action space in stochastic and uncertain environments is really time-consuming to converge, due to the sparse and delayed rewards.

To accelerate the training process of RL approaches in continuous action environment, many efforts have been dedicated to find efficient ways to speed up the convergence rate [24]. Reward shaping (RS) is a kind of general principled methods to improve the learning procedure by providing an additional reward term that would be initialized to the value function [25]. This shaping reward is indeed not come from the environment, but is incorporated by the system designer and estimated on the basis of some prior problem knowledge [26]. The work of [27] proposed a potential-based RS method to improve the RL agent performance, wherein the difference of some potential function is designed over a source and a destination state. It also proved that the RS defined with this way can lead to substantial reductions in learning time for obtaining optimal actions while the optimality of the original optimal policy unchanged.

In this paper, a novel RS-based actor-critic DRL (ACDRL) algorithm for real-time autonomous residential energy management is proposed, with the aim of minimizing electricity bills and sustaining user comfort. Specifically, the proposed RS-ACDRL algorithm featuring an actor-critic architecture combines a DPG with a DQN and an RS mechanism to boost its performance. In this configuration, the actor-critic is realized by two DNNs used for function approximation, in which the actor learns via the Q-value estimated from the critic. More detailed, the actor is represented by a policy network that can learn policies in a high-dimensional continuous state and action spaces, and the critic is represented by a Q-network to exploit the success of the target Q-value and the experience replay to stabilize learning. In addition, an effective RS mechanism is also introduced in episodic ACDRL, attempting to add information about state-action pairs closer to terminal states and thereby increases the convergence speed of the ACDRL. The effectiveness of the proposed RS-ACDRL algorithm for residential energy management is verified via numerous numerical simulations with real-world data considering uncertainties arising from electricity price and appliance demand. Case studies demonstrate the performance of the proposed algorithm by comparing it against the case without scheduling, and another two state-of-the-art RL methods of DQN and DDPG, in terms of the cost reduction, solution optimality, and learning speed.

To emphasize major research gap of the proposed algorithm in light of the related methods in literature, a comparative analysis is carried out in terms of 4 aspects as given in Table I, including the optimization approach, the state space, the action space and the approximate method. In summary, this work contributes in four-fold.

A residential energy management problem is investigated, considering uncertainties in electricity price and appliance demand to reflect realistic scenarios, in which the scheduling procedure is formulated to an MDP without known the transition probability.

A novel RS-ACDRL algorithm blended an actor and critic network with an RS mechanism, that does not require system model information in a multi-dimensional continuous state and action spaces, is proposed to obtain the real-time autonomous optimal energy management policies.

Numerical results based on real-world data demonstrate that the proposed algorithm achieves better performance than the other two baselines, and appears a more beneficial computational property because of its utilization of the RS mechanism.

Case studies verify that the RS-ACDRL algorithm gains approximately 38.57% lower electricity costs than that of the case without scheduling, and further promotes nearly 600 reductions in training episodes when compared with the DDPG.

The rest of this paper is organized as follows. Section II introduces the problem formulation. Section III describes the proposed RS-ACDRL algorithm in detail. Section IV provides simulation results demonstrating the effectiveness of the proposed algorithm. Finally, Section V gives the conclusions and future work.

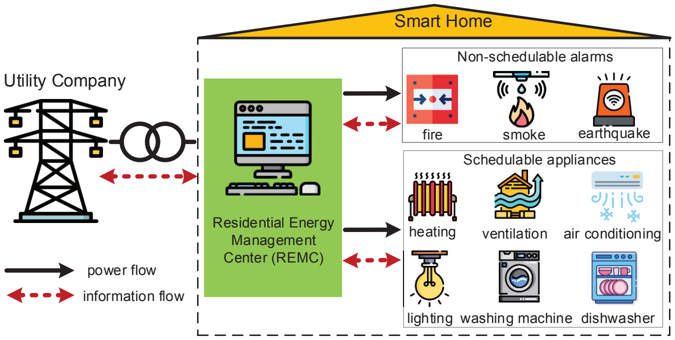


Fig. 1. System model.

II. Problem Formulation

As illustrated in Fig. 1, we consider a smart home equipped with one REMC and diversified residential loads. The REMC, as a coordinator of user load scheduling, is connected to the utility company via a two-way communication network that receives real-time pricing termly from the utility company and then regulates the various loads energy consumption, with the goal of minimizing electricity bills while satisfying comfort requirement. Due to the control actions of the REMC are taken sequentially to the different loads, MDP affords a logical structure to build the interactions between the REMC and the loads. The following sections present the formulations of residential loads and MDP.

A. Residential Loads

Loads in a household are usually divided into two categories according to their priorities and characteristics: nonschedulable and schedulable. The energy demands of nonschedulable loads must be satisfied completely in their operation time as needed and do not respond to price variations (e.g., alarm systems). In contrast, schedulable loads can be stopped, shifted, or adjusted to freely change their working time and energy demand, abiding by some functional constraints (e.g., dishwashers). For each load $n \in \{1, ..., N\}$, its control variable is represented by $c_n^t \in \{0,1\}$ denoting the operating status, i.e., $c_n^t = 1$ if the load n operates with a task at time t or $c_n^t = 0$ if otherwise. $\left[T_n^{ini}, T_n^{end}\right]$ indicates the scheduling window corresponding to a working period that includes the initial time T_n^{ini} and the end time T_n^{end} for the task; and E_n^t represents the required energy consumption to function the task.

1) Nonschedulable Load: A nonschedulable load has tight energy demand requirements that have to be satisfied with no intervention. As soon as it begins running, it should be worked consecutively and will not be regulated. Therefore, the energy consumption of nonschedulable load n, non is equivalent to its energy demand:

$$E_{n,non}^t = e_{n,non} \cdot c_{n,non}^t \tag{1}$$

$$c_{n,non}^t = \begin{cases} 1, & t \in \left[T_{n,non}^{ini}, T_{n,non}^{end}\right] \\ 0, & otherwise \end{cases} \tag{2}$$

where $e_{n,non}$ is the energy demand and $c_{n,non}^t$ is the control variable at time t of the nonschedulable load n, non.

TABLE I
RESEARCH GAP OF THE PROPOSED ALGORITHM WITH OTHER LITERATURES

References	Optimization Approach	State Space	Action Space	Approximate Method
Ref [4]–[8]	model-based	-	-	-
Ref [12]–[15]	learning-based	discrete	discrete	deep neural network
Ref [16]–[19]	learning-based	continuous	discrete	deep neural network
Ref [20]–[23]	learning-based	continuous	continuous	deep neural network
This work	learning-based	continuous	continuous	deep neural network + long short-term memory + reward shaping

2) Schedulable Load: Schedulable loads include all loads whose working time can be shifted or those requiring energy that can be regulated throughout the day in response to price variations, and can also be broken into two technology groups: shiftable and controllable.

Shiftable loads can be regulated from on-peak to off-peak hours, effectively reducing not only the peak energy demand, but also the electricity bill for the user. Assume that a shiftable load n, sh requires continuous manipulation of $T_{n,sh}^{need}$ time periods to accomplish a task. The starting time of the task is denoted by $T_{n,sh}^{sta}$. Then, the energy consumption of the shiftable load n, sh is

$$E_{n,sh}^{t} = e_{n,sh} \cdot c_{n,sh}^{t} \tag{3}$$

$$c_{n,sh}^{t} = \begin{cases} 1, & t \in \left[T_{n,sh}^{sta}, T_{n,sh}^{sta} + T_{n,sh}^{need}\right] \\ 0, & otherwise \end{cases} \tag{4}$$

$$T_{n,sh}^{ini} \le T_{n,sh}^{sta} \le T_{n,sh}^{end} - T_{n,sh}^{need} \tag{5}$$

where Eq. (4) restricts the control variable $c_{n,sh}^{t}$ to operate continuously because the shiftable load n, sh cannot be interrupted during its operation horizon, and Eq. (5) limits the load n, sh to complete the task within the deadline.

In contrast, the energy demand of the controllable load is continuously adjustable in response to prices. Thus, the energy consumption of a controllable load n, con at time slot t is given by:

$$E_{n,con}^{t} = e_{n,con}^{\min} + \left(e_{n,con}^{\max} - e_{n,con}^{\min}\right) \cdot c_{n,con}^{t} \tag{6}$$

$$c_{n,con}^{t} = \begin{cases} I_{n,con}^{t} \cdot 1, & t \in \left[T_{n,con}^{ini}, T_{n,con}^{end}\right] \\ 0, & otherwise \end{cases} \tag{7}$$

where $I_{n,con}^{t}$ is a continuously varying decimal indicating that the controllable load n, con can be regulated between the minimum energy demand $e_{n,con}^{\min}$ and maximum energy demand $e_{n,con}^{\max}$ respectively, with an elastic energy consumption.

B. Markov Decision Process Formulation

In the smart household under consideration, the residential load energy consumption level at the next time slot relies on the current energy demand level and the control variable in the current time slot according to Eqs. (1), (3), (6), which is independent of foregoing actions and states, so the residential loads energy consumption scheduling is formulated as an MDP. In the MDP, the sequential decision-making process is represented by a four-tuple (S, A, R, S'), where S and S' are the state sets, A is the action sets, and $R(s, a, s')$ is the reward function. The following sections present the key elements of the MDP, containing the agent, environment, state, action, and reward.

1) Agent: The decision-maker (i.e., REMC) denotes the agent that learns how to accelerate its control policies gradually to maximize rewards via experiences obtained by repeatedly interacting with the environment. Depending on the presented smart home system, the agent is used to realize the switching or adjustment of the residential loads.

2) Environment: The other objects outside the agent (i.e., nonschedulable and schedulable loads) to be controlled represent the environment, as depicted in Fig. 1.

3) State: The state s^t at time slot t is regarded as a feedback of the REMC agent, which reflects its control action effect on the residential load status. It consists of three types of information:

$$s^{t} = \left(t, E_{n,non}^{t}, E_{n,sh}^{t}, E_{n,con}^{t}, P^{t-23}, P^{t-22}, \ldots, P^{t-1}, P^{t}\right) \tag{8}$$

where t denotes the time step identifier given that the energy demand of residential loads and electricity prices from utility company are repetitive with a daily pattern; $E_{n,non}^{t}, E_{n,sh}^{t}, E_{n,con}^{t}$ represent the energy consumption of the endogenous physical state features of residential loads that are managed and can be influenced by the agent actions; and $P^{t-23}, P^{t-22}, \ldots, P^{t-1}, P^{t}$ indicate the electricity prices over the past 24 time slots, which are accessed through the utility company and can be considered as exogenous information state features that are needed to make a control decision and calculate the reward. These features are characterized by the intrinsic variability and uncertainty of the system and are independent of the agent actions.

4) Action: Given the system state s^t at time slot t, the action a^t of the agent is to ideally determine the energy consumption amount of the various residential loads, defined as:

$$a^{t} = \left(c_{n,non}^{t}, c_{n,sh}^{t}, c_{n,con}^{t}\right) \tag{9}$$

where $c_{n,non}^{t}$ is the constant control variable of the nonschedulable loads, and $c_{n,sh}^{t}$ and $c_{n,con}^{t}$ are the binary and continuous control variables of the schedulable loads, respectively.

5) Reward: The goal of the agent is to schedule the elastic loads to minimize user electricity cost, i.e., regulating energy demand from high electricity price time periods to low electricity price time periods. However, the scheduled energy consumption can lead to user dissatisfaction, as it may deviate from a user preferred working time or required energy demand.

Thus, from a user perspective, the corresponding reward r^t includes three parts:

$$r^t = r^t_{com} - r^t_{wait} - r^t_{ele} \tag{10}$$

$$r^t_{com} = \sum_{n=1}^{N} \alpha_{n,con} \cdot \left(E^t_{n,con} - e^{\min}_{n,con}\right)^2 \tag{11}$$

$$r^t_{wait} = \sum_{n=1}^{N} \beta_{n,sh} \cdot \left(T^{sta}_{n,sh} - T^{ini}_{n,sh}\right) \tag{12}$$

$$r^t_{ele} = \sum_{n=1}^{N} P^t \cdot \left(E^t_{n,non} + E^t_{n,sh} + E^t_{n,con}\right) \tag{13}$$

where r^t_{com} denotes the user comfort index, which is defined by a quadratic function form. $\alpha_{n,con}$ is a controllable load-dependent parameter measuring the user sensitivity toward the energy consumption reduction, i.e., a load with a smaller $\alpha_{n,con}$ prefers to consume more energy to enhance its comfort level, and vice versa. r^t_{wait} indicates the waiting penalty for a shiftable load to begin its task, which is calculated based on the deviation from its initial working time. For example, a dishwasher commonly functions during a specific working period $\left[T^{ini}_{n,sh}, T^{end}_{n,sh}\right]$ (e.g., 6pm-10pm), which can be shifted from on-peak time slots to off-peak time slots. For example, if the dishwasher starts to function at $T^{sta}_{n,sh}$ (i.e., 9pm), then the waiting time would be $T^{sta}_{n,sh} - T^{ini}_{n,sh}$ (i.e., 3h). $\beta_{n,sh}$ is a weighting factor that reflects the linear relationship slope of the waiting inconvenience; a larger $\beta_{n,sh}$ will result in a bigger penalty. r^t_{ele} represents the electricity cost for loads energy consumption, and P^t is the dynamic real-time electricity price.

C. Objective

For each time slot t, the REMC agent makes control decisions about residential load energy consumption scheduling according to a set of available information and forms a MDP sequences of environmental states, actions, and rewards: $s^1, a^1, r^1; s^2, a^2, r^2; \ldots; s^t, a^t, r^t; \ldots; s^T, a^T, r^T$. The return $R = \sum_{t=1}^{T} \gamma r^t$ is the sum of the discounted reward from the initial learning step thereafter, where $\gamma \in [0, 1]$ is a discounted factor that represents the relative importance of long-term (future) rewards and short-term (immediate) reward. The aim of the REMC agent is to find an optimal policy μ^* (a policy μ maps the possibility of choosing an available action to the corresponding state) that maximizes the cumulative discounted reward over all feasible policies, as represented by the optimal Q-value function $Q^*(s, a) = E\left[R | s^t = s, a^t = a; \mu^*\right]$, which constitutes an estimation of the discounted, accumulative, and expected reward, considering an action a^t at state s^t, and following the policy μ^* from the succeeding states onward.

To analytically determine the optimal $Q^*(s, a)$, the transition probability from s^t to s^{t+1} is needed to indicate the uncertainty. This requires a precise mathematical model, but in the problem under consideration, the state transition from s^t to s^{t+1} is influenced not only by the agent control actions a^t but also by the stochastic exogenous features, including the price pattern P^t and energy demand e^t_n. It is very challenging to identify an explicit joint probability distribution model to exactly reflect such indeterminacy, which is affected by many factors, including the utility company, weather conditions, and user behavior.

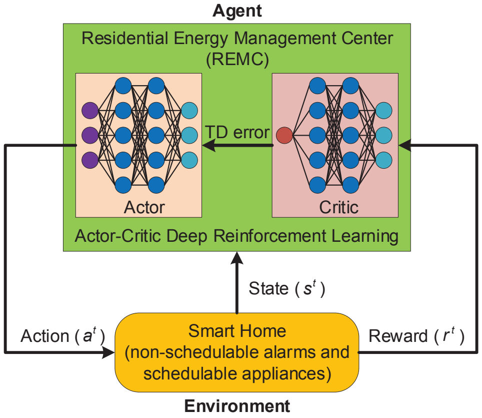

Agent
Residential Energy Management Center (REMC)
TD error
Actor
Critic
Actor-Critic Deep Reinforcement Learning
State (s^t)
Action (a^t)
Reward (r^t)
Smart Home (non-schedulable alarms and schedulable appliances)
Environment

Fig. 2. Actor-critic deep reinforcement learning framework.

To address this issue, an RS-ACDRL algorithm is proposed to allow the agent to overcome the uncertainties within in real-world dataset and implicitly learn the state transition probability via machine learning techniques. Fig. 2 presents a diagram of interaction between the agent and the environment in the MDP of the proposed RS-ACDRL algorithm. The agent behaves in a dynamic environment by acting sequentially over a sequence of time slots. At each time slot t, the agent observes the environment state s^t and utilizes the RS-ACDRL algorithm to refresh its control actions a^t (expressed by $a^t = \mu(s^t)$). The environment then executes action a^t, moves to the next state s^{t+1} and generates a reward r^t. This is finally returned to the agent to determine the next time slot control action a^{t+1}. It should be pointed out that the agent does not have any prior knowledge about how the reward and next state are associated with each action. Instead, the agent learns this linking via constantly interacting with the stochastic environment to minimize the electricity cost while meeting the satisfaction index of the user.

III. Reward Shaping-based Actor-Critic Deep Reinforcement Learning Algorithm

A. Overall Scheme

This section presents an RS-ACDRL algorithm to perform optimal residential energy management. The algorithm constructs an actor-critic structure and utilizes two DNNs that learn approximations for both the value function and policy function, wherein the detailed workflow is depicted in Fig. 3. The actor network (represented by DPG), parameterized by θ^μ (light yellow color), generates the executable action a^t given the current state s^t, while the critic network (represented by DQN), parameterized by ϕ^Q (light red color), estimates the Q-value function $Q\left(s^t, a^t | \phi^Q\right)$ for the state-action pair $\left(s^t, a^t, r^t, s^{t+1}\right)$. Because the critic just requires to generate an estimated Q-value for the action carried out by the actor, the proposed RS-ACDRL algorithm can operate on continuous action spaces [28]. Through controlling continuous actions,

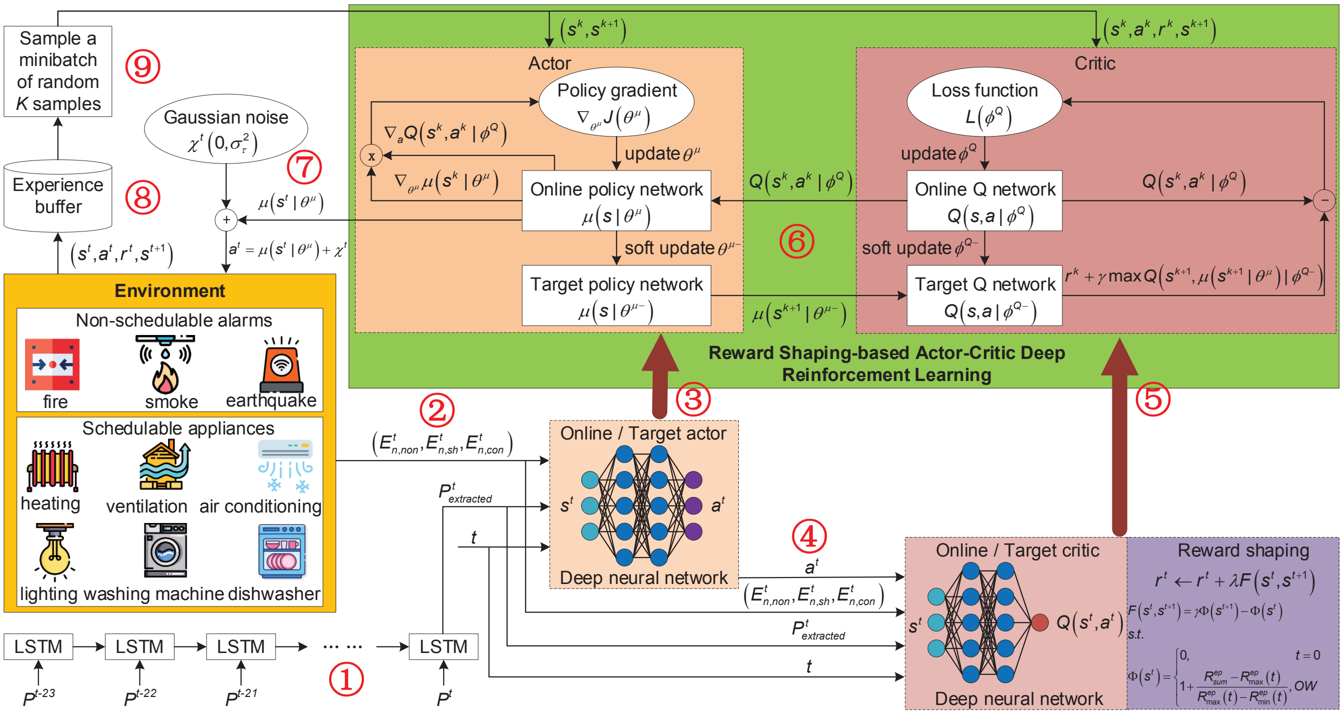

Sample a minibatch of random K samples
⑨
Gaussian noise $\chi^t(0,\sigma_\tau^2)$
⑦
Experience buffer
⑧
$\mu(s^t|\theta^\mu)$
$a^t=\mu(s^t|\theta^\mu)+\chi^t$
(s^t,a^t,r^t,s^{t+1})
(s^k,s^{k+1})
(s^k,a^k,r^k,s^{k+1})
Actor
Policy gradient $\nabla_{\theta^\mu}J(\theta^\mu)$
$\nabla_a Q(s^k,a^k|\phi^Q)$
$\nabla_{\theta^\mu}\mu(s^k|\theta^\mu)$
update θ^μ
Online policy network $\mu(s|\theta^\mu)$
soft update $\theta^{\mu-}$
Target policy network $\mu(s|\theta^{\mu-})$
$Q(s^k,a^k|\phi^Q)$
⑥
$\mu(s^{k+1}|\theta^{\mu-})$
Critic
Loss function $L(\phi^Q)$
update ϕ^Q
Online Q network $Q(s,a|\phi^Q)$
soft update ϕ^{Q-}
Target Q network $Q(s,a|\phi^{Q-})$
$Q(s^k,a^k|\phi^Q)$
$r^k+\gamma\max Q(s^{k+1},\mu(s^{k+1}|\theta^\mu)|\phi^{Q-})$
Reward Shaping-based Actor-Critic Deep Reinforcement Learning
Environment
Non-schedulable alarms
fire
smoke
earthquake
Schedulable appliances
heating
ventilation
air conditioning
lighting
washing machine
dishwasher
②
$(E^t_{n,non},E^t_{n,sh},E^t_{n,con})$
$P^t_{extracted}$
t
③
Online / Target actor
s^t
a^t
Deep neural network
④
a^t
$(E^t_{n,non},E^t_{n,sh},E^t_{n,con})$
$P^t_{extracted}$
t
⑤
Online / Target critic
s^t
$Q(s^t,a^t)$
Deep neural network
Reward shaping
$r^t\leftarrow r^t+\lambda F(s^t,s^{t+1})$
$F(s^t,s^{t+1})=\gamma\Phi(s^{t+1})-\Phi(s^t)$
s.t.
$\Phi(s^t)=\begin{cases}0, & t=0\\ 1+\frac{R^{ep}_{sum}-R^{ep}_{\max}(t)}{R^{ep}_{\max}(t)-R^{ep}_{\min}(t)}, & OW\end{cases}$
LSTM
LSTM
LSTM
LSTM
P^{t-23}
P^{t-22}
P^{t-21}
P^t
①

Fig. 3. The overall workflow of RS-ACDRL.

the proposed algorithm explores the action spaces more thoroughly and avoids the discretization computational cost, and also produces a smooth action control strategy to schedule the residential loads.

Specifically, the proposed RS-ACDRL algorithm first employs long short-term memory (LSTM) neural network to extract identifiable price data features, which is a critical process for improving the DNN approximation quality (mark ① in Fig. 3). Given that the electricity price varies in a near-periodic manner and has a physical chronological order, it is rational to extrapolate future electricity price tendency from former electricity prices [29]. The LSTM networks are very familiar for their capability to capture the time-series data dependencies, so the electricity price tendency in this work is modeled by an LSTM network, in which the input is the previous 24-h prices and the output is the extracted price pattern. After that, the output of the LSTM network concatenated with the time slot and energy consumption information, are inputted into both DNNs to approximate the optimal function. The actor first extracts feature states from the multidimensional state s^t including the time slot, energy consumption, and extracted price pattern information and outputs an action a^t derived by policy $\mu\left(s^t|\theta^\mu\right)$ (marks ② and ③ in Fig. 3). Then, the same feature state s^t plus the action a^t are inputted to the critic, whose output is a scalar estimated Q-value function for that corresponding state-action pair with the aid of an RS mechanism and employs the reward value received from the stochastic environment to regulate the precision of the estimated Q-value function $Q\left(s^t,a^t|\phi^Q\right)$ by changing its parameter ϕ^Q (marks ④ and ⑤ in Fig. 3). Finally, the actor network updates its parameter θ^μ based on the Q-value function of the critic network to improve the policy and the action taken on each state (mark ⑥ in Fig. 3).

More specifically, the critic network criticizes the policy by providing an approximation of the Q-value function via minimizing the following loss function:

$$L\left(\phi^Q\right)=E_{s,a,r,s'\sim D}\left[y^t-Q\left(s^t,a^t|\phi^Q\right)\right]^2_{|a^t=\mu(s^t|\theta^\mu)} \tag{14}$$

$$y^t=r\left(s^t,a^t\right)+\gamma\max Q\left(s^{t+1},\mu\left(s^{t+1}|\theta^\mu\right)|\phi^{Q-}\right) \tag{15}$$

which leads to the following gradient:

$$\nabla_{\phi^Q}L\left(\phi^Q\right)= E_{s,a,r,s'\sim D}\left[\left(y^t-Q\left(s^t,a^t|\phi^Q\right)\right)\frac{\partial Q\left(s^t,a^t|\phi^Q\right)}{\partial\phi^Q}\right] \tag{16}$$

Then, the gradient of the actor network can be written as:

$$\nabla_{\theta^\mu}J\left(\theta^\mu\right)=\nabla_a Q\left(s^t,a^t|\phi^Q\right)_{|a^t=\mu(s^t|\theta^\mu)}\nabla_{\theta^\mu}\mu\left(s^t|\theta^\mu\right) \tag{17}$$

However, the DQN tends to exhibit unstable when use a DNN to approximate the Q-value function. First, the online Q-network $Q\left(s^t,a^t|\phi^Q\right)$ being updated in Eq. (14) is also utilized to compute the target value y^t in Eq. (15); thus, the update of Q-value is apt to diverge. A valid way to address this oscillation is to bring in a target network for the critic and actor respectively, represented by $\mu\left(s^t|\theta^{\mu-}\right)$ and $Q\left(s^t,a^t|\phi^{Q-}\right)$, and use them to compute the target values. The parameters of these two target networks are then updated by periodically tracking the online networks $\mu\left(s^t|\theta^\mu\right)$ and $Q\left(s^t,a^t|\phi^Q\right)$. The principle behind this soft change is to enforce the target values to vary slowly and thereby stabilize the learning process. Second, at each iteration, state transition samples are collected as the agent interacts with the stochastic environment sequentially, which means that these transitions are highly correlated. Learning from these consecutive tuples may result in a divergence in the training. To overcome this problem, an experience reply buffer D, which stores collected past experiences (an experience is a transition tuple $\left(s^t, a^t, r^t, s^{t+1}\right)$) and samples uniformly a minibatch (of size K) of experiences, is employed to update the actor and critic at each time step. Sampling in the experience replay buffer destroys the temporal correlations between the selected tuples, thus stabilizing the training. Besides, the experience replay buffer allows the experiences to be reused, thereby enhancing the sampling efficiency.

To further improve the convergence speed and accelerate the learning process, an RS mechanism is utilized. The rationale behind RS is to improve the stochastic environment reward feedback by importing extra rewards to make progress toward discovering high reward actions, which helps RL algorithms reduce the required training transition number, to obtain the optimal policy more rapidly [27]. A potential-based RS can result in massive reductions in learning time to boost the convergence process and guarantee the optimal policy learned keeps optimality under the effect of RS [30], which is defined as:

$$F\left(s^t, s^{t+1}\right) = \gamma \Phi\left(s^{t+1}\right) - \Phi\left(s^t\right) \tag{18}$$

where Φ is a to-be-defined potential function. In this work, we carefully design the potential function as follows:

$$\Phi\left(s^t\right) = \begin{cases} 0, & t = 0 \\ 1 + \frac{R_{sum}^{ep} - R_{\max}^{ep}(t)}{R_{\max}^{ep}(t) - R_{\min}^{ep}(t)}, & otherwise \end{cases} \tag{19}$$

where R_{sum}^{ep} is the sum of rewards in the current episode; $R_{\max}^{ep}(t)$ and $R_{\min}^{ep}(t)$ are the maximum and minimum value of episode reward until now. This shaping reward is added to each environmental reward for every transition in each episode. As such, the Eq. (15) in the presence of potential-based RS becomes:

$$\begin{aligned} y^t = r\left(s^t, a^t\right) + \lambda F\left(s^t, s^{t+1}\right) \\ + \gamma \max Q\left(s^{t+1}, \mu\left(s^{t+1} | \theta^{\mu}\right), | \phi^{Q-}\right) \end{aligned} \tag{20}$$

where λ is a tuning parameter that weights the shaped term $\gamma F\left(s^t, s^{t+1}\right)$. RS not only enables the agent to derive its optimal policy rapidly from each effective trial but also presents the agent from choosing bad actions in certain states, thus improving the learned policy quality.

The gradient decent algorithm is then used to update the parameters of the online actor and critic networks respectively by performing the following computations:

$$\theta^{\mu} \leftarrow \theta^{\mu} - \eta^{\theta} \nabla_{\theta^{\mu}} J\left(\theta^{\mu}\right) \tag{21}$$

$$\phi^{Q} \leftarrow \phi^{Q} - \eta^{\phi} \nabla_{\phi^{Q}} L\left(\phi^{Q}\right) \tag{22}$$

where η^{θ} and η^{ϕ} are the corresponding learning rates. The target actor and critic networks are updated to track the online actor and critic networks according to:

$$\theta^{\mu-} \leftarrow \tau \theta^{\mu} + (1 - \tau) \theta^{\mu-} \tag{23}$$

$$\phi^{Q-} \leftarrow \tau \phi^{Q} + (1 - \tau) \phi^{Q-} \tag{24}$$

where a small τ with $\tau \ll 1$ should be selected to improve learning stability.

B. Detailed Algorithm

The proposed RS-ACDRL algorithm for residential energy management consists of two parts: the training algorithm and the execution algorithm, as illustrated by Algorithm 1 and Algorithm 2 below.

Algorithm 1 Training process of the RS-ACDRL

```
Input: Electricity prices, time information, energy consumption and other related parameters.
Output: Optimal parameters of the critic and actor networks.
Randomly initialize the online critic and actor networks with parameters $\phi^Q$ and $\theta^\mu$;
Initialize the target critic and actor networks with parameters $\phi^{Q-} \leftarrow \phi^Q$ and $\theta^{\mu-} \leftarrow \theta^\mu$;
Initialize the replay memory buffer $D$.
for $episode = 1$ to $M$ do
    Observe the residential environment state $s^t$;
    for $t = 1$ to $T$ do
        Select action $a^t = \mu\left(s^t|\theta^\mu\right) + \chi^t$ according to the policy network and exploration noise;
        Execute action $a^t$, receive the reward $r^t$, and observe next state $s^{t+1}$;
        Store transitions $\left(s^t, a^t, r^t, s^{t+1}\right)$ in $D$;
        Sample a random mini-batch of $K$ transitions $\left(s^k, a^k, r^k, s^{k+1}\right)$ from $D$;
        Set $y^k = r\left(s^k, a^k\right) + \lambda F\left(s^k, s^{k+1}\right) + \gamma \max Q\left(s^{k+1}, \mu\left(s^{k+1}|\theta^\mu\right), |\phi^{Q-}\right)$;
        Minimize the loss:
        $L\left(\phi^Q\right) = \frac{1}{k}\sum_k E_{s,a,r,s' \sim D}\left[y^k - Q\left(s^k, a^k|\phi^Q\right)\right]^2$
        Leading to the following gradient:
        $\nabla_{\phi^Q} L\left(\phi^Q\right) = \left[\left(y^k - Q\left(s^k, a^k|\phi^Q\right)\right) \frac{\partial Q\left(s^k, a^k|\phi^Q\right)}{\partial \phi^Q}\right]$
        Update the critic network:
        $\phi^Q \leftarrow \phi^Q - \eta^\phi \nabla_{\phi^Q} L\left(\phi^Q\right)$
        Calculate the sampled policy gradient:
        $\nabla_{\theta^\mu} J\left(\theta^\mu\right) = \frac{1}{k}\sum_k \nabla_a Q\left(s^k, a^k|\phi^Q\right) \nabla_{\theta^\mu} \mu\left(s^k|\theta^\mu\right)$
        Update actor network:
        $\theta^\mu \leftarrow \theta^\mu - \eta^\theta \nabla_{\theta^\mu} J\left(\theta^\mu\right)$
        Update target critic and actor networks:
        $\phi^{Q-} \leftarrow \tau\phi^Q + (1-\tau)\phi^{Q-}$
        $\theta^{\mu-} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu-}$
    end for
end for
```

1) Training Algorithm: Algorithm 1 corresponds to the training process of the RS-ACDRL. The inputs are the electricity price, time information, energy consumption, and other parameters defined in Section II. Its output is the optimal parameters θ^{μ} and ϕ^{Q} of the actor and critic network.

First, the parameters ϕ^{Q} and θ^{μ} of the online networks $Q\left(s, a | \phi^{Q}\right)$ and $\mu\left(s | \theta^{\mu}\right)$ are randomly initialized. The target networks $Q\left(s, a | \phi^{Q-}\right)$ and $\mu\left(s | \theta^{\mu-}\right)$ then initialize the weights by copying, i.e., $\phi^{Q-} \leftarrow \phi^{Q}$ and $\theta^{\mu-} \leftarrow \theta^{\mu}$. The replay memory buffer D is also initialized to store the transition tuple.

Next, the algorithm begins running with episodic iterations. At each time slot of each episode, the REMC first observes the states s^t of appliances, and selects actions $a^t = \mu\left(s^t | \theta^{\mu}\right) + \chi^t$

based on the current policy and the exploration noise (mark ⑦ in Fig. 3). When selecting actions, it is essential to keep a proper balance between exploitation and exploration. Exploitation means that the agent makes full use of the current information to select the best actions, while exploration refers to the agent explores more useful knowledge via attempting different available actions in the action spaces. In this work, an exploration policy is constituted by appending a random noise process χ^t to the actor policy to help the agent explore the environment thoroughly. The REMC then executes action a^t on the schedulable loads, and the reward r^t and new state s^{t+1} are returned from the residential environment.

Correspondingly, the transition sample $\left(s^t, a^t, r^t, s^{t+1}\right)$ is stored in the replay memory buffer D to train the critic and actor networks (mark ⑧ in Fig. 3). First, K transitions $\left(s^k, a^k, r^k, s^{k+1}\right)$ are randomly sampled for training DNNs (mark ⑨ in Fig. 3). As shown in Lines 11–13, $Q\left(s^k, a^k|\phi^Q\right)$ and y^k produced by the online critic network and target critic network are utilized to compute mean square error loss. Through minimizing the loss function, the parameter of the critic network is updated in Line 17. Then, the sampled policy gradient is calculated in Line 19, which is utilized to refresh the parameter of the actor network in Line 21. Finally, the parameters of the target critic and actor networks are updated in Lines 23–24.

After the training procedure, the parameters ϕ^Q and θ^μ will be output for the residential energy management.

2) Execution Algorithm: Algorithm 2 presents the execution process of the RS-ACDRL. The parameters of the critic and actor networks trained by Algorithm 1 are loaded when the training process is finished, and the actor network is reserved for real-time decision-making. In the loop starting from Line 2, the actor network is utilized to produce the residential energy management schedules from time slot t to T. At each time slot, the actor network calculates policy value $\mu\left(s^t|\theta^\mu\right)$ based on the observed state features. Then, the load control action a^t is selected in Line 5 as $\mu\left(s^t|\theta^\mu\right) + \chi^t$. Because only the current state s^t is used to make decisions, the proposed RS-ACDRL control algorithm requires no prior knowledge of system uncertain parameters or residential load dynamics.

Algorithm 2 Execution process of the RS-ACDRL

```
1: Load the actor network parameter $\theta^\mu$ trained by Algorithm 1.
2: for time step $t$ to $T$ do
3:     Observe system $s^t$;
4:     Calculate actor network $\mu\left(s^t|\theta^\mu\right)$;
5:     Select action $a^t = \mu\left(s^t|\theta^\mu\right) + \chi^t$;
6:     Execute action $a^t$ in residential environment and observe next state $s^{t+1}$.
7: end for
```

IV. Numerical Results

A. Case Studies

For case studies, we consider three nonschedulable loads (the fire (L1), smoke (L2), and earthquake (L3) alarms), and six schedulable appliances, including four controllable loads (a heating (L4), a validation (L5), an air conditioning (L6), and a lighting (L7)), and two shiftable loads (a dishwasher (L8) and a washing machine (L9)). Table II lists the corresponding parameters of the loads [31], [32]. For the proposed RS-ACDRL algorithm, two DNNs for the actor and critic are employed to learn the optimal residential energy management strategy. The corresponding parameters of the RS-ACDRL methodology are listed in Table III, i.e., the hidden layers and neurons amount of actor and critic, activation function, discounted factor, replay buffer size, maximum training episode, optimizer, learning rate, soft updating rate and minibatch size are determined by some empirical guidelines in the relevant literature survey and extensive preliminary performance analysis [33], [34]. Specifically, the actor has three hidden layers with 128, 64, and 32 neurons, and the critic has two hidden layers with 64 and 32 neurons, wherein all hidden layers are realized by rectified non-linearity (ReLU) as the activation function. The output layer of the actor is a softsign layer to bound the continuous actions. These are collected according to common practice recommended by the machine learning community. For the critic, the discounted factor γ is set to 0.95 so that the proposed algorithm can obtain a foresighted strategy, and L2 regularization is used in its loss function to avoid large weights. The size of the replay buffer D is 10^6, and the maximum episode is set to 1000. Each episode contains 72 hours. The Adam is utilized to optimize the two DNN parameters with a learning rate η^μ and η^Q of 0.01 respectively, which are the same as that used in [33]. The target network soft updating rate τ is set to 0.001, and the minibatch size of K is set to 1024 in the training process. In the LSTM network, a 24 neuron units are utilized to extract the feature from the past 24-h electricity prices. Then, this feature $P^t_{extracted}$ is fed into the two DNNs, concatenated with the measured energy consumption and time information. The algorithm code is programmed in Python with TensorFlow, an open-source machine learning package developed by Google Brain.

The performance of the proposed algorithm is evaluated under a real-world scenario, wherein the hourly electricity prices [35] from January 1 to October 31 2020 are selected to train the algorithm, and the prices from November 1 to December 31 2020 are used for performance evaluation. All case studies are conducted on a computer with an Intel Core i7-4790 CPU, 3.60 GHz, 16GB RAM, and one Nvidia 1080 GPU.

B. Energy Management

The proposed RS-ACDRL algorithm is trained with 1000 episodes to learn the optimal residential energy management strategy. The evolution of the cumulative rewards during the training procedure is shown in Fig. 4. Specifically, the agent obtains a very low reward in the first 100 episodes, since the parameters of DNNs are originally initialized and the actions are randomly selected to explore the environment. However, with each iteration, the cumulative reward continues to increase from 100 episodes onward, due to the actions selected by the DNNs whose parameters are optimized concurrently. Finally, the agent converges to an optimal value after nearly 300

TABLE II
PARAMETERS OF DIFFERENT RESIDENTIAL LOADS

Device Type	Nonschedulable			Schedulable					
				Controllable				Shiftable	
Device Name	L1	L2	L3	L4	L5	L6	L7	L8	L9
Energy Demand (kWh)	0.03	0.03	0.05	0 - 1.4	0 - 1.4	0 - 1.4	0 - 1.4	0.75	0.70
Working Period	24h	24h	24h	24h	24h	24h	24h	6pm - 10pm	6pm - 10pm
$\alpha_{n,con}$	-	-	-	0.013	0.014	0.015	0.016	-	-
$\beta_{n,sh}$	-	-	-	-	-	-	-	0.4	0.1
$T_{n,sh}^{need}$	-	-	-	-	-	-	-	1h	2h

TABLE III
THE PARAMETER VALUES OF THE RS-ACDRL METHODOLOGY

Parameter description	Value
hidden layers and neurons amount of the actor	3 / 128, 64, 32
hidden layers and neurons amount of the critic	2 / 64, 32
activation function	ReLU
discounted factor γ	0.95
replay buffer size D	10^6
maximum training episode	1000
optimizer	Adam
learning rate η^{μ} and η^{Q}	0.01 / 0.01
soft updating rate τ	0.001
minibatch size K	1024

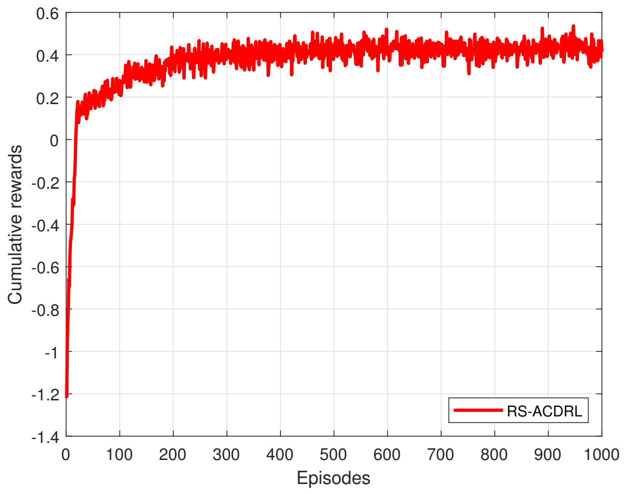


Fig. 4. Convergence process of the proposed algorithm.

episodes, with small oscillations due to the encouragement of the exploration strategy. This process indicates that the agent gradually evolves and successfully learns a stable policy to maximize the cumulative rewards. After the training procedure (Algorithm 1) of the RS-ACDRL is complete, the weights of the actor network at convergence are loaded to determine real-time residential load energy management decisions (Algorithm 2).

To better illustrate the performance of the learned control policies of the proposed algorithm, Figs. 5 and 6 present the obtained energy consumption profiles of nine residential loads, with and without scheduling on a three consecutive test days from December 29 to 31 in 2020 that have different electricity price patterns. When the proposed algorithm is applied, all schedulable loads consume more energy when the electricity prices are low and reduce their demand when electricity prices are high, such that the energy consumption of each residential load is properly scheduled in response to the hourly dynamic electricity prices. For the case without scheduling, the shiftable loads are operated immediately as soon as they require to function, and the controllable loads are operated at their maximum energy demand. Clearly, the loads have no incentive to shift or reduce their energy consumption. Table IV lists the corresponding system cumulative electricity cost; the cost with the proposed algorithm is reduced significantly by 38.57% compared to the case without scheduling, validating the effectiveness of the proposed algorithm in optimizing the real-time residential load energy consumption.

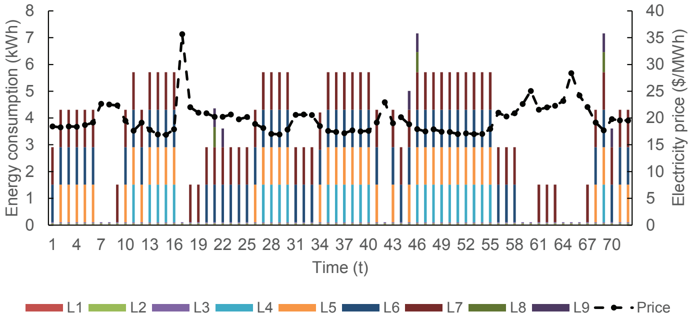


Fig. 5. Aggregated energy consumption of all loads with scheduling.

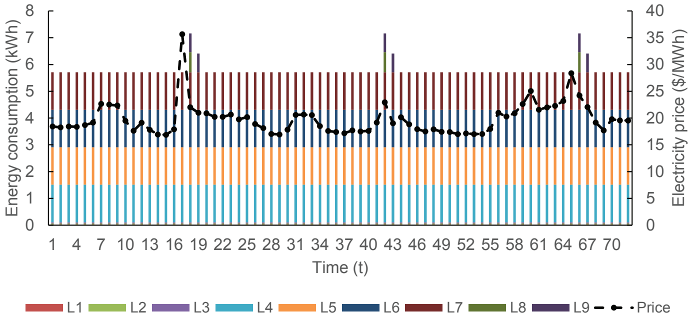


Fig. 6. Aggregated energy consumption of all loads without scheduling.

TABLE IV
The Cumulative Electricity Cost

	With Scheduling	Without Scheduling
Total cost ($)	5.079	8.268

C. Performance Comparison

To further demonstrate the superiority of the proposed RS-ACDRL algorithm, another two methods of DQN and DDPG are selected to compare the performance. For the DQN, it employs a DNN to approximate the Q-value for each discrete action, and then selects the corresponding action with the highest Q-value at a given state. To apply the DQN, the $I^{t}_{n,con}$ of the controllable load is discretized into six decimal values representing 0, 0.2, 0.4, 0.6, 0.8 and 1.0, following a similar practice adopted in [36]. For the DDPG, the DNN takes a state vector as input and outputs a Gaussian policy for each continuous action dimension, in which the probability distribution of the action is represented with a Gaussian distribution, and the DNN is utilized to estimate its mean and variance.

Fig. 7 illustrates the performance of the three examined approaches in terms of their policy quality and learning speed. As illustrated in the figure, the average cumulative reward is negative during the initial learning phases as the agents are gathering experiences by casually exploring different actions. Whereas, as the learning procedure progresses and more experiences are collected, the cumulative rewards turn positive, keep increasing, and eventually reaching the maximum for all three approaches. The RS-ACDRL promotes its control policy in a higher accumulated reward than the DQN, attributing to its ability to exploit more accurate information from multidimensional continuous action space and generate more efficient control strategies, differ from the naive discretization manner adopted in DQN. Furthermore, RS-ACDRL exhibits favorable convergence properties compared to DDPG with regard to learning speed, given that the RS-ACDRL incorporates the target networks and the RS mechanism, facilitating the agent to improve its policy rapidly and stably.

To gain insight into the effectiveness of the proposed algorithm, Table V illustrates the optimal electricity cost, number of convergence episodes and computational time achieved by the three approaches of RS-ACDRL, DDPG and DQN. As shown, the overall electricity cost of the residential system optimized by the proposed algorithm, two benchmarks of DDPG and DQN are $ 5.079, $ 5.079 and $ 6.741, respectively. The convergence episodes amount of the three approaches are approximately 300, 900 and 100, respectively. Accordingly, the total computational time required to reach convergence are 52 s, 218 s and 27 s, respectively. It can be concluded the optimized total electricity cost of the proposed algorithm is reduced by 24.65% when compared to DQN, indicating that the DQN learns in discrete action space may lead to sub-optimal scheduling strategies while the proposed RS-ACDRL preserves all relevant information concerning the entire continuous action space, and thus is equipped to learn more cost-efficient energy management strategies. Although the final electricity cost of DDPG and RS-ACDRL is the same, the number of convergence episode in DDPG is nearly 600 bigger than that of RS-ACDRL (mainly due to the employment of RS strategy in RS-ACDRL), thereby resulting in the longest computational time. These results demonstrate that beyond obtaining a lower overall electricity cost in regard to the DQN method, the proposed RS-ACDRL algorithm also exhibits a more beneficial computational performance, making it the most valid methodology to address the examined residential energy management problem.

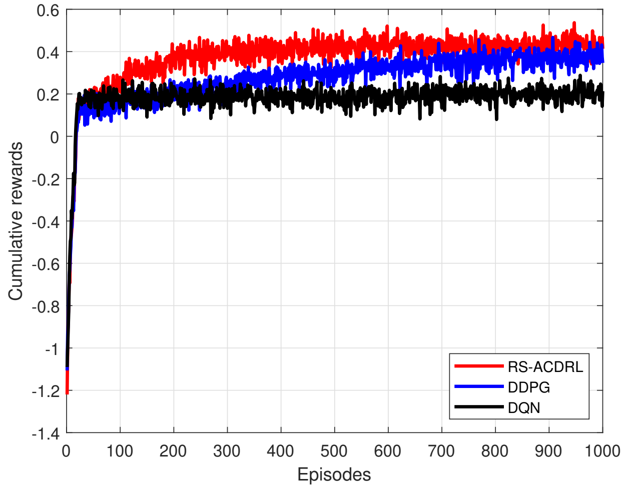


Fig. 7. Performance comparison of different approaches.

TABLE V
Cost and Computational Performance of the Three Methods

Approach	Electricity cost	Number of convergence episodes	Computational time
DQN	6.741 $	100	27 s
DDPG	5.079 $	900	218 s
RS-ACDRL	5.079 $	300	52 s

V. Conclusions and Future Work

In this work, an RS-ACDRL based energy management algorithm for various residential loads is proposed to obtain real-time autonomous control policies, considering the uncertainty and variability of electricity price, energy demand and user behavior. In particular, the proposed RS-ACDRL algorithm leverages the actor-critic architecture by combining it with an RS mechanism. Case studies involving real-world data verify that the RS-ACDRL algorithm gains approximately 38.57% lower electricity costs than that of the case without scheduling, and further promotes the learning process and enhances the policy quality via an RS mechanism compared with the DQN and DDPG. Future work will involve enhancing the generalization capability of the presented algorithm to render it more robust to exogenous uncertainties of power grid conditions (i.e., stochastic renewable energy generation), going beyond the problem formulated in the current work. In addition, the proposed algorithm will also be extended in a multi-agent setting, aiming to provide optimal energy management strategies for multiple smart homes, wherein all participants have learning capabilities and they may cooperate or compete with each other.

References

[1] X. Yan, Y. Ozturk, Z. Hu, and Y. Song, "A review on price-driven residential demand response," *Renewable and Sustainable Energy Reviews*, vol. 96, pp. 411–419, 2018.

[2] L. Park, Y. Jang, S. Cho, and J. Kim, "Residential demand response for renewable energy resources in smart grid systems," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3165–3173, 2017.

[3] B. Parrish, P. Heptonstall, R. Gross, and B. K. Sovacool, "A systematic review of motivations, enablers and barriers for consumer engagement with residential demand response," *Energy Policy*, vol. 138, p. 111221, 2020.

[4] S. Pal and R. Kumar, "Electric vehicle scheduling strategy in residential demand response programs with neighbor connection," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 3, pp. 980–988, 2018.

[5] Z. Zhang, C. Dou, D. Yue, B. Zhang, S. Xu, T. Hayat, and A. Alsaedi, "An event-triggered secondary control strategy with network delay in islanded microgrids," *IEEE Systems Journal*, vol. 13, no. 2, pp. 1851–1860, 2019.

[6] N. Mahdavi and J. H. Braslavsky, "Modelling and control of ensembles of variable-speed air conditioning loads for demand response," *IEEE Transactions on Smart Grid*, vol. 11, no. 5, pp. 4249–4260, 2020.

[7] Z. Zhang, Y. Mishra, D. Yue, C. Dou, B. Zhang, and Y.-C. Tian, "Delay-tolerant predictive power compensation control for photovoltaic voltage regulation," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4545–4554, 2021.

[8] Z. Zhang, C. Dou, D. Yue, and B. Zhang, "Predictive voltage hierarchical controller design for islanded microgrids under limited communication," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 2, pp. 933–945, 2022.

[9] R. Lu, R. Bai, Z. Luo, J. Jiang, M. Sun, and H.-T. Zhang, "Deep reinforcement learning-based demand response for smart facilities energy management," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 8, pp. 8554–8565, 2022.

[10] T. Yang, L. Zhao, W. Li, and A. Y. Zomaya, "Reinforcement learning in sustainable energy and electric systems: A survey," *Annual Reviews in Control*, vol. 49, pp. 145–163, 2020.

[11] R. Lu and S. H. Hong, "Incentive-based demand response for smart grid with reinforcement learning and deep neural network," *Applied Energy*, vol. 236, pp. 937–949, 2019.

[12] Z. Wen, D. O'Neill, and H. Maei, "Optimal demand response using device-based reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2312–2324, 2015.

[13] F. Ruelens, B. J. Claessens, S. Vandael, B. De Schutter, R. Babuška, and R. Belmans, "Residential demand response of thermostatically controlled loads using batch reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2149–2159, 2017.

[14] R. Lu, S. H. Hong, and X. Zhang, "A dynamic pricing demand response algorithm for smart grid: reinforcement learning approach," *Applied Energy*, vol. 220, pp. 220–230, 2018.

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

[16] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg, "On-line building energy optimization using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3698–3708, 2019.

[17] H.-M. Chung, S. Maharjan, Y. Zhang, and F. Eliassen, "Distributed deep reinforcement learning for intelligent load scheduling in residential smart grids," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2752–2763, 2020.

[18] A. Mathew, A. Roy, and J. Mathew, "Intelligent residential energy management system using deep reinforcement learning," *IEEE Systems Journal*, vol. 14, no. 4, pp. 5362–5372, 2020.

[19] S. Lee and D.-H. Choi, "Federated reinforcement learning for energy management of multiple smart homes with distributed energy resources," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 488–497, 2020.

[20] L. Yu, Y. Sun, Z. Xu, C. Shen, D. Yue, T. Jiang, and X. Guan, "Multi-agent deep reinforcement learning for hvac control in commercial buildings," *IEEE Transactions on Smart Grid*, vol. 12, no. 1, pp. 407–419, 2020.

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

[22] S. Bahrami, Y. C. Chen, and V. W. Wong, "Deep reinforcement learning for demand response in distribution networks," *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1496–1506, 2020.

[23] Z. Qin, D. Liu, H. Hua, and J. Cao, "Privacy preserving load control of residential microgrid via deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 12, no. 5, pp. 4079–4089, 2021.

[24] R. R. Afshar, J. Rhuggenaath, Y. Zhang, and U. Kaymak, "A reward shaping approach for reserve price optimization using deep reinforcement learning," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[25] O. Marom and B. Rosman, "Belief reward shaping in reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[26] Y.-S. Peng, K.-F. Tang, H.-T. Lin, and E. Chang, "Refuel: Exploring sparse features in deep reinforcement learning for fast disease diagnosis," *Advances in neural information processing systems*, vol. 31, 2018.

[27] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *ICML*, vol. 99, 1999, pp. 278–287.

[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *Proceedings of the 4th International Conference on Learning Representations*, 2016.

[29] R. Lu, R. Bai, Y. Ding, M. Wei, J. Jiang, M. Sun, F. Xiao, and H.-T. Zhang, "A hybrid deep learning-based online energy management scheme for industrial microgrid," *Applied Energy*, vol. 304, p. 117857, 2021.

[30] Y. Dong, X. Tang, and Y. Yuan, "Principled reward shaping for reinforcement learning via lyapunov stability theory," *Neurocomputing*, vol. 393, pp. 83–90, 2020.

[31] M. Yu and S. H. Hong, "A real-time demand-response algorithm for smart grids: A stackelberg game approach," *IEEE Transactions on Smart Grid*, vol. 7, no. 2, pp. 879–888, 2016.

[32] R. Lu, S. H. Hong, and M. Yu, "Demand response for home energy management using reinforcement learning and artificial neural network," *IEEE Transactions on Smart Grid*, vol. 10, no. 6, pp. 6629–6639, 2019.

[33] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6382–6393.

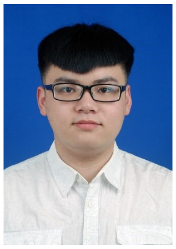
[34] R. Lu, Y.-C. Li, Y. Li, J. Jiang, and Y. Ding, "Multi-agent deep reinforcement learning based demand response for discrete manufacturing systems energy management," *Applied Energy*, vol. 276, p. 115473, 2020.

[35] PJM, "Data Miner 2," http://dataminer2.pjm.com/list, 2021, [Online; accessed July-2021].

[36] H. Li, Z. Wan, and H. He, "Real-time residential demand response," *IEEE Transactions on Smart Grid*, vol. 11, no. 5, pp. 4144–4154, 2020.

Renzhi Lu (Member, IEEE) received the B.E. in electronic information engineering from the School of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan, China, in 2014, and the Ph.D. degree in electronic systems engineering from the Department of Electronic Systems Engineering, Hanyang University, Ansan, South Korea, in 2019.

He is currently an Associate Professor with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China, where he was a Lecturer from 2019 to 2021. His research interests include learning, optimization, and control with applications to smart manufacturing and smart grid.



Zhenyu Jiang received the B.E. degree in rail transit signals and control from the School of Electrical Engineering, Zhengzhou University, Zhengzhou, China, in 2021. He is currently pursuing the M.E. degree in control science and engineering with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China.

His research interests include reinforcement learning algorithm design and its application in sequential decision making.

Huaming Wu (Senior Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, and the Ph.D. degree in computer science from Freie Universität Berlin, Berlin, Germany, in 2015.

He is currently an Associate Professor with the Center for Applied Mathematics, Tianjin University, Tianjin, China. His research interests include wireless networks, mobile edge computing, Internet of Things, and deep learning.

Yuemin Ding (Senior Member, IEEE) received the Ph.D. degree in electronic systems engineering from Hanyang University, Ansan, South Korea, in 2014.

Since 2021, he has been an Associate Professor with the Department of Electrical and Electronic Engineering, University of Navarra, San Sebastian, Spain. From 2019 to 2021, he was a Postdoctoral Fellow with the Department of Energy and Process Engineering, Norwegian University of Science and Technology, Trondheim, Norway. From 2015 to 2019, he was an Associate Professor with the School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China. From 2017 to 2018, he was a Visiting Fellow with the Queensland University of Technology, Brisbane, QLD, Australia. His research interests include Internet of Things, communication networks in smart grid, smart homes/buildings, and smart manufacturing.

Dong Wang (Senior Member, IEEE) received the B.Sc. degree in automation and the M.Eng. degree in control theory and control engineering from the Shenyang University of Technology, Shenyang, China, in 2003 and 2006, respectively, and the Ph.D. degree in control theory and control engineering from the Dalian University of Technology, China, in 2010.

Since 2010, he has been with the Dalian University of Technology, where he is currently a Professor with the School of Control Science and Engineering. His current research interests include multiagent systems, distributed optimization, fault detection, and switched systems. He is an Associate Editor of Information Sciences and Neurocomputing.

Hai-Tao Zhang (Senior Member, IEEE) received the B.E. and Ph.D. degrees in automation and control science and technology from the University of Science and Technology of China, Hefei, China, in 2000 and 2005, respectively.

In 2007, he was a Postdoctoral Researcher with the University of Cambridge, Cambridge, U.K. Since 2005, he has been with the Huazhong University of Science and Technology, Wuhan, China, where he was an Associate Professor from 2005 to 2010, and has been a Full Professor, since 2010. He is also a Cheung Kong Young Scholar. His research interests include swarming intelligence, model predictive control, and unmanned system cooperation control.

Dr. Zhang is/was an Associate Editor for the IEEE TRANSACTION ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-PART II: EXPRESS BRIEFS, ASIAN JOURNAL OF CONTROL, the IEEE Conference on Decision and Control, and the American Control Conference.