Reinforcement Contract Design for Vehicular-Edge Computing Scheduling and Energy Trading via Deep Q-Network with Hybrid Action Space

Li Wang, *Senior Member, IEEE*, Luyang Hou, Sixuan Liu, Zhu Han, *Fellow, IEEE*, and Jie Wu, *Fellow, IEEE*

***Abstract*—The advancements in information and communication technology have led to the emergence of innovative edge computing models that incorporate the computing power of vehicles into the energy sector. Electric vehicles (EVs), functioning as edge computing nodes, offer flexible computing offloading services for charging stations (CS). However, coordinating EV computing and charging should consider the interdependence with CS's specific computing requirements due to information asymmetry. Additionally, it is crucial to consider EV's charging demands and their social distance to computing tasks. In this context, it is natural to view EVs and CSs as self-interested prosumers who prioritize their individual utilities. To address the integration of strategic EV-CS interactions and uncertainties into the joint computing scheduling and energy trading, this paper proposes a parameterized deep Q-network-based reinforcement contract design framework, which employs a hybrid action space to design contracts that facilitate CSs in pairing computing tasks and charging resources with EVs. The objective is to incentivize EV participation and maximize long-term social welfare by incorporating incentive compatibility, individual rationality constraints, and capacity constraints into the contract design. Experimental results demonstrate that the proposed framework surpasses parameterized deep deterministic policy gradient-based and greedy-based contract designs, and achieves near-optimal solutions by solving deterministic optimizations.**

***Index Terms*—Smart charging network, vehicular edge computing, charging scheduling, contract design, parameterized deep Q-network, social welfare.**

I. Introduction

Mobile edge computing revolutionizes the mobile network landscape by bringing computation and storage resources to the network edge, enabling the execution of resource-intensive applications while meeting stringent latency requirements [1], [2]. In this context, mobile vehicles play a crucial role in supporting the sustainability and functionality of sensors and connected devices by leveraging their computational capabilities for complex computing tasks [3], [4]. With the rise of electric vehicles (EVs), there is a growing expectation to integrate them into edge computing networks, fostering synergistic interactions between information processing and energy management [5], [6]. Charging stations (CSs) have the potential to act as edge nodes in these networks, aggregating energy data and performing computing tasks such as analyzing charging behaviours and predicting traffic conditions. Additionally, CSs can establish data exchange and communication channels with EVs, effectively extending computational, communication, and storage capabilities to the edge of the EV ecosystem. As a result, offloading computations to more resourceful edges, such as EVs, becomes a viable solution to enhance the capabilities of CSs [7]. This paradigm shift towards a smart charging network also transforms traditional CSs and EVs into prosumers who manage private information and resources.

Computing offloading involves task partition, offloading decision, and resources allocation [8]. Given the limited computation and charging resources, it becomes crucial to jointly address computation offloading and resource allocation challenges [9]. However, since charging and computing resources are widely distributed among CSs and EVs, the decentralized nature of the environment needs to be considered. Additionally, it is no longer feasible to assume the presence of a central authority with complete knowledge and control over EV behaviours [10]. Instead, a more realistic approach is to deem CSs and EVs as self-interested agents driven by their individual utilities. Consequently, incomplete or unreliable disclosure of EV's private preferences can hinder the effective management of energy and computing resources. The evolving smart charging network, reliant on EV-edge computing, is transforming into a dynamic, distributed market that facilitates contract design. In light of these complexities, the integration of EVs into the charging network entails information gathering and distributed decision-making, making contract design an essential interaction process between CSs and EVs. In such market setting, CSs must provide appropriate incentives to EVs, encouraging their participation in computation offloading through enticing rewards [11]–[13].

A thoughtful contract design for joint computing scheduling and energy trading presents three key challenges. Firstly, both CSs and EVs function as resource requesters and suppliers

This work is partially supported by the National Natural Science Foundation of China (NSFC) under Grant No. U2066201 and No. 62171054, in part by the Fundamental Research Funds for the Central Universities under Grant No. 24820232023YQTD01 and No. 2023RC96, in part by the Double First-Class Interdisciplinary Team Project Funds under Grant No. 2023SYLTD06, in part by the Beijing Municipal Natural Science Foundation under Grant No. L222041, in part by CCF-Baidu Open Fund 2022007, in part by NSF CNS-2107216, CNS-2128368, CMMI-2222810, ECCS-2302469, US Department of Transportation, Toyota and Amazon. *(Corresponding author: Luyang Hou).*

Li Wang, Luyang Hou, Sixuan Liu are with the School of Computer Science (National Pilot Software Engineering School), Beijing University of Posts and Telecommunications, Beijing, 100876, China. (e-mail: {liwang, luyang.hou}@bupt.edu.cn, sixuanliu1@gmail.com).

Zhu Han is with the Department of Electrical and Computer Engineering at the University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul, South Korea, 446-701. (e-mail: hanzhu22@gmail.com).

Jie Wu is with Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA (e-mail: jiewu@temple.edu).

(prosumer) within their respective capacities, posing difficulties in optimizing the scheduling of computation tasks and energy trading. CSs require computing resources while providing charging services, whereas EVs require charging while offering computing power. Secondly, the presence of partially-known asymmetric information and conflicting objectives between EVs and CSs adds complexity to deriving an optimal contract. The efficiency of scheduling computations and charging relies on the information provided by EVs and CSs, with EVs potentially lacking a complete understanding of their own preference profile. Thirdly, dynamic factors and uncertainties that can impact decision-making need to be considered in scheduling computing and charging resources. Evaluating the effectiveness of the current contract in light of future outcomes presents a challenge as well. In summary, an effective contract design should address user discomfort and incorporate human feedback into the control loop [14]. To tackle task offloading and resource allocation issues in mobile-edge computing within dynamic environments, reinforcement learning (RL) offers effective solutions that enable centralized policy training with decentralized execution [15].

This study focuses on addressing the aforementioned challenges by proposing a reinforcement contract design for the coordination of vehicular-edge computing scheduling and energy trading in a smart charging network. The main question this study aims to answer is: *considering the limited capacity and partial information revelation, how can a charging station maximize its long-term rewards by effectively coordinating EVs' computing resources with their charging requirements*? The objective is to maximize the utility and social welfare across both EVs and CSs. To tackle this problem, the study formulates the joint optimization of vehicular-edge computing scheduling and energy trading as a unified problem. A reinforcement contract design framework is implemented, leveraging solutions from reinforcement learning, to facilitate interactions and coordination between EVs and CSs.

The contributions of this study are specified as follows:

- We propose an optimal contract design scheme for the joint vehicular-edge computing scheduling and energy trading problem, where incentive compatible, individual rationality, and system constraints are integrated into a mixed-integer linear program to ensure a voluntarily participation and truthful information revelation. The aim is to achieve efficient computing resource and energy trading, manage information disclosure, and defend against malicious behaviours of selfish participants.
- We incorporate economic constraints and system uncertainties into a sequential, collective contracting process by proposing a parameterized deep Q-network (PA-DQN) based contract design framework. This framework enables CSs to simultaneously coordinate EV's charging and computing capability using a hybrid discrete-continuous action space, such that the resources are efficiently scheduled while maximizing the social welfare.
- We conduct extensive experiments to validate the reinforcement contract design (RCD) framework. The results demonstrate that RCD achieves close-optimal solutions and outperforms other baseline approaches that rely on RL. Notably, our approach efficiently handles uncertainties without requiring information sharing among EVs.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III presents the EV-CS architecture. Section IV introduces the optimal contract design problem. Section V develops a reinforcement learning-based framework for contract design. Section VI carries out an experiment study to validate the proposed methodology. Finally, Section VII draws a conclusion.

II. Related Work

This section reviews the existing literature on the models, strategies, and related approaches with EV-based edge computing and charging scheduling problems, which have been extensively studied recently [13], [16], [17]. The traditional way is to develop mathematical models to optimally schedule the resources. For example, S. Yu *et al.* [17] formulated the energy-efficient task assignment problem by considering the necessary constraints and proposed a Monte Carlo Tree Search based algorithm for the task assignment problem. Although the optimization techniques are still widely used today, they exhibit several limitations in dealing with the distributed and dynamic nature of the joint computing task and charging scheduling problems. Specifically, many optimization-based approaches rely on linearized models and static assumptions of the system, which may not accurately capture the system dynamics and the market behaviours of agents therein. Moreover, solving optimization problems is often computationally expensive and time-consuming in practical scenarios.

Considerable research has been dedicated to developing RL-based approaches for addressing the challenges outlined above. Several notable examples can be found in the literature [16], [18]–[20]. For instance, a model-free deep RL-based distributed algorithm is proposed in [16], where each device determines its offloading decision without knowing the task models and decisions of other devices. The model considered non-divisible and delay-sensitive tasks with an objective of minimizing the expected long-term cost. Y. Zhang *et al.* [18] proposed a distributed scheme based on multi-agent RL, where each cloud center jointly determines the task offloading and resource allocation strategy based on its inference of other cloud center's decisions. A vehicular edge computing-based offloading scheduling problem is investigated in [19], which considers diverse task characteristics, dynamic wireless environment, and frequent handover events caused by vehicle movements. A multi-agent reinforcement mechanism design framework is proposed in [21] for dynamic pricing in a charging network, where the station-user interaction is modelled as a mechanism design problem, and station-station cooperation is captured by the Markov game and solved by multi-agent deep deterministic policy gradient.

Most RL algorithms deal with either discrete or continuous action space solely. For instance, Ref. [22] modelled resource allocation and trajectory design of multiple unmanned aerial vehicles as a decentralized partially observable Markov decision process and solved by multi-agent RL. In terms of more complicated action space, J. Xiong *et al.* [23] extended

NOMENCLATURE

Abbreviations	
CS	Charging Station
EV	Electric Vehicle
IC	Incentive Compatible
IR	Individual Rationality
PA-DDPG	Parameterized Deep Deterministic Policy Gradient
PA-DQN	Parameterized Deep Q-Network
RCD	Reinforcement Contract Design
RL	Reinforcement Learning
SoC	State of Charge
Parameters	
$\boldsymbol{x}$, $\boldsymbol{X}$	Action profile of all users
a, $\mathcal{A}$	Markov decision process: charging station action
i, I, $\mathcal{I}$	Index, number, and set of charging stations
j, J, $\mathcal{J}$	Index, number, and set of electric vehicles
k, K, $\mathcal{K}$	Index, number, and set of types
m, M, $\mathcal{M}$	Index, number, and set of computing task
o, $\mathcal{O}$	Markov decision process: observation
s, $\mathcal{S}$	Markov decision process: system state
t, T, $\mathcal{T}$	Index, number, and set of time step
Index and Set	
α_n	Tendency index of EV to connect to CS
$\beta_{i,k}$	Probability of the CS i's type being θ_k
$\epsilon_{i,t}$	The decayed noise
η	The profit coefficient of CS
γ	Reward discount factor
$\Lambda_{i,k}^{min}, \Lambda_{i,k}^{max}$	The maximal and minimal payment for CS i's k type
μ^*	The optimal deterministic policy
$\mu_i(o_{i,t-1})$	The deterministic policy trained by PA-DDPG
ν_j, ρ_j	The CPU-cycle frequency (cycles/second), and computing density (cycles/byte) of EV j
θ_i^{μ}	The deterministic policy parameter of target critic for charging station i
θ_k, Θ_k	Charging station's type
$\tilde{m}_j, g, f_g$	Mass, gravitational acceleration, ground friction coefficient of EV
c_j^{cpu}, c^c, c^m	The unit cost of computing energy consumption, charging, and mobile consumption
C_i^{in}	Penalty for the unfinished computing task
$d_{i,j}$, $sd_{i,j}$	Physical distance and Social distance between CS i and EV j
$e_{i,j}$	The energy consumption of EV j for moving to CS i
E_j, ΔE_j	The battery capacity and the required energy of EV j
f_g, ϕ	Air resistance, and windward area of EV when moving
H_i, S_i, F_i	The type, size and feature of CS i's computing task
m, n	The number of computing task type, and number of social distance's attributes
P_i^h,P_i^l	The charging power of Level 2 and Level 3 of CS i
P_i^{max}	The maximum output power of CS i
q_i	Number of chargers of CS i
$u_{i,j,k}^{cs}$	Utility function of CS i served by EV j
$u_{i,j}^{ev}$	Utility function of EV j serving CS i
v_j	EV j's moving speed
Decision Variables	
$\lambda_{i,j,k}$	The payment by CS i to EV j for the computing resources
$I_{i,t}$	The number of plugged-in EVs being charged at station i at t
$p_{i,j,t}$	The charging power of EV j in CS i at t
$r_{i,t}$	Hourly revenue (immediate reward) of station i at t
R_i	The accumulated reward of charging station i
$s_{i,j,k}$	The computing resources provided by EV j to CS i's type k
$SoC_{j,t}$	The State-of-Energy of EV j at time t
$t_{i,j}^{chg}, t_{i,j}^{com}$	Charging time and task computing time of EV j
$T_{i,j,k}$	The stay time of EV j at CS i when serving i's k-type task
$z_{i,j,t}$	Binary variable, equals to 1 if EV j accepts CS i's contract

deep Q-network and proposed a PDQN for the discrete-continuous hybrid action space. Furthermore, M. Hausknecht and P. Stone extends the deep deterministic policy gradient (DDPG) algorithm into a parameterized action space by featuring a small set of discrete action types, each of which is parameterized with continuous variables [24]. Ref. [20] proposed a Dirichlet DDPG algorithm to optimize the task partitioning and computational power allocation for computation offloading in a dynamic environment with multiple Internet of Thing devices and edge servers. The goal is to maximize the number of tasks processed before deadlines and minimize the energy cost and service latency. L. Hsieh *et al.* [25] compared three task assignment algorithms for the edge-to-edge horizontal cooperation and the edge-to-cloud vertical cooperation, based on different deep reinforcement learning approaches, value-based, policy-based, and hybrid approaches. Ref. [26] proposed an evolutionary multi-objective RL algorithm to address the trajectory control and task offloading problems of unmanned aerial vehicles. However, the proposed RL algorithms in the literature lack capability to deal with the information asymmetry and selfishness of agents in the joint vehicular-edge computing scheduling and energy trading, so its applicability to market setting should be further enhanced.

Economic theory can capture the assumptions arising from the self-interest nature of agents [27]. C. Li *et al.* [5] developed a contract-based incentive mechanism to motivate parked EVs to contribute their idle on-board resources, and the objective is to maximize the utility of the service provider. Ref. [13] formulated the CSs' profit maximization as a non-collaborative energy contract problem under the smart grid provider's unknown information and common constraints as well as other CSs' contracts. A novel contract-based incentive mechanism is proposed by [28] based on deep RL that combines resource contribution and utilization. An auction mechanism is developed by [29] to deal with multi-charge scheduling of EV users on highway. By considering edge server such as EVs as self-interested agent, H. Zhou *et al.* proposes a reverse auction-based computation offloading and resource allocation mechanism for the mobile cloud-edge computing [9]. However, these approaches are less incorporated with reinforcement learning-based solutions in dealing with dynamics or uncertainties underlying in the joint scheduling of offloading and charging problem in this study.

To sum up, this study improves the existing literature from three aspects: Firstly, unlike works that focus on isolated topics where computing scheduling and energy trading are not simultaneously considered [17], [27], this study considers computing scheduling and energy trading as a joint optimization problem in a decentralized environment. Secondly, unlike contract-based incentive mechanism which motivates EVs to provide computing resources [5], mechanism design-based solutions for mobile edge computing [9] or EV charging scheduling [29], our model captures the dynamics in maximizing long-term benefits using reinforcement learning. Thirdly, unlike other reinforcement learning-based solutions [16], [20], we consider the economic constraints of contract design and

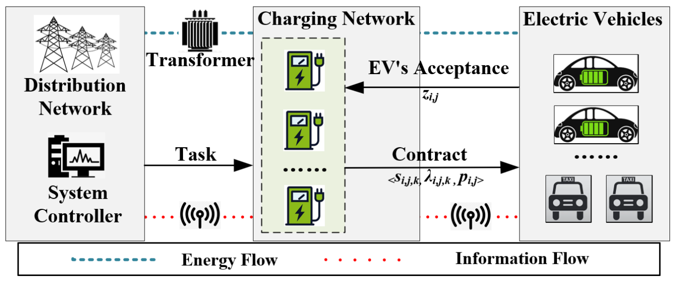


Fig. 1. The topology of the smart EV-CS network under investigation.

achieve utility-maximum scheduling scheme that maximizes the social welfare. To the best of our knowledge, this is the first work to develop a RL-based contract design scheme for the joint energy and computing scheduling.

III. System Architecture

This section provides an overview and explanation of the system model for the investigated smart charging network. The architecture of the electric vehicle-charging station (EV-CS) network is depicted in Fig. 1. It is structured as an Internet of Energy, incorporating information and communication devices that enable interaction and resource exchange between EVs and CSs. Electric vehicles, functioning as intelligent computing devices, can engage with CSs to exchange computing and energy resources. The operational period, denoted as $\mathcal{T}$, is defined as one day, which is further divided into equal-length time slots represented by $t \in \mathcal{T}$. Within this network, there are two types of EVs interested in providing computing services to CSs while simultaneously recharging their batteries. These EVs are classified as individual EVs and commercial EVs, such as taxis, and they differ in terms of battery capacity and computing capability. Each CS randomly generates different types of computing tasks and publishes information specifying the computing resources required from EVs. In particular, a CS accomplishes computing tasks by coordinating one or several EVs that possess the necessary computation capabilities for the specific task. CSs are primarily concerned with long-term objectives, such as maximizing cumulative revenue over the course of a day. To achieve this, CSs offer attractive contracts that outline the required resources and corresponding payments to EVs, thereby incentivizing them to accept and complete the tasks.

The formulation of a well-designed contract is essential for eliciting and integrating the private preferences of EVs regarding coordinated charging and computing. In a decentralized setting, EVs are not under the control of any CS; instead, both EVs and CSs prioritize their individual benefits. The main objective is to identify a successful contract design, from the perspective of charging stations, that can efficiently facilitate in-vehicle computation tasks while maximizing the benefits for both CSs and EVs.

A. Charging Network Model

In the smart charging network, computing tasks originate from different stakeholders, each with their own unique characteristics. These tasks are then transmitted to the charging network for processing. The charging network controller assumes a crucial role in consolidating essential information regarding CSs, including their size, location, and availability status. Using this comprehensive information, the controller effectively assigns tasks to CSs at different times. To facilitate this allocation process, the charging network is equipped with a total of I charging stations, denoted by the set $\mathcal{I}$. There are M types of tasks which can be load forecasting, charging demand prediction, EV charging habit analysis, traffic data analysis, image processing, among many others. A task $m \in \mathcal{M}$ is randomly assigned to a CS $i \in \mathcal{I}$ at a time, characterized by a 3-tuple: $\langle H_{i,m}, S_{i,m}, F_{i,m} \rangle$, where $H_{i,m}$ is the task type, $S_{i,m}$ is the task size that determines the required workload, and $F_{i,m}$ is the feature of this task. To ensure reasonable computing times and avoid excessive delays, it is assumed that the sizes of different tasks are evenly distributed. The tasks assigned to a CS are divisible and can be partitioned into multiple parts of varying sizes. In addition, it is assumed that the number of CPU cycles required for computing each input bit remains constant.

Charging stations can earn by completing the computation tasks given by the charging network and provide charging services to EVs. The maximum output power P_i^{max} of CS i is limited by the transformer, and the charging power of EV j at CS i is denoted by $p_{i,j}$. CS i owns q_i chargers with two charging modes, i.e., level-3 DC fast charge and level-2 AC slow charge with power P_i^h and P_i^l, respectively. In order to overcome the limitations of EVs' limited computing capability for large computing tasks, a solution is to decompose the task and assign it to a group of EVs. In return for completing the sub-tasks, EVs are rewarded with a share of the profits as their payment. This involves establishing a one-to-many relationship between the CS and EVs, where each EV can choose only one CS while a CS can invite multiple EVs to undertake computing tasks.

B. Electric Vehicle Model

Consider J EVs which are classified into two types: individual EV and commercial EV with set represented by $\mathcal{J}$, and EV $j \in \mathcal{J}$'s with the battery capacity of E_j. EVs can provide low-cost and flexible computing services for nearby charging station nodes, and EVs can gain by providing computing services to CSs and have preferences on the charging power provided by CS based on its current state-of-charge (SoC) $SoC_{j,t}$. The computing capability of EV j can be represented by ν_j which denotes the CPU-cycle frequency.

The physical distance $d_{i,j}$ between CS i and EV j is the Euclidean distance in a urban area. The average moving speed of EV j is represented by v_j, and its energy consumption due to overcome friction resistance and air resistance for moving to CS i is defined by [30] as follows:

$$e_{i,j} = \frac{d_{i,j}}{v_j} \cdot \left\{ \frac{\tilde{m}_j \cdot g \cdot f_g \cdot v_j}{3,600} + \frac{f_a \cdot \phi \cdot v_j^3}{76,140} \right\}, \tag{1}$$

where $\tilde{m}_j, g, f_g, v_j, f_a$, and ϕ represent the mass of EV i, gravitational acceleration, ground friction coefficient, speed of EV i, air resistance, and windward area, respectively.

EVs should ensure they have enough energy to arrive at CS. The stay time $T_{i,j}$ of EV j in CS i takes the largest part of the charging time $t_{i,j}^{ch}$ and the task computing time $t_{i,j}^{co}$:

$$T_{i,j} = max\left\{t_{i,j}^{chg}, t_{i,j}^{com}\right\} = max\left\{\frac{\Delta E_j}{p_{i,j}}, \frac{s_{i,j,k} \cdot \rho_j}{\nu_j}\right\}, \quad (2)$$

where $t_{i,j}^{chg} = \Delta E_j / p_{i,j}$, ΔE_j is the energy that EV j desires to charge, $p_{i,j}$ is the charging power that CS i provides for EV j in the contract. The computing time $t_{i,j}^{ch}$ of EV j serving k-type CS i is $t_{i,j}^{ch} = \frac{s_{i,j,k} \cdot \rho_j}{\nu_j}$, where $s_{i,j,k}$ is the computing task allocated to EV j, ρ_j denotes the computing density (in cycles/byte), ν_j denotes its CPU-cycle frequency (in cycles/second), which measures EV's computing ability.

C. EV-CS Interaction

The architecture of the EV-CS system comprises both social interactions and physical interactions involving EVs and CSs. Physical interactions refer to the need for EVs to recharge when their power levels are insufficient. In this context, when offering a contract to EVs, CSs must consider the availability of idle chargers at their stations. On the other hand, social interactions involve a many-to-many contract framework that connects CSs with EVs in an energy Internet. Within this framework, EVs provide computing resources to assist in completing smart charging station tasks, while CSs provide charging services to EVs. Social interactions are more complex than physical interactions due to the continuous closed-loop feedback between agents and the presence of uncertainties [31]. The interaction between CSs and EVs requires intricate decision-making processes, compelling CSs to generate optimal contracts and EVs to carefully consider whether to accept the contract offered by a CS.

In this study, we propose a social interaction-based definition of the contract design problem as follows: *"A dynamic sequence of actions that mutually consider the actions and reactions through a decision-making and information exchange process between EVs and CSs to maximize economic benefits."* In this context, both EVs and CSs are considered rational and self-interested agents that strive to maximize their own utility. However, there exists information asymmetry during the trading process, where each side is unaware of the other's type or characteristics. This lack of information poses challenges in the trading of computing tasks and energy in this two-sided market. Factors such as the distance between EVs and CSs, transformer capacity limitations, and space restrictions within the charging network further constrain the trading process. Given these constraints, it becomes crucial to design contracts thoughtfully in order to capture EVs' preferences and appropriately match them with suitable tasks. By doing so, we aim to maximize long-term social welfare by efficiently utilizing available resources and optimizing interactions between EVs and CSs. This approach seeks to strike a balance between the selfishness of individual agents and the overall societal benefit.

Despite the physical distance $d_{i,j}$ between CS i and EV j, their social distance $sd_{i,j}^{m}$ measures the degree of acceptance or rejection of contract offered by CS i for completing a specific task m. Correspondingly, the *physical distance* and *social distance* should be considered in assigning computing task to EVs with proper payment. Therefore, we define the social distance of our model based on [32] as:

$$sd_{i,j}^{m} = \sum_{n=1}^{N} \zeta_n \cdot sd_{i,j,n}^{m}, \quad (3)$$

where ζ_n is a normalized weight factor measuring the importance that each social attribute has in the process of formation of connections. Each attribute represents a distinctive social feature of EV (individual or commercial) and computing task, such as task type $H_{i,m}$, profession, geographic location, and so on. It is possible to define a set of social distances corresponding to each attribute, and $sd_{i,j,n}^{m} \in [0, \infty)$, $n = 1, ..., N$. The social distance will influence EV's willingness or probability to accept a specific computing task.

Understanding the underlying rules governing the formation of social networks is a challenging task due to the multitude of factors that influence human interactions. Individuals sharing the same interests, common places, similar ideas or akin objectives, for example, tend to form acquaintances [32]. In the investigated clusters of EVs, both individual and commercial EVs exhibit similar social distances when it comes to their association with specific types of computing tasks.

IV. Contract Design For Computing Scheduling and Energy Trading

In this section, we present a mathematical model for the contract design of joint computation scheduling and energy trading between CSs and EVs. To begin with, we establish the contract design problem by introducing the necessary terminology and concepts.

A. EV-CS Contract

Definition 1 (Contract design [33])**.** *A general form of contracting problem considers a principal (CS in our model) who interact with an agent (EV). They play the following game:*

- *The CSs offers a contract: $< s, \lambda, p >$, s: computing resource, λ: payment for task, p: charging power;*
- *The EV publicly accepts or rejects this contract. Rejection ends the game;*
- *If the EV accepts the contract, he privately chooses an action $(F; c)$, where F is a probability distribution and c is a cost. If the EV accepts the contract, the utility for EV and CS are calculated.*

We consider a three-party contract design structure involving the charging network, individual EVs, and commercial EVs. The optimal contract design determines the division and allocation of task resources and the pricing policy based on its distance to EVs, task type, and the available chargers provided for EVs. The CSs aim to offer contracts to individual EVs for small tasks, while aggregating multiple commercial EVs to collaborate on more complex tasks, often accompanied by a group price. This approach efficiently utilizes edge computing resources while meeting the charging demands of EVs.

In the proposed model, CSs are considered self-interested agents motivated to acquire larger computing resources while minimizing their payments. To prevent EVs from gaining knowledge of the true value of computing resources, CSs strategically conceal their demands for computation resources. Based on contract theory principles, CSs are classified into multiple types based on their specific computing resource requirements. Specifically, let CS's possible types be θ_k, and $\theta_k \in \{\theta_1, \cdots, \theta_K\}$, which satisfies $\theta_1 < \cdots < \theta_K$. The CS with a higher type requires more computing resources and needs to pay more rewards. Assuming that the types of CSs obey a uniform distribution. In general, a type θ encapsulates all the information possessed by agents that is not publicly known, which includes their beliefs about others' utilities, and beliefs about their own utilities [10]. In our model, a type of a CS is the social distance of the current task and an EV. To cope with the information asymmetry on EVs' computing resource and CSs' requirements, each CS owns m types of contracts, and EV maximizes utility by selecting the contract item designed for its type. Finally, according to the contract items selected by EV, it will consume corresponding resources to provide services, and CS will pay corresponding remuneration.

The contract built for CS i's k-th task is denoted by $(s_{i,j,k}, \lambda_{i,j,k}, p_{i,j}), \theta_k \in \Theta_k$, where $s_{i,j,k}$ is the amount of computing resources provided by EV j to CS i's type k and $\lambda_{i,j,k}$ is the payment by CS i to EV j for the computing resources, $p_{i,j}$ is the charging power for EV, $p_{i,j} \in \{P_i^h, P_i^l\}$. The payment model is introduced as service level agreement between the individual EVs and commercial EVs, which is dependent on the task type, required workload and CS's type, and limited by $\lambda_{i,j,k} \in [\Lambda_{i,k}^{min}, \Lambda_{i,k}^{max}]$. The CS must give EVs the incentive to accept the contract, which is guaranteed by the incentive compatible (IC) constraint, and anticipating this effort, the CS must be willing to offer the contract guaranteed by the individual rationality (IR) constraint, which will be explained in details in later subsection.

B. Utility Function

1) Utility of electric vehicles: EVs choose task assigned by CS i at time t based on its SoC status, availability, the social distance with this CS, and the incentive signal from CS. To quantify EV acceptance probability, it is necessary to understand the psychological impact of the internal economic stimuli. Specifically, the decision is made on EV's utility which is the difference of the benefits for providing computing services and the cost that includes the calculation cost for computing task, the charging cost, and the energy cost for moving. The utility function[1] of EV j serving CS i's kth type is defined as follows:

$$u_{i,j,k}^{ev} = \sum_{k \in \mathcal{K}} \beta_{i,k} \left[\lambda_{i,j,k} - c_j^{cpu} w \nu_j (s_{i,j,k})^2 - c^c t_{i,j}^{chg} - c^m e_{i,j} \right], \tag{4}$$

where $\beta_{i,k}$ denotes the probability of CS's θ_k type, c_j^{cpu} is EV j's CPU unit cost of energy consumption for computing, w is constant power coefficient based on CPU chip architecture of EV j, $s_{i,j,k}$ is the computing resources allocated by CS i, c^c and c^m denotes the unit cost of charging time (\$/min) and mobile energy consumption, respectively.

2) Utility of charging station: The utility of CS i is calculated by the difference between profits for completing task $\eta \log\left(1 + \frac{\theta_k}{T_{i,j,k}}\right)$ and providing charging services $\sum_{j \in \mathcal{J}} z_{i,j} c^c t_{i,j}^{chg}$ minus payment $\lambda_{i,j,k}$ to EV j, and $T_{i,j,k}$ is the stay time of EV j at CS i when serving i's k-type computing task. In real-world scenarios, it is always hard to precisely model utility function, whereas different functions do not significantly affect the problem formulation and solutions. Due to this, the profit (the first item in (5)) for the computing task of our model adopts a $Logarithm$ function of the ratio of θ_k-type and the computation delay, which indicates that CSs have a decreasing marginal value and its revenue will not increase indefinitely, similar use cases can be found in [34], [35]. The utility of CS i served by EV j is defined by:

$$u_{i,j,k}^{cs} = \eta \log\left(1 + \frac{\theta_k}{T_{i,j,k}}\right) + \sum_{j \in \mathcal{J}} z_{i,j} c^c t_{i,j}^{chg} - \lambda_{i,j,k} - C_i^{in}, \tag{5}$$

where η denotes the profit coefficient. This utility function is twice continuously differentiable, concave, and has range $\mathbb{R}^+$. The C_i^{in} is the penalty for CS i as it cannot finish the computing task $S_{i,m}$. If a CS does not engage enough EVs or computing task timeout caused by irrational allocation within one time step, then CS has to pay the incompletion costs with unit price c^{in} for the unfinished part, which is expressed by $C_i^{in} = c^{in}[S_{i,m} - \sum_{j \in \mathcal{J}} z_{i,j,t} s_{i,j,k}]^+$, and $[h]^+ = max\{0, h\}$ according to [21].

C. The Optimal contract design problem

Building upon the previous section's explanation, an EV can choose a maximum of one CS, whereas a CS can cater to multiple EVs simultaneously. Consequently, this scenario presents a many-to-one selection problem. Assuming the EV engages in the optimal contract, the CS's optimal contract entails solving a constrained maximization problem. The optimal contract design problem for the joint vehicular-edge computing scheduling and energy trading is formulated as a mixed-integer linear program, where IC and IR, and system constraints such as non-overlapping, capacity constraints are incorporated into the model outlined below to ensure voluntarily participation and truthful information revelation. This model determines which EVs should be assigned contracts, which computing tasks should be allocated to these EVs with corresponding payment, and the charging power for EV charging requests. Let $z_{i,j,t} = 1$ if EV j is assigned to CS i at time t, otherwise $z_{i,j,t} = 0$. The objective of contract design is to maximize the social welfare, i.e., the utilities of CSs and EVs. Specifically, the optimal contract design problem solves φ_{opt}^*:

$$\varphi_{opt}^* : \quad \max_{\mathbf{z},\mathbf{s},\lambda,\mathbf{p}} \sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{j=1}^{J} z_{i,j,t}(u_{i,j,k,t}^{cs} + u_{i,j,k,t}^{ev}), \tag{6}$$

s.t.

$$0 \le u_{i,j,k,t}^{ev}, \quad \forall k \in \mathcal{K}, i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{7a}$$

[1]For simplicity, we omit the subscript t in the utility function and the following optimization model.

$$u^{ev}_{i,j,k',t} \leq u^{ev}_{i,j,k,t}, \quad \forall k' \neq k, i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{7b}$$

$$\sum_{j \in \mathcal{J}} z_{i,j,t} p_{i,j,t} \leq P^{max}_i, \quad \forall i \in \mathcal{I}, t \in \mathcal{T} \tag{7c}$$

$$S_{i,m,t} \leq \sum_{j \in \mathcal{J}} z_{i,j,t} s_{i,j,k,t}, \quad \forall i \in \mathcal{I}, t \in \mathcal{T} \tag{7d}$$

$$\sum_{j \in \mathcal{J}} z_{i,j,t} \leq q_{i,t}, \quad \forall i \in \mathcal{I}, t \in \mathcal{T} \tag{7e}$$

$$\sum_{i \in \mathcal{I}} z_{i,j,t} \leq 1, \quad \forall j \in \mathcal{J}, t \in \mathcal{T} \tag{7f}$$

$$z_{i,j,t} e_{i,j} \leq SoC_{j,t}, \quad \forall j \in \mathcal{J}, t \in \mathcal{T} \tag{7g}$$

$$\Lambda^{min}_{i,k} \leq \lambda_{i,j,k,t} \leq \Lambda^{max}_{i,k}, \ \forall i \in \mathcal{I}, j \in \mathcal{J}, k \in \mathcal{K}, t \in \mathcal{T} \tag{7h}$$

$$p_{i,j,t} \in \{P^h_i, P^l_i\}, \quad \forall i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{7i}$$

$$z_{i,j,t} \in \{0, 1\}, \quad \forall i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, \tag{7j}$$

where the objective function (6) maximizes the social welfare which consists of the utilities of EVs and CSs. (7a) is the IR constraint, which guarantees that the participation of EVs can benefit from the trading; (7b) is the IC constraint, which indicates that each EV can obtain the maximum utility when it selects the contract item that suits CS's type. Constraints (7c) indicate that the total charging power of all charged EVs in CS i is limited by P^{max}_i. Constraints (7d) denote that CS i's computing task should acquire enough resources from EVs who accept the contract. Constraints (7e) restrict that the number of all charging EVs should not exceed the idle number of chargers $q_{i,t}$ of CS i at time t. Constraints (7f) denote that an EV can select at most one CS for completing the task. Constraints (7g) ensure that an EV has enough energy to arrive at the destination. Constraints (7h), (7i), and (7j) define the range of $\lambda_{i,j,k}$, $z_{i,j,t}$, and $p_{i,j,t}$.

D. EV acceptance

Once the optimal contract is derived and sent to EVs, EVs need to decide whether to accept it. The acceptance probability of EVs depends on two key factors: the EV's preference and the social distance. We expect that the probability of acquaintance decreases with larger social distance. Following the connection probability proposed in [32], we define the acceptance probability of EV j on CS i's contract as:

$$Prb_n(i, j) = \frac{1}{1 + [\lambda_{i,j,k} \cdot sd^m_{i,j}]^{\alpha_n}}, \tag{8}$$

where $\alpha_n > 1$ measures the tendency of EVs to connect to a CS. The probability of accepting a contract also depends on the payment $\lambda_{i,j,k}$ offered by CS i.

V. Reinforcement Contract Design Framework

This section develops a reinforcement contract design (RCD) framework as depicted in Fig. 2, where the CS-EV interaction is modelled as a Markov decision process (MDP) and trained by a parameterized deep Q-network (PA-DQN)-based algorithm with hybrid actions [23]. Specifically, the environment describes the charging network between EVs and CSs. The contract design of each CS (learning agent) is acquired by the task allocator and will be sent to ambient EVs. Each CS then trains its own policy, receives and processes the data from the environment and obtains the observation vector to calculate the reward. The policy of each CS is to take an action (i.e., assign contracts to EVs) based on its observation from the last time slot. To train and acquire the optimal policy, we represent each policy as neural networks whose parameters are tuned based on PA-DQN, and model the contract $\{s, \lambda, p\}$ as a discrete-continuous hybrid action. In this PA-DQN framework, the action is defined by the hierarchical structure. The idea is that a high-level action $z_{i,j}$ (CS i's contract allocated to EV j) is first chosen from a discrete set Z; upon choosing Z, a low level parameters $\{s_{i,j,k}, \lambda_{i,j,k}, p_{i,j}\}$ is then chosen associated with the high level action $z_{i,j}$.

A. MDP formulation

The CS-EV interaction is formulated as an Markov decision process [36], characterized by a 5-tuple $\langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$, where $\mathcal{S}$ denotes the global state space of the environment and $s_t = \{o_1, ..., o_I\} \in \mathcal{S}$ denotes the observation of each CS. $\mathcal{A}$ is the action space $a_t \in \mathcal{A}$. The function $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the Markov transition probability distribution, where $p(s_{t+1}|s_t, a_t)$ represents the transition probability from s_t to s_{t+1} after a_t is taken. The function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the expected rewards for state–action pairs, where $r(s_t, a_t)$ is the immediate reward received when a_t is taken at s_t. Let R_t denote the discounted sum of rewards from the state s_t, then $R_t = \sum_{t \in \mathcal{T}} \gamma^t r(s_t, a_t)$, where $\gamma \in (0, 1]$ is the discount factor. A deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ maps each state to a particular action in $\mathcal{A}$. In the case of RCD, a CS chooses a hybrid action from the given set in the current state, and EVs respond strategically based on the social distance. At the end of t, CS receives an immediate reward associated with the outcome, the system then progresses to $t + 1$ with all information of CSs and EVs updated accordingly.

In what follows, we will elaborate on these components.

1) Agent: We model each CS $i \in \mathcal{I}$ as a learning agent who calls for the neighbor EVs to implement a series of computing tasks by offering contracts to them throughout the day.

2) Observation: The observation $o_{i,t}$ of CS i at t is defined as a 6-tuple: $\langle \mathbf{D}_{l,t}, \mathbf{SoC}_{l,t}, \mathbf{N}_{l,t}, H_{i,t}, S_{i,t}, F_{i,t} \rangle$, where l represents the group of EVs that a CS observes, $\mathbf{D}_{l,t}$, $\mathbf{SoC}_{l,t}$ and $\mathbf{N}_{l,t}$ correspond to the computing resource, SoC level, the type of these EVs of CS i, respectively. The information specific to CS includes $H_{i,t}$ (type of computing task), $S_{i,t}$ (size of computing task), and $F_{i,t}$ (features of computing task). Each CS obtains this observation at the beginning of time t.

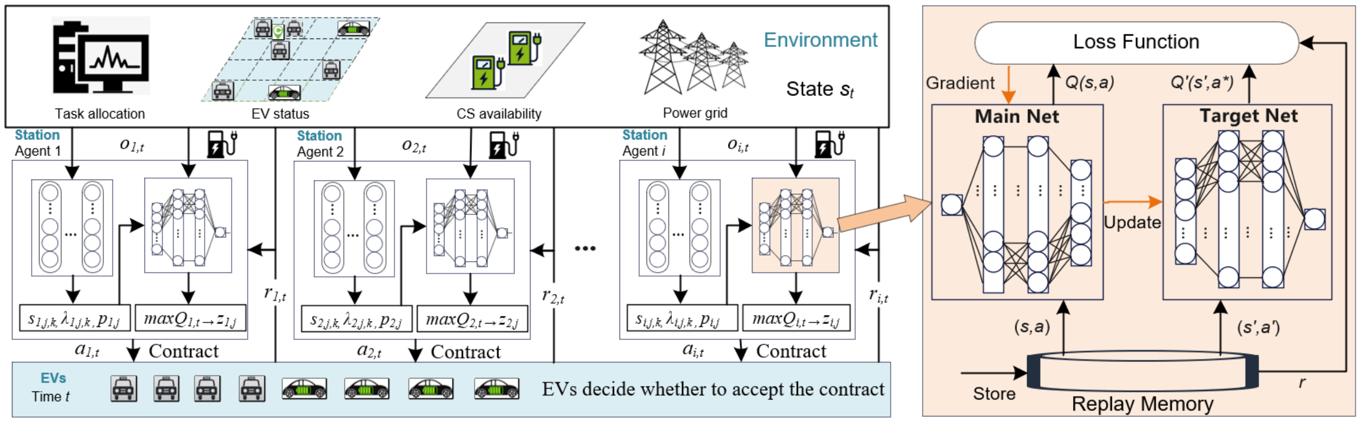


Fig. 2. The PA-DQN based reinforcement contract design framework with discrete-continuous action space.

3) Action: The action $a_{i,t} = \{z_{i,j,t}, s_{i,j,k}, \lambda_{i,j,t}, p_{i,j,t}\}_{j \in \mathcal{J}}$ is taken by CS i at time t to design a contract based on the state $s_{i,t}$. The EV selection $z_{i,j,t}$ corresponds to a discrete action which is encoded to represent a bunch of EVs $j \in \mathcal{J}$ that a CS sent contract to. $\{s_{i,j,k}\}_{j \in \mathcal{J}}$ is the computing task allocated to these EVs and constraint (7d) holds. And $\{\lambda_{i,j,k}\}_{j \in \mathcal{J}}$ is the payment of CS i to EVs, $\lambda_i \in \Lambda_i = [\Lambda_i^{min}, \Lambda_i^{max}]$, where all CSs share the same payment set, i.e., $\Lambda_1 = ... = \Lambda_I = \Lambda$. The charging power $\{p_{i,j,t}\}_{j \in \mathcal{J}}$ provided for these EVs are chosen from level-2 P_i^h and level-3 P_i^l option.

4) Reward: The reward (social welfare) $r_{i,t}$ of agent i includes EV utility $\sum u_{i,j,t}^{ev}$ and CS utility $\sum u_{i,j,t}^{cs}$, i.e., $r_{i,t} = \sum_{j \in \mathcal{J}} z_{i,j,t}(u_{i,j,k,t}^{ev} + u_{i,j,k,t}^{cs})$. The quality of contract will influence the computing task completion while the unfinished task or insufficient acceptance on the contracts might cause potential profit loss to CSs in the long run. Let $R_i^\pi = \sum_{t=1}^{T} \gamma^{t-1} r_{i,t}$ denote the accumulated discounted rewards of CS i under policy π. RCD aims to find an optimal collective policy $\boldsymbol{\pi}^*$ that maximizes the sum of the expected $\mathbb{E}[R_i^{\boldsymbol{\pi}}]_{i \in \mathcal{I}}$ of all CSs. Therefore, the objective function of station i is formulated as $J(\theta_i^\pi) = \mathbb{E}[R_i^{\boldsymbol{\pi}}]$, the key is to train the optimal parameter vector θ_k^* for its policy π_i^*.

The actions taken by the learning agent during training satisfy the constraints introduced in the mathematical formulation φ_{opt}^*. Specifically, the size of computing task $s_{i,j,k}$ to each EV satisfies $S_{i,m} \leq \sum_{j \in \mathcal{J}} z_{i,j,t} s_{i,j,k}$ (constraint (7d)). The size of actor $z_{i,j,t}$ equals to the number of chargers $q_{i,t}$, which corresponds with constraint (7e). In addition, the IR in (7a) and IC in (7b) constraints also hold by adding penalty to the reward function and by EV acceptance based on the social distance, respectively. EVs will stay until its SoC reaches to the expected level, and EV with unfinished charging will be automatically put into this CS's contract list for the next time step, which corresponds with constraint (7g). By refining the range of the hybrid action and defining reward function, we can transform the optimization problem φ_{opt}^* into the MDP considering long-term objectives.

B. System state transition

EVs are faced with the decision of accepting or rejecting the contract offered by CS i at a given time t, taking into account their SoC level and utility. When EVs enter into contracts with CSs, each CS calculates its utility, updates its state, and proceeds to the next time step. The system's overall status is also updated to reflect the transitions occurring between consecutive time slots. For agent i, the observation transition encompasses various elements. These include the completion of computing tasks, where unfinished tasks are carried over to the next time step, updates to the EV's SoC level, changes in the EV's position due to movement, and whether the EV remains at the CS to continue charging in the subsequent time step. The state transformation of agent i after taking $a_{i,t}$ is defined as $(o_{i,t}, a_{i,t}, o_{i,t+1})$, where $o_{i,t}$ is the current observation, and $o_{i,t+1}$ is for time $t+1$.

1) EV SoC level: The SoC level of EVs are updated by:

$$SoC_{i,t+1} = SoC_{i,t} + z_{i,j,t}(p_{i,j,t} - e_{i,j} - w\nu_j(s_{i,j,k})^2), \quad (9)$$

where $w\nu_j(s_{i,j,k})^2$ is the energy consumption for completing the computing task.

2) CS task completion: The stay time of EV j depends on the charging time and computing time, and j will stay at this CS if it cannot accomplish the charging or computation task. Thus, we have $z_{i,j,t} = z_{i,j,t+1}, \ t = 1, ..., T_{i,j}$, if $T_{i,j} > |t|$.

3) Computing task completion status: If a CS fails to engage enough computing power (the number of accepted EVs) to complete a task at time t, it has to postpone the unfinished task to time $t+1$, which may degrade the profits and affect the successive contract offered to EVs.

4) CS availability: The charging station should update the availability, i.e., the idle number of chargers $q_{i,t}$ at time t based on the task completion status.

C. DQN-based solution for hybrid action space

We develop an independent multi-agent reinforcement learning using PA-DQN [23] to train the optimal policy $\boldsymbol{\pi}^*$ of RCD. PA-DQN extends DQN to address hybrid action space rather than approximation or relaxation. Specifically, each agent trains its own policy and actor-parameter for discrete and continuous action. DQN is employed to capture the high-dimensional observations and generate plausible policies in order to maximize long-term rewards. The pseudo-code of the PA-DQN-based reinforcement contract design framework is depicted in Algorithm 1.

Algorithm 1 Contract design training by PA-DQN

```
1: Exploration parameter $\epsilon$, CS $i$'s $Q_i(\cdot)$ and $\mu_i(\cdot)$ with parameters $\theta_i^Q$ and $\theta_i^\mu$ with He initialization;
2: Initialize the target $Q_i'(\cdot)$ and $\mu_i'(\cdot)$ with parameters $\theta_i^{Q'} \leftarrow \theta_i^Q$, $\theta_i^{\mu'} \leftarrow \theta_i^\mu$, and the replay buffer $\mathcal{D}$;
3: for $episode$ = 1 to $2,000$ do
4:     for $t$ = 1 to $T$ do
5:         for Agent $i$ = 1 to $I$ do
6:             Each agent obtains observation $\boldsymbol{o}_{i,1}$ independently;
7:             Agent $i$ select a random action $a_{i,t}$ with probability $\epsilon$; and selects $a_{i,t} = (z, x_z)$, such that $z = \max_z Q^*(s_{i,t}, z, x_z; \omega_t)$ with probability $1-\epsilon$;
8:             Take action $a_{i,t}$, observe reward $r_{i,t}$, and the next state $s_{t+1}$;
9:             Store the transition $(\boldsymbol{s}_{i,t}, \boldsymbol{a}_{i,t}, \boldsymbol{r}_{i,t}, \boldsymbol{s}_{i,t+1})$ in $\mathcal{D}_i$;
10:            Sample a random minibatch of $N_B$ transitions from $\mathcal{D}_i$;
11:            Update the Q net $Q_i(\cdot)$ by minimizing Eq. (16);
12:            Update the policy $\mu_i(\cdot)$ based on the Eq. (17);
13:            Soft-update the target $Q_i'(\cdot)$, $\mu_i'(\cdot)$:
                   $\theta_i^{Q'} \leftarrow \tau \cdot \theta_i^Q + (1-\tau) \cdot \theta_i^{Q'}$;
                   $\theta_i^{\mu'} \leftarrow \tau \cdot \theta_i^\mu + (1-\tau) \cdot \theta_i^{\mu'}$.
14:        end for
15:    end for
16: end for
```

We define the discrete-continuous action and its space as:

$$a_t = \{z_t, x_{z_t}\}, \quad a_t \in \mathcal{A}_t, \tag{10}$$

$$\mathcal{A} = \{(z, x_z) \mid x_z \in \mathcal{X}_z, \forall z \in [Z]\}, \tag{11}$$

which indicates that we first choose a discrete action z from the set $[Z]$; upon choosing z, we further choose its associated continuous action $x_z = \{s_{i,j,k}, \lambda_{i,j}, p_{i,j}\}_{j \in \mathcal{J}}$, and $x_z \in \mathcal{X}_z$ is associated with the z-th high level action, where $\mathcal{X}_z$ is a continuous set for all $z \in [Z]$. For $a \in \mathcal{A}$, we denote the action value function by $Q(s, a) = Q(s, z, x_z)$, where $s \in \mathcal{S}$, $z \in \mathcal{Z}$, and $x_z \in \mathcal{X}_z$. Let z_t be the discrete action selected at time t and let x_{z_t} be the associated continuous parameter. Then the Bellman equation becomes:

$$\begin{aligned} Q(s_t, z_t, x_{z_t}) = & \underset{r_t, s_{t+1}}{\mathbb{E}} [r_t + \\ & \gamma \max_{z \in [Z]} \sup_{x_z \in \mathcal{X}_z} Q(s_{t+1}, z, x_z) \mid s_t = s, a_t = (z_t, x_{z_t})], \end{aligned} \tag{12}$$

where on the right-hand side of (12), we denote $x_z^* = \operatorname{argsup}_{x_z \in \mathcal{X}_z} Q(s_{t+1}, z, x_z)$ for each $z \in [Z]$, and then take the largest $Q(s_t + 1, z, x_z^*)$. When the function Q is fixed, for any $s \in S$ and $z \in [Z]$, we can view $\operatorname{argsup}_{x_z \in \mathcal{X}_z} Q(s, z, x_z)$ as a function x_z^Q: $\mathcal{S} \to \mathcal{X}_z$.

Then, the Bellman function can be rewritten as:

$$\begin{aligned} & Q(s_t, z_t, x_{z_t}) = \\ & \underset{r_t, s_{t+1}}{\mathbb{E}} \left[r_t + \gamma \max_{z \in [Z]} Q\left(s_{t+1}, z, x_z^Q(s_{t+1})\right) \mid s_t = s \right]. \end{aligned} \tag{13}$$

Similar to deep Q-network, we use a deep neural network $Q(s, z, x_z; \omega)$ to approximate $Q(s, z, x_z)$, where ω is network weight. For $Q(s, z, x_z; \omega)$, we approximate $x_z^Q(s)$ with a deterministic policy network $x_z(\cdot; \theta) : \mathcal{S} \to \mathcal{X}_z$, where θ denotes the network weights of the policy network[2]. In other words, when ω is fixed, we try to find a θ such that

$$Q(s, z, x_z(s; \theta); \omega) \approx \sup_{x_z \in \mathcal{X}_z} Q(s, z, x_z; \omega), \ \forall z \in [Z]. \tag{14}$$

Following the approach of DQN, we estimate the weights ω by minimizing the mean-squared Bellman error through gradient descent. Let ω_t and θ_t represent the weights of the value network and the deterministic policy network at time t, respectively, and the target y_t is defined as follows:

$$y_t = r_t + \gamma \max_{z \in [Z]} Q(s_t, z, x_z(s_t, \theta_t); \omega_t). \tag{15}$$

To optimize the parameter θ while keeping ω fixed and maximizing $Q(s, z, x_z(s; \theta); \omega)$, we employ the least squares loss function. The loss function for θ is formulated as follows:

$$\ell_t^Q(\omega) = \frac{1}{2} \left[Q(s_t, z_t, x_{z_t}; \omega) - y_t \right]^2, \tag{16}$$

$$\ell_t^\mu(\theta) = -\sum_{z=1}^{Z} Q(s_t, z, x_z(s_t; \theta); \omega_t). \tag{17}$$

To address the issue of unbounded continuous parameters, such as the payment from CSs to EVs, the charging power of CSs, and the computing tasks assigned to EVs, it is essential to establish reasonable boundaries for these parameters. Without these boundaries, successive parameter updates during training over multiple episodes can lead to values exceeding their intended ranges, potentially resulting in parameter values approaching infinity. To mitigate this issue, a technique called Inverting Gradients [24] can be employed, which aims to restrict the values of continuous parameters within a reasonable range:

$$\nabla_\sigma = \nabla_\sigma \cdot \begin{cases} (\sigma_{\max} - \sigma) / (\sigma_{\max} - \sigma_{\min}), & \text{increase } \sigma, \\ (\sigma - \sigma_{\min}) / (\sigma_{\max} - \sigma_{\min}), & \text{otherwise}, \end{cases} \tag{18}$$

where σ is continuous parameter bounded by $[\sigma_{\min}, \sigma_{\max}]$. Gradients are down-scaled as the parameter approaches to the boundaries and inverted if the parameter exceeds the range.

Each CS is modelled as an independent learning agent whose continuous action uses a deterministic policy network, and the discrete action adopts a value network based on the outcomes of continuous action network. After that, the state transition computes the optimal contract at each time step and update the system status. If an EV receives multiple contracts from CSs, it has to choose one with the closest social distance or one can maximize its utility.

VI. Experiment Study

To validate the reinforcement contract design framework, we perform a comparative analysis with several benchmark approaches: the optimal solutions obtained by solving the deterministic optimization problem denoted as φ_{opt}^*, the parameterized Deep Deterministic Policy Gradient (PA-DDPG)-based contract design [24], and the greedy strategy. For the greedy strategy, the CSs send contracts to the nearest EVs

[2]In Section V, we slightly abuse θ to represent network weights in reinforcement learning, notably θ also denotes type in contract design.

TABLE I
PARAMETER SETTING

Parameter	Value
Number I of CSs	$[3, 5, 10]$
Number J of EVs	$[30, 50, 100]$
Charging load capacity of CS$_i$ P_i^{max}	300kW
Number of chargers q_i of CS i	6
The probability of type-θ_k CS β_i	$1/I$
Speed v_j of EV j	$[30, 40]$ km/h
Fast and Slow charging power P_i^h, P_i^l	50kW, 12kW
The cost of computing energy consumption c_j^{cpu}	2.8×10^{-4} \$/J [37]
The parameter of computing energy consumption ω	10^{-8} [38]
The CPU unit time speed ν_j of EV j	$[1, 2]$ GHz [39]
The size of computing task B_i	$1,800 - 2,200$ M
Cost of mobile energy consumption c_e	0.79
Profit coefficient η	15
Number of CPU cycles of computation task c_d	2.4381×10^9 cycles/M

with a fixed payment for evenly-allocated computing tasks. The charging power provided to the EVs is based on their state of charge (SoC), ensuring that EVs with lower SoC receive faster charging services.

A. Experimental Setting

We designed three groups, namely G1, G2, and G3, each with a different number of charging stations (learning agents): $I = 3$, 5, and 10, respectively. Each charging station is equipped with three chargers capable of providing Level-2 AC (240-volt) slow charging and Level-3 DC fast charging to three EVs simultaneously. We define one time step as 30 minutes, and an operation period for our model is set to one day, consisting of $|\mathcal{T}| = 48$ steps. Following the approach in [40], we assume that the arrival rate I_t of EVs in the charging network at time t follows a Poisson distribution, given by $P(I_t) = \delta(t)^{I_t} e^{-\delta(t)}/I_t!$, where $I_t \in I$. The payment for computing tasks $[\Lambda_i^{min}, \Lambda_i^{max}]$ of CS 1-3 is chosen from the range of [6.00,6.67], [6.67,7.33], [7.33,8.00], respectively. The EVs are randomly distributed in an area of 10km$\times$10km for G1, 15km$\times$15km for G2, and 20km$\times$20km for G3. The distance between any two charging stations is approximately 5km, and the positions of each charging station are fixed. The level-2 and level-3 charging of the charging price to EV is \$0.04/min and \$0.25/min. The parameters of the social distance between EV and CS is defined as: the tendency factor α_n is set as 1.2, and $sd_{i,j}$ is taken from the range of [0,1], where a lower value indicates a closer social distance. The battery capacity of private and commercial EV is 55kWh and 60kWh, respectively. For EVs, the mass $\tilde{m}_j$ is set to 1,000kg, the gravitational acceleration g is 9.8m/s^2, the ground friction coefficient f_g is 0.018, the moving speed v_j of EV i is randomly chosen from the range of [30,40]km/h, the air resistance f_a is 0.4, and the windward area ϕ is $2m^2$. The rest parameter setting of the system model is listed in Table I.

Both of the parameter actor and Q networks have three hidden layers with $[512, 256, 128]$ nodes in each layer where Leaky ReLU is used as the activation function. The activation functions of the Parameter Actor output layer use ReLU, Tanh, Softmax for CS payment, charging power and computing task allocation, respectively, for different branches of continuous actions. As for the parameter setting of PA-DQN, the learning rate is set as 0.001 and 0.0001 for Parameter Actor and Q networks, and the soft-update parameters are $\tau = 0.01$ and 0.001. The discount factor γ is set to 0.99. The replay buffer capacity is 100,000. The initial value of exploration parameter ϵ is set as 0.8 with a decay to 0.01 during 1,000 episodes. The minibatch size is 128, and the number of training episodes is 2,000. We compare our PA-DQN-based RCD framework with a PA-DDPG-based approach, a greedy method, and the optimal solutions as a baseline by solving optimization problem φ_{opt}^*. The parameters used in PA-DDPG-based RCD is the same as the proposed RCD with PA-DQN, using OU-Noise to explore. These hyperparameters are selected empirically to accelerate the policy improvement.

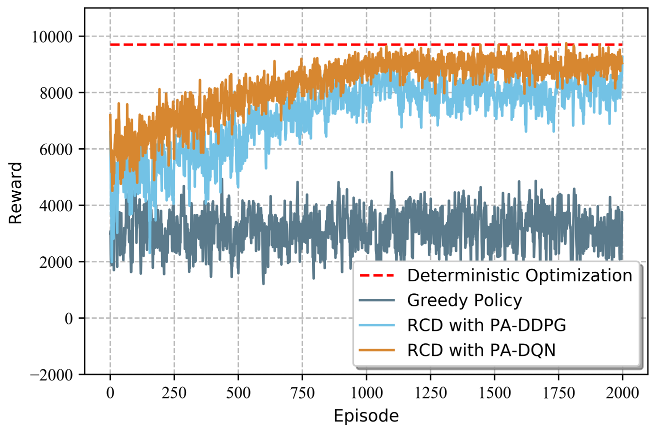


Fig. 3. Training process of different approaches (G1: 3 CSs/30 EVs).

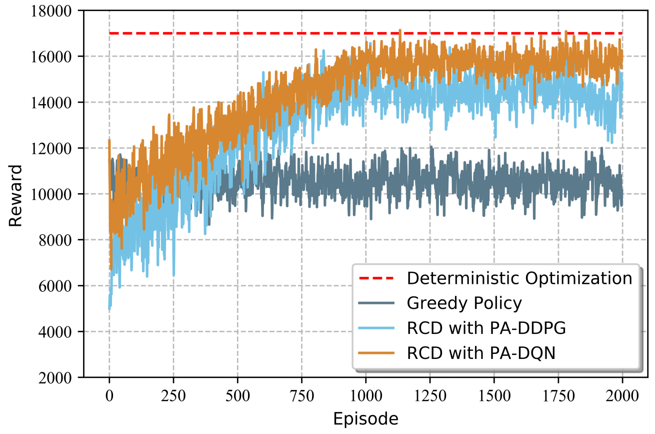


Fig. 4. Training process of different approaches (G2: 5 CSs/50 EVs).

To guarantee the solution optimality, the model φ_{opt}^* is coded in Python and solved by Gurobi. The greedy policy is also coded in Python. The reinforcement contract design using PA-DQN and PA-DDPG is coded in Python with PyTorch library. All experiments are carried out on an Intel Xeon E5-2620 v4 server with two RTX 2080Ti-11G GPU.

TABLE II
PERFORMANCE COMPARISON OF THREE GROUPS (G1: 3 CSS/30 EVS; G2: 5 CSS/50 EVS; G3: 10 CSS/100 EVS)

No.	Social welfare (/$)	Efficiency (/%)	CS utility (/$)	EV utility (/$)	Penalty (/$)	EV acceptance rate	Time (/min)
G1-RCD-PADQN	$9,015.7	92.3%	$9,925.7	$-910.0	$1,091.3	413/432	170.5
G1-RCD-PADDPG	$8,527.3	87.3%	$9,505.7	$-978.4	$1,527.3	401/432	665.1
G1-Greedy	$3,196.6	32.7%	$4,243.9	$-1,047.3	$5,841.1	363/432	2.3
G1-Optimization	$9,767.4	100%	$10,655.6	$-888.2	$402.2	425/432	11.5
G2-RCD-PADQN	$15,851.7	93.2%	$16,650.5	$-798.8	$2,540.4	678/720	289.9
G2-RCD-PADDPG	$14,460.3	85.0%	$15,326.1	$-865.8	$3,560.3	657/720	1,125.8
G2-Greedy	$10,541.3	62.0%	$11,442.8	$-901.5	$6,670.9	626/720	3.9
G2-Optimization	$17,014.9	100%	$17,758.3	$-743.4	$1480.5	698/720	21.6
G3-RCD-PADQN	$32,825.4	94.1%	$33,373.5	$-548.1	$6,964.0	1,331/1,440	599.3
G3-RCD-PADDPG	$26,970.4	77.3%	$27,459.4	$-489.0	$11,965.3	1,269/1,440	2,208.1
G3-Greedy	$21,178.5	60.7%	$21,818.4	$-639.9	$15,930.5	1,229/1,440	9.9
G3-Optimization	$34,876.7	100%	$35,345.6	$-468.9	$4,993.8	1,362/1,440	112.4

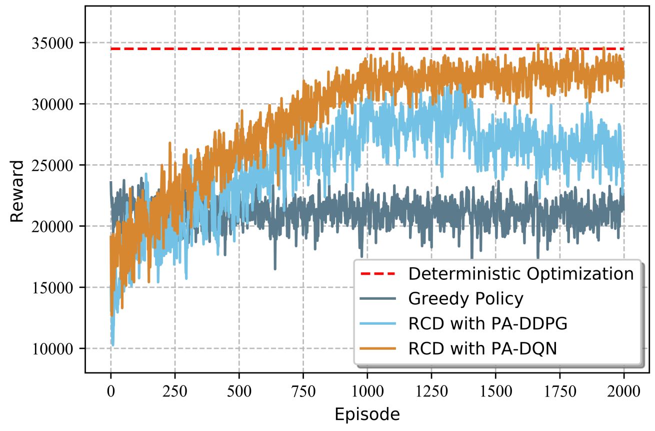


Fig. 5. Training process of different approaches (G3: 10 CSs/100 EVs).

B. Result Analysis

1) Performance comparison: We run each policy for duplicated 2,000 episodes (days) and draw the training process in Figs. 3, 4 and 5, where RCD with PA-DQN is compared with PA-DDPG-based RCD, the greedy policy, and the deterministic optimization as a baseline. In Figs. 3 and 4, we observe that the median of the social welfare (we take the mean value of converged rewards of the last 200 episodes) of PA-DQN $9,015.7 for Group 1 (3CSs and 30 EVs) and $15,851.7 for Group 2 (5CSs and 50 EVs) is higher than the the approach with PA-DDPG $8,527.3 and $14,460.3 by 5.73% and 9.62%, respectively. Unlike PA-DDQG which does not properly connect these hybrid actions by updating their policy weights independently, PA-DQN sequentially chooses the discrete and continuous actions. With the merits of both DQN and DDPG, PA-DQN can find the optimal discrete action as well as avoid exhaustive search over continuous action parameters. As a result, PA-DQN outperforms PA-DDPG in terms of computing time and efficiency. For G1, PA-DQN takes 170.5 mins while PA-DDPG costs 665.1 mins for 2,000 episodes. Moreover, PA-DQN takes around 1,000 episodes to converge while PA-DDPG takes 1,100 episodes.

The RCD framework can achieve 92.3% and 93.2% efficiency for G1 and G2 compared to the optimal solution (represented by the red dashed line) of $9,767.4 and $17,014.9 (deemed as 100% efficiency). Moreover, our approach surpasses the rewards obtained from the greedy policy, which amounts to $3,196.6 and $10,541.3 for G1 and G2, respectively, by a margin of 182.0% and 50.4%. This highlights the limitations of the greedy policy, as CSs following it prioritize short-term gains without considering potential long-term benefits. Uncoordinated charging and computing scheduling often lead to a significant number of unfinished tasks and penalties.

The detailed indices of the performance comparison are listed in Table II. The acceptance rate represents how many EVs accept the contracts from CSs in light of the social distance. For the acceptance rate, taking the last episode as an example, our approach accommodates 413 out of 432 EVs, achieving a 95.6% acceptance rate during a day. RCD with PA-DDPG achieves 92.8%, while the greedy policy only sees 84.0% of EVs accepting the contracts. Compared to RCD with PA-DDPG, CSs using our approach receive fewer penalties due to a reduced number of unfinished computing tasks accumulated during a day. This indicates a more reasonable schedule for computing and charging resources. It is worth noting that EVs receive negative utility across different approaches, as the payment for implementing computing tasks does not fully cover the charging costs. However, EVs providing computing services still pay less compared to solely charging at CSs, indicating potential cost savings.

2) Scalability testing: We design Group 3 (10 CSs with 100 EVs) to test the scalability of our RCD framework, the results are shown in Fig. 5 and Table II, which depicts that PA-DQN $32,825.4 improves the rewards by 21.7% compared to PA-DDPG with $26,970.4. Moreover, we observe from Table II that the performance of the RCD framework for different problem sizes. Specifically, the RCD framework achieves a higher efficiency of 94.1% compared to the optimal solution with a reward value of $34,876.7. In contrast, the greedy policy only achieves a reward of $21,178.5 but incurs a penalty of $15,930.5 due to a large number of unfinished tasks during a day. Notably, our approach is more time-efficient compared to PA-DDPG, despite the increasing problem scale. PA-DQN requires only 25.6%, 25.7%, and 27.4% of the computation time costed by PA-DDPG for G1-G3. Although reinforcement learning-based approaches require more time

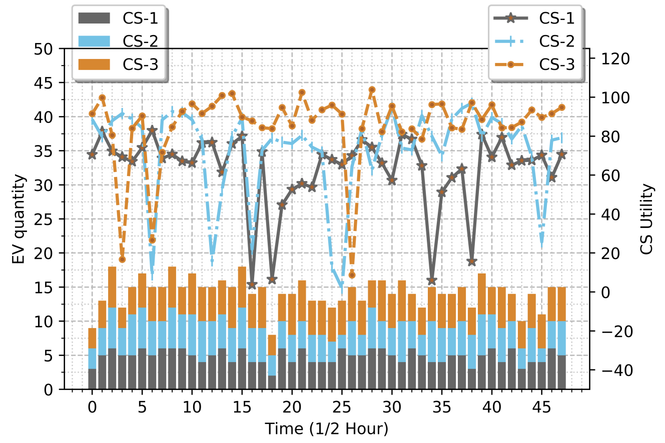


Fig. 6. Number of EVs served by each CS (RCD-PADQN).

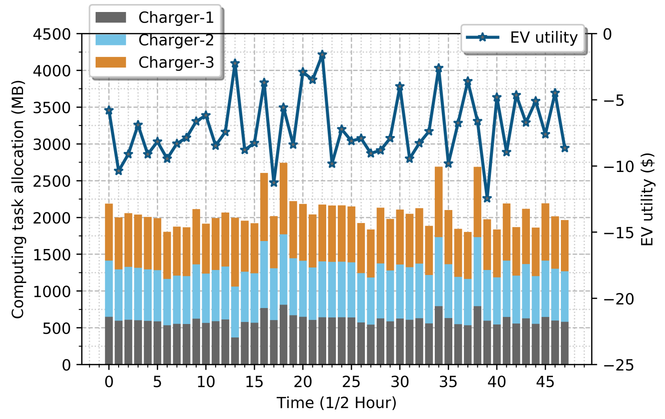


Fig. 7. Task allocation result (3 CSs and 30 EVs).

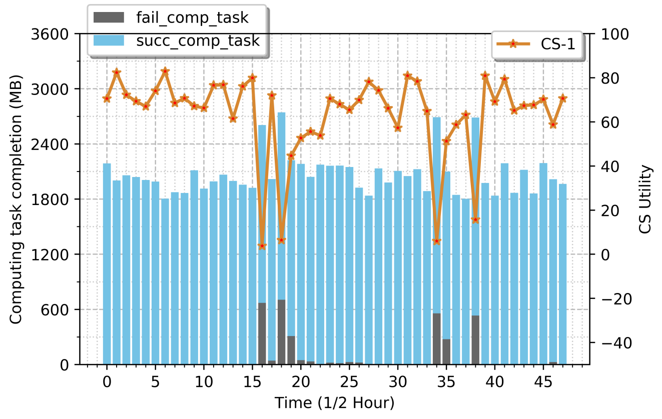


Fig. 8. Task completion level (3 CSs and 30 EVs).

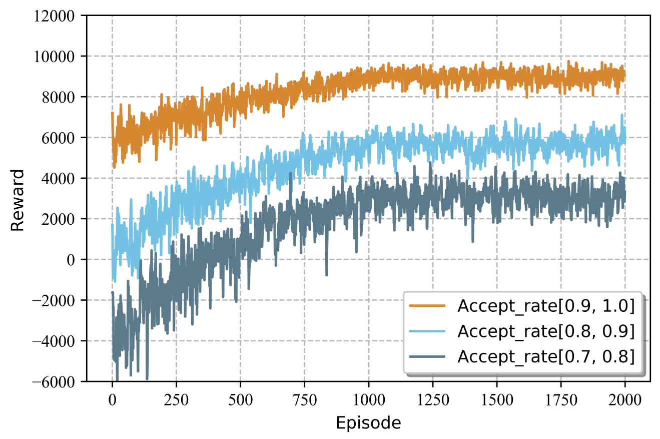


Fig. 9. Sensitivity test of different EV acceptance level (3 CSs and 30 EVs).

for the offline training, they are advantageous in dealing with online decision-making. Our approach takes less than one minute to design a one-day contract.

3) Contract design of one episode: To explicitly explain the EV-CS contracting process during a day, we take one episode of G1 (3 CSs and 30 EVs) as an example to illustrate the total number of EVs served and the utility of each CS in Fig. 6, the computing task allocation and charging process in Fig. 7, as well as the task completion level in Fig. 8 during a day.

Especially, Fig. 6 depicts the number of EVs served (the bar graph on the left side) by these three CSs during a day and the utility made by each CS, taking one episode as an example. Furthermore, we specify the task allocation results of this episode (3-CS/30-EV case) in Fig. 7. Taking CS-1 as an example, Fig. 7 illustrates the task execution of three chargers during a day, where the left side (bar graph) reveals the computing tasks (/MB) executed by each charger in CS-1 at each time step, and the right side displays the sum of EV utility during a day. It can be observed that after CS-1 sends the contract to three EVs, most of the EVs accept their contracts. Furthermore, there is a correlation between the size of computing tasks and the corresponding EV utilities, indicating that larger computing tasks result in higher EV utilities. Figure 8 demonstrates the CS utility curve with penalty (right side) accompanied with the amount of incomplete tasks (left side) during a day. The irrational task allocation or low EV acceptance will result in a sharp decrease of CS's utility.

4) Sensitivity testing: To examine the correlation between EV acceptance and social welfare, we conducted three additional experiments to compare different EV acceptance rates by manipulating the social distance between CSs and EVs. Specifically, we compared the acceptance rates of $[0.7, 0.8]$ and $[0.8, 0.9]$ with the existing results of G1 (consisting of 3 CSs and 30 EVs), which demonstrated a relatively high acceptance rate of $[0.9, 1.0]$. As depicted in Fig. 9, the mean reward for the case with an acceptance rate of $[0.7, 0.8]$ and $[0.8, 0.9]$ was found to be \$5,761.7 and \$3,092.7, respectively. These findings indicate that a lower EV acceptance rate corresponds to a greater social distance between CSs and EVs, which can potentially compromise the benefits for both parties involved. Moreover, each 10% increase in EV acceptance probability led to an approximate \$2,961.5 increase in rewards. From above, we conclude that the key to successful contract design lies in obtaining sufficient information and accurate estimation of EV users' preferences. Due to the presence of information asymmetry and privacy concerns, CSs can leverage historical data to acquire this knowledge and propose more efficient incentive mechanisms. These mechanisms aim to encourage EVs to truthfully disclose their preference information, ultimately enhancing the effectiveness of the overall charging system.

VII. Conclusion

This paper introduces a novel reinforcement contract design framework for addressing the vehicular-edge computing scheduling and energy trading problem. The framework utilizes a parameterized Deep Q-Network and draws inspiration from contract theory and reinforcement learning principles. By integrating the preferences of EV users, the utilities of charging stations, and various system constraints, the proposed model enables sequential decision-making in the presence of information asymmetry. To maximize long-term social welfare and handle discrete and continuous action spaces, we developed a reinforcement learning algorithm based on PA-DQN to leverage the sequential contract design with discrete and continuous action space. The experimental results demonstrated that our approach, utilizing PA-DQN, consistently outperformed the PA-DDPG-based RCD model. On average, our framework achieved a 15.5% improvement in social welfare and reduced computing time by 26.2%. Furthermore, our approach achieved near-optimal solutions when compared to deterministic optimization methods and significantly outperformed the greedy policy.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE communications surveys & tutorials*, vol. 19, no. 4, pp. 2322–2358, Aug. 2017.

[2] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE communications surveys & tutorials*, vol. 19, no. 3, pp. 1628–1656, Mar. 2017.

[3] T. N. Dang, A. Manzoor, Y. K. Tun, S. A. Kazmi, R. Haw, S. H. Hong, Z. Han, and C. S. Hong, "Joint communication, computation, and control for computational task offloading in vehicle-assisted multi-access edge computing," *IEEE Access*, vol. 10, pp. 122 513–122 529, Nov. 2022.

[4] J. Wu, "Collaborative mobile charging and coverage," *Journal of computer science and technology*, vol. 29, no. 4, pp. 550–561, Jul. 2014.

[5] C. Li, S. Wang, X. Huang, X. Li, R. Yu, and F. Zhao, "Parked vehicular computing for energy-efficient internet of vehicles: A contract theoretic approach," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6079–6088, Sep. 2018.

[6] Q. Tang, K. Wang, Y. Song, F. Li, and J. H. Park, "Waiting time minimized charging and discharging strategy based on mobile edge computing supported by software-defined network," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6088–6101, Dec. 2019.

[7] K. Kumar, J. Liu, Y.-H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile networks and Applications*, vol. 18, no. 1, pp. 129–140, Apr. 2013.

[8] H. Lin, S. Zeadally, Z. Chen, H. Labiod, and L. Wang, "A survey on computation offloading modeling for edge computing," *Journal of Network and Computer Applications*, vol. 169, p. 102781, Nov. 2020.

[9] H. Zhou, T. Wu, X. Chen, S. He, D. Guo, and J. Wu, "Reverse auction-based computation offloading and resource allocation in mobile cloud-edge computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 6144–6159, Jul. 2022.

[10] L. Hou, C. Wang, and J. Yan, "An incentive-compatible combinatorial auction design for charging network scheduling of battery electric vehicles," *Journal of Integrated Design and Process Science*, vol. 24, no. 2, pp. 75–92, Dec. 2020.

[11] C. Su, F. Ye, T. Liu, Y. Tian, and Z. Han, "Computation offloading in hierarchical multi-access edge computing based on contract theory and bayesian matching game," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13 686–13 701, Nov. 2020.

[12] L. Hou, S. Ma, J. Yan, C. Wang, and J. Y. Yu, "Reinforcement mechanism design for electric vehicle demand response in microgrid charging stations," in *2020 International Joint Conference on Neural Networks (IJCNN)*, Glasgow, UK, Sep. 2020, pp. 1–8.

[13] Y. M. Saputra, D. Nguyen, H. T. Dinh, T. X. Vu, E. Dutkiewicz, and S. Chatzinotas, "Federated learning meets contract theory: economic-efficiency framework for electric vehicle networks," *IEEE Transactions on Mobile Computing*, vol. 21, no. 8, pp. 2803–2817, Aug. 2022.

[14] J. R. Vázquez-Canteli and Z. Nagy, "Reinforcement learning for demand response: A review of algorithms and modeling techniques," *Applied Energy*, vol. 235, pp. 1072–1089, Feb. 2019.

[15] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in neural information processing systems (NIPS)*, California, USA, Dec. 2017, pp. 6379–6390.

[16] M. Tang and V. W. Wong, "Deep reinforcement learning for task offloading in mobile edge computing systems," *IEEE Transactions on Mobile Computing*, vol. 21, no. 6, pp. 1985–1997, Jun. 2022.

[17] S. Yu, B. Dab, Z. Movahedi, R. Langar, and L. Wang, "A socially-aware hybrid computation offloading framework for multi-access edge computing," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1247–1259, Jun. 2019.

[18] Y. Zhang, B. Di, Z. Zheng, J. Lin, and L. Song, "Distributed multi-cloud multi-access edge computing by multi-agent reinforcement learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2565–2578, Dec. 2020.

[19] W. Zhan, C. Luo, J. Wang, C. Wang, G. Min, H. Duan, and Q. Zhu, "Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5449–5465, Jun. 2020.

[20] L. Ale, S. A. King, N. Zhang, A. R. Sattar, and J. Skandaraniyam, "D3pg: Dirichlet ddpg for task partitioning and offloading with constrained hybrid action space in mobile edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 19 260–19 272, Oct. 2022.

[21] L. Hou, Y. Li, J. Yan, C. Wang, L. Wang, and B. Wang, "Multi-agent reinforcement mechanism design for dynamic pricing-based demand response in charging network," *International Journal of Electrical Power & Energy Systems*, vol. 147, p. 108843, May. 2023.

[22] S. Yin and F. R. Yu, "Resource allocation and trajectory design in uav-aided cellular networks based on multiagent reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2933–2943, Jul. 2021.

[23] J. Xiong, Q. Wang, Z. Yang, P. Sun, L. Han, Y. Zheng, H. Fu, T. Zhang, J. Liu, and H. Liu, "Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space," *arXiv preprint arXiv:1810.06394*, 2018.

[24] M. Hausknecht and P. Stone, "Deep reinforcement learning in parameterized action space," *arXiv preprint arXiv:1511.04143*, 2015.

[25] L.-T. Hsieh, H. Liu, Y. Guo, and R. Gazda, "Deep reinforcement learning-based task assignment for cooperative mobile edge computing," *IEEE Transactions on Mobile Computing*, Apr. 2023, early Access, DOI: 10.1109/TMC.2023.3270242.

[26] F. Song, H. Xing, X. Wang, S. Luo, P. Dai, Z. Xiao, and B. Zhao, "Evolutionary multi-objective reinforcement learning based trajectory control and task offloading in uav-assisted mobile edge computing," *IEEE Transactions on Mobile Computing*, pp. 1–18, Sep. 2022.

[27] L. Hou, C. Wang, and J. Yan, "Bidding for preferred timing: An auction design for electric vehicle charging station scheduling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 8, pp. 3332–3343, Jul. 2019.

[28] J. Zhao, M. Kong, Q. Li, and X. Sun, "Contract-based computing resource management via deep reinforcement learning in vehicular fog computing," *IEEE Access*, vol. 8, pp. 3319–3329, Dec. 2019.

[29] L. Hou, J. Yan, C. Wang, and L. Ge, "A simultaneous multi-round auction design for scheduling multiple charges of battery electric vehicles on highways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8024–8036, Jul. 2022.

[30] Y. Wang, L. Wang, L. Xu, Z. Sun, and K. Sima, "Matching based joint trading contract of energy and computation in virtual power plant," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 2560–2565.

[31] W. Wang, L. Wang, C. Zhang, C. Liu, L. Sun *et al.*, "Social interactions for autonomous driving: A review and perspectives," *Foundations and Trends® in Robotics*, vol. 10, no. 3-4, pp. 198–376, Nov. 2022.

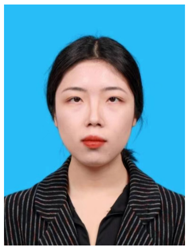
[32] M. Boguná, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, "Models of social networks based on social distance attachment," *Physical review E*, vol. 70, no. 5, p. 056122, Nov. 2004.

[33] P. Bolton and M. Dewatripont, *Contract theory*. MIT press, 2004.
[34] R. Srikant, *The mathematics of Internet congestion control*. Springer Science & Business Media, 2004.
[35] S. Maharjan, Q. Zhu, Y. Zhang, S. Gjessing, and T. Basar, "Dependable demand response management in the smart grid: A stackelberg game approach," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 120–132, Mar. 2013.
[36] R. S. Sutton, A. G. Barto *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 2, no. 4.
[37] Y. Ruan, Y. Li, C.-X. Wang, and R. Zhang, "Energy efficient adaptive transmissions in integrated satellite-terrestrial networks with ser constraints," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 210–222, Oct. 2017.
[38] M. Guan, B. Bai, L. Wang, S. Jin, and Z. Han, "Joint optimization for computation offloading and resource allocation in internet of things," in *IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Toronto, ON, Canada, Sep. 2017, pp. 1–5.
[39] L. Xu, Z. Yang, H. Wu, Y. Zhang, Y. Wang, L. Wang, and Z. Han, "Socially driven joint optimization of communication, caching, and computing resources in vehicular networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 1, pp. 461–476, Jul. 2021.
[40] A. Y. Lam, K.-C. Leung, and V. O. Li, "Capacity estimation for vehicle-to-grid frequency regulation services with smart charging mechanism," *IEEE transactions on smart grid*, vol. 7, no. 1, pp. 156–166, Jun. 2015.

Li Wang (Senior Member, IEEE) received the Ph.D. degree from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2009. She is currently a Full Professor with the School of Computer Science (National Pilot Software Engineering School), BUPT. She is an Associate Dean and the Head of the High Performance Computing and Networking Laboratory. She is also the Vice Dean of Key Laboratory of Application Innovation in Emergency Command Communication Technology, Ministry of Emergency Management, and Member of the Key Laboratory of the Universal Wireless Communications, Ministry of Education, Beijing. She held visiting positions with the School of Electrical and Computer Engineering, Georgia Tech, Atlanta, GA, USA, from December 2013 to January 2015, and the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden, from August 2015 to November 2015 and from July 2018 to August 2018. She has authored/coauthored almost 70 journal papers and four books. Her current research interests include wireless communications, distributed networking and storage, vehicular communications, social networks, and edge AI. She currently serves on the Editorial Boards for IEEE Transactions on Vehicular Technology, IEEE Transactions on Cognitive Communications and Networking, Computer Networks, and China Communications. She was the Associate Editor of IEEE Transactions on Green Communications and Networking, the Symposium Chair of IEEE ICC 2019 on Cognitive Radio and Networks Symposium and a Tutorial Chair of IEEE VTC 2019-fall. She also serves as the vice chair of Meetings and Conference Committee (MCC) for IEEE Communication Society (ComSoc) Asia Pacific Board (APB) for the term of 2020-2021, and chairs the special interest group (SIG) on Sensing, Communications, Caching, and Computing (C3) in Cognitive Networks for IEEE Technical Committee on Cognitive Networks. She was the recipient of the 2013 Beijing Young Elite Faculty for Higher Education Award, Best Paper Awards from several IEEE conferences, e.g., IEEE ICCC 2017, IEEE GLOBECOM 2018, IEEE WCSP 2019, and so forth. She was also the recipient of the Beijing Technology Rising Star Award in 2018. She has served on TPC of multiple IEEE conferences, including IEEE Infocom, Globecom, International Conference on Communications, IEEE Wireless Communications and Networking Conference, and IEEE Vehicular Technology Conference in recent years.

Luyang Hou (S'17-M'21) received the B.Eng. degree in mechanical design, manufacturing, and automation from Henan Polytechnic University, China, in 2013, and the M.S. degree in micro-electromechanical engineering from Dalian University of Technology, China, in 2016. He received the Ph.D. degree in information systems engineering from Concordia University, Canada, in 2020.

He is currently a Research Associate Professor with the School of Computer Science (National Pilot Software Engineering School), Beijing University of Posts and Telecommunications (BUPT), China. Before joining BUPT, he was a Postdoctoral Research Fellow at the Energy Innovation Laboratory of MéridaLabs, The University of British Columbia from January 2021 to February 2022, and at the Institute for Information Systems Engineering, Concordia University from March 2022 to October 2022. He was also a Visiting Researcher at the Institute of Semiconductors, Chinese Academy of Sciences, from July to October in 2022. His research interests include operational optimization, mechanism design, and machine learning with applications to Energy Internet and Intelligent Command & Scheduling. He also develops AI-economic paradigm and applies it to the cross-realm energy and information network, such as vehicle edge computing, multi-energy system, electricity market design, packetized energy, UAV formation, and so on.



Sixuan Liu received the B.Eng degree in Internet of Things Engineering from Beijing University of Posts and Telecommunications (BUPT), China, in 2020, and the master's degree of Computer Technology at BUPT in 2023. Her research interests include reinforcement learning theory with applications to the energy Internet and virtual power plant.

Zhu Han (S'01–M'04-SM'09-F'14) received the B.S. degree in electronic engineering from Tsinghua University, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively.

From 2000 to 2002, he was an R&D Engineer of JDSU, Germantown, Maryland. From 2003 to 2006, he was a Research Associate at the University of Maryland. From 2006 to 2008, he was an assistant professor at Boise State University, Idaho. Currently, he is a John and Rebecca Moores Professor in the Electrical and Computer Engineering Department as well as in the Computer Science Department at the University of Houston, Texas. Dr. Han's main research targets on the novel game-theory related concepts critical to enabling efficient and distributive use of wireless networks with limited resources. His other research interests include wireless resource allocation and management, wireless communications and networking, quantum computing, data science, smart grid, carbon neutralization, security and privacy. Dr. Han received an NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the Journal on Advances in Signal Processing in 2015, IEEE Leonard G. Abraham Prize in the field of Communications Systems (best paper award in IEEE JSAC) in 2016, and several best paper awards in IEEE conferences. Dr. Han was an IEEE Communications Society Distinguished Lecturer from 2015-2018, AAAS fellow since 2019, and ACM distinguished Member since 2019. Dr. Han is a 1% highly cited researcher since 2017 according to Web of Science. Dr. Han is also the winner of the 2021 IEEE Kiyo Tomiyasu Award (an IEEE Field Award), for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: "for contributions to game theory and distributed management of autonomous communication networks."

Jie Wu is Laura H. Carnell Professor at Temple University and the Director of the Center for Networked Computing (CNC). He served as Chair of the Department of Computer and Information Sciences from the summer of 2009 to the summer of 2016 and Associate Vice Provost for International Affairs from the fall of 2015 to the summer of 2017. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University, where he received his Ph.D. in 1989. His current research interests include mobile computing and wireless networks, routing protocols, network trust and security, distributed algorithms, applied machine learning, and cloud computing. Dr. Wu regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Service Computing and Journal of Computer Science and Technology. Dr. Wu is/was general chair/co-chair for IEEE DCOSS'09, IEEE ICDCS'13, ICPP'16, IEEE CNS'16, WiOpt'21, ICDCN'22, IEEE IPDPS'23, and ACM MobiHoc'23 as well as program chair/cochair for IEEE MASS'04, IEEE INFORCOM'11, CCF CNCC'13, and ICCCN'20. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a Fellow of the AAAS and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award. He is a Member of the Academia Europaea (MAE).