A Multi-Agent Reinforcement Learning Approach for Blockchain-based Electricity Trading System

Yifan Cao[1], Xiaoxu Ren[1], Chao Qiu[1], Xiaofei Wang[1], Haipeng Yao[2], F. Richard Yu[3]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]School of Information and Communications Engineering,
Beijing University of Posts and Telecommunications, Beijing, China
[3]School of Information Technology, Carleton University, Ottawa, Canada
Email: {yifancao, xiaoxuren, chao.qiu, xiaofeiwang}@tju.edu.cn
yaohaipeng@bupt.edu.cn, richardyu@cunet.carleton.ca

GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685510

***Abstract*—In microgrid, peer-to-peer (P2P) electricity trading has quickly ascended to the spotlight and gained enormous popularity. However, there are inevitable credit problems and system security problems. Besides, the current model in the electricity trading system cannot balance the utilities of multiple trading entities. In this paper, we propose a blockchain-based distributed P2P electricity trading system. We define elecoins as currency in circulation within our trading system. In order to jointly optimize the utilities of both parties in the elecoins trading, we formulate the elecoins purchasing problem as a hierarchical Stackelberg game. Then, we design a distributed multi-agent utility-balanced reinforcement learning (DMA-UBRL) algorithm to search the Nash equilibrium. Finally, we factually build a blockchain system with a blockchain explorer and deploy an electricity trading smart contract (ETSC) on Ethereum, with a website interface for operating. The numerical results and the implemented realistic system show the advantages of our work.**

***Index Terms*—Microgrid, P2P electricity trading, blockchain, Stackelberg game, multi-agent reinforcement learning**

I. Introduction

With the increase of distributed generators and storage devices participating in microgrid, the Energy Internet has been proposed [1]. Specifically, the Energy Internet fully unleashes the potential of traditional power consumers, enabling them to work as prosumers with the capabilities of generating, consuming, and selling electricity. Several prosumers with their own power demands and generations in short distance can be modeled as a peer-to-peer (P2P) network. Such an electricity trading mode promises several benefits, such as low load peaks, low transmission loss and so on [2]. However, the traditional P2P trading mode has inevitable credit crisis and privacy security issues for prosumers to conduct large-scale decentralized electricity trading [3]. Therefore, it is necessary to establish an authentic and secure electricity trading system.

Blockchain technology, with the characteristics of decentralization, anonymity, and credibility [4], has been introduced into the energy trading system, aiming to felicitously solve the above-mentioned issues. For example, a local P2P trading mode, has been proposed in plug-in hybrid electric vehicles [5]. The authors adopt an iterative double pricing mechanism to solve the maximum problem of social welfare in electricity trading. In [6], the authors propose a unified energy consortium blockchain and design a payment scheme based on a credit bank. This work solves the problem of cold start in blockchain and maximizes the benefits of credit banks for loan.

Although these works are capable of achieving some advantages using blockchain in energy trading, numerous handicaps prevent it from being used as a generic platform: **I) Credit crisis:** the current bank-based credit mechanism is vulnerable for single point of attack, while discouraging prosumers from purchasing electricity. **II) Unbalanced utility:** the current solution of the loan process focuses on maximizing the utility of credit banks, without considering the utility balance between credit banks and prosumers. **III) System implementation:** the current works about blockchain-enabled electricity trading system have not been factually designed and deployed.

In order to reach a utility-balanced situation, the majority of current researches leverage the game-theoretic approaches to model the interaction between parties in the trading process. [7]–[11]. Generally, by transforming the bi-level model into an equivalent single-level model using Karush-Kuhn-Tucher (KKT) conditions, quite a few works solve the Stackelberg game model [12]. Besides, some optimization algorithms, such as backward induction and differential evolution algorithm [13], are also applied to solve the Stackelberg game model. These traditional methods usually assume that there is a centralized organization to collect users' information and assist them to formulate the relevant policies. However, in reality, individuals' complete information cannot be well acquired, especially for some privacy parameters. Furthermore, the sequence game is widely used between trading entities, which means entities' observation of opponents and corresponding policies are decided by themselves, instead of the centralized organization. In order to conquer these strong assumptions, we adopt a multi-agent reinforcement learning (MARL) algorithm with incomplete information to search for the game equilibrium.

In this article, we propose a blockchain-based electricity trading system, aiming to solve the problem of credit crisis and privacy security. It should be noted that a currency elecoin is designed and circulated in the trading system. The main contributions of this article are summarized as follows:

978-1-7281-8104-2/21/$31.00 ©2021 IEEE

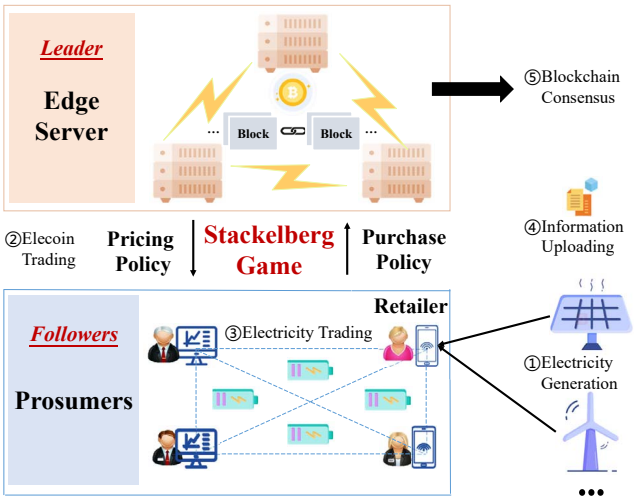


Fig. 1: Blockchain-enabled P2P electricity trading framework.

- We propose a blockchain-based distributed P2P electricity trading system, which ensures the reliability and privacy security in the trading process.
- We formulate the elecoins trading problem as a hierarchical Stackelberg game, giving the proof of existence and uniqueness of the Nash equilibrium (NE). Besides, in view of the dynamic game of distributed multiple participants, we design a distributed multi-agent utility-balanced reinforcement learning algorithm (DMA-UBRL) to search the equilibrium.
- We factually construct a local blockchain system with three edge nodes. In response to the credit crisis and trading security, we design and deploy an electricity trading smart contract (ETSC) on Ethereum[1], providing a website interface and blockchain explorer for prosumers to operate their accounts.

The rest of this article is organized as follows. The elecoins trading model and problem formulation are shown in Section II. In Section III, the distributed multi-agent utility-balanced reinforcement learning algorithm is conceived for searching the Nash equilibrium of the proposed game model. Besides, a smart contract for electricity trading is designed. Section IV presents the simulation results and the system deployment. Finally, conclusion is given in Section V.

II. System Model and Problem Formulation

A. System Description

We design an electricity trading system based on blockchain. The entities and process of electricity trading are illustrated in Fig. 1. In the electricity trading system, there exist three types of roles, including **1) Prosumers**, who purchase elecoins from edge servers, as well as request electricity from retailers with adequate electricity. **2) Retailers**, who are also prosumers in the trading system, generating, storing and selling reliable electricity from distributed energy sources, such as photovoltaic power, solar thermal power and wind power. **3) Edge servers**, on the one hand, who could provide elecoins and make the pricing policy for prosumers. On the other hand, the edge servers play the role of miners in blockchain.

The trading framework can be described as shown in Fig.1. Specifically, the process is summarized as follows. **1) Electricity generation:** the retailers generate and store electricity from the distributed energy sources. **2) Elecoin trading:** when prosumers have demands of electricity from others, they firstly need elecoins by paying to the edge servers. **3) Electricity trading:** then, prosumers can freely conduct electricity trading with retailers through elecoins. **4) Information uploading:** the detailed transaction data and account information updates will be uploaded to the blockchain. **5) Blockchain consensus:** once an edge server receives the data and successfully obtains the priority to record transactions, it will package a block and broadcast it to the whole blockchain. Other edge servers receive the new block, validate its rationality, and attach it to their blockchain.

B. System Model

In the elecoins trading system, we consider there are N prosumers and M edge servers, denoted as $\mathcal{N} = \{1, 2, \ldots, i, \ldots, N\}$ and $\mathcal{M} = \{1, 2, \ldots, j, \ldots, M\}$, respectively. There are two models used:

1) PoW-based Mining Model: The edge servers not only offer elecoins to prosumers, but also participate in the mining process. We assume that all the hashing computing capacity α_j of edge server j is used for mining and the hashing computing capacity proportion of edge server j in the whole blockchain is $\beta_j = \frac{\alpha_j}{\sum_{j=1}^{M} \alpha_j}$, where $\sum_{j=1}^{M} \beta_j = 1$. Specifically, we model the appearance of solving the PoW hashing puzzle as a Poisson process [9]. Accordingly, the probability of edge server j successfully solving the puzzle and reaching consensus can be expressed as: $\rho_j = \beta_j e^{-\lambda l s}$, where λ is the propagation factor of the blockchain and l is an evaluation metric of puzzle in the blockchain, s means the size of a block. For simplification, we assume each block contains the same number of transactions.

Additionally, with the generation of blocks, there are two types of reward for the edge server: fixed reward denoted by R_f and performance reward R_p defined by $r \times s$. Therein, r is a given variable reward factor and s decides the number of transactions in a block. Thus, the reward function of edge server j who successfully mines a block is expressed as:

$$U_j^m = (R_f + R_p) \cdot \beta_j e^{-\lambda l s}. \tag{1}$$

2) Stackelberg Game Model: We formulate the interaction between an edge server and prosumers as a hierarchical Stackelberg game, assuming that there is an edge server supplying elecoins to the N prosumers. At the beginning, the edge server declares a uniform unit elecoin price, denoted as $p \in [p_{min}, p_{max}]$, where p_{min} and p_{max} represent the minimum and maximum unit elecoin price, respectively. According to the declared price, prosumers consider their own demands of elecoins, denoted as $X = \{x_1, x_2, \ldots, x_i, \ldots, x_N\}$, and

[1]https://ethereum.org

$X \in [x_{min}, x_{max}]$, where x_{min} and x_{max} are the minimum and maximum demands of elecoins purchased by the prosumers, respectively. Both the edge server and prosumers expect to achieve optimal utilities.

Edge server's pricing strategies in Stage I: The utility of edge server j is consists of the reward that the edge server successfully mines a block, the charging fee from prosumers for electricity service, and the service cost. Therefore, the utility function of service server j can be denoted as follows:

$$U_j^e = U_j^m + \sum_{i=1}^{N} p x_i - k(\frac{\sum_{i=1}^{N} x_i}{Q} + e_c), \tag{2}$$

where k is a conversion factor and e_c represents electricity consumption of the edge server. We assume that the unit electricity price Q is related with the electricity price of the centralized utility grid [8].

Prosumers' demand strategies in Stage II: Prosumers have different intentions to utilize electricity service [10], defined by $W = \{w_1, w_2, \ldots, w_i, \ldots, w_N\}$. Considering the marginal benefit of electricity consumption, we adopt a logarithm model for services revenue. Regarding θ and η as conversion factors, the utility of prosumer i can be obtained by the services revenue minus the payment for elecoins:

$$U_i^p = \theta ln(\frac{x_i}{Q} w_i + 1) - \eta x_i p. \tag{3}$$

C. Game Analysis

In order to verify the existence and uniqueness of the Stackelberg equilibrium in our hierarchical Stackelberg game, we can arrive the following theorems.

Theorem 1. *The Nash equilibrium for prosumer i in the Stackelberg game is given by*

$$x_i^* = \frac{\theta}{\eta p} - \frac{Q}{w_i}. \tag{4}$$

Proof: According to the equation (3), we have the second derivatives of utility U_i^p with respect to demands x_i,

$$\frac{\partial^2 U_i^p}{\partial^2 x_i} = \frac{-\theta w_i^2}{(x_i w_i + Q)^2}. \tag{5}$$

The second derivative is negative due to $\theta > 0$ and $Q > 0$, proving the concavity of the utility function. Then, if there exists x_i that makes the first derivative 0, it is the unique maximum of the function. Thus, we set $\frac{\partial U_i^p}{\partial x_i} = 0$ and the proof is completed. ∎

Theorem 2. *The edge server achieves the utility maximization, under the unique optimal price.*

Proof: In the stage I, the pricing strategy of the edge server depends on hypothetical elecoins demand x_i. When we get the NE in the stage II, we try to find the best pricing strategy for the edge server to obtain the optimal utilities. After substituting (4) into (2), we can arrive

$$\begin{aligned} U_j^e = (R_f + R_p) \cdot \beta_j e^{-\lambda l s} + \frac{p - \frac{k}{Q}}{p} \cdot \frac{N\theta}{\eta} - \\ (p - \frac{k}{Q}) \cdot \sum_{i=1}^{N} \frac{Q}{w_i} - k e_c. \end{aligned} \tag{6}$$

From (6), we have the second derivatives of U_j^e with respect to unit elecoin price p given as follows:

$$\frac{\partial^2 U_j^e}{\partial^2 p} = -2\frac{k}{p^3} \cdot \frac{N\theta}{\eta Q} < 0. \tag{7}$$

Due to the negativity of (7), the strict concavity of the objective function is ensured. Thus, the prosumer is able to achieve the maximum utility with the unique optimal price. The proof is now completed. ∎

Accordingly, the Stackelberg equilibrium exists while it is unique. As such, both the edge server and prosumers can achieve the optimal utilities in the utility-balanced trading.

III. MARL based Elecoins Trading and Smart Contract based Electricity Sharing

A. Multi-Agent Reinforcement Learning based Elecoins Trading

To prevent privacy parameters from disclosure among trading and obtain balanced utilities of all the trading entities, we model the elecoins trading process as a Markov Decision Process (MDP) and design DMA-UBRL algorithm to solve the problem, as shown in Fig. 2. Then the state, action, and reward function are presented.

We consider $\mathcal{A}_p$ and $\mathcal{A}_e$ as the action spaces of the prosumers and edge server j, respectively. At time slot t, in the beginning of MDP, the edge server first sets the uniform unit elecoin price p^t based on state $x_i^t = [x_i^{t-1}]_{i \in N}$ observed from the underling game, where x_i^{t-1} indicates the submitted amount of elecoin in prosumer i at time slot $(t-1)$. Then, we define the MDP of the edge server. Therein, $s_e^t = [x_i^{t-1}]_{i \in N}$ denotes the state space, the action space is signified by $p^t \in \mathcal{A}_e$, while the reward function of edge server j is the utility function in (2).

Additionally, for the prosumers, after observing the action of the edge server at time slot t, each of them determines its submitted purchase action x_i^t based on the observed state. Similarly, we define the MDP of the prosumers, where $s_p^t = p^t$ represents the state space and $X^t \in \mathcal{A}_p$ represents the action space. The reward function of a prosumer is the utility function in (3).

We design a DMA-UBRL algorithm for searching the NE in the non-stationary environment constructed by multi-agent. Algorithm 1 shows the details of DMA-UBRL algorithm for the edge server in the multi-agent Stackelberg game. Let $\sigma_e \in (0, 1]$ and $\gamma_e \in (0, 1]$ represent the learning rate and future reward importance factor, respectively. Then, the updating rule of the Q-value related with uniform unit elecoin price p in state s_e^t can be updated by step 4.

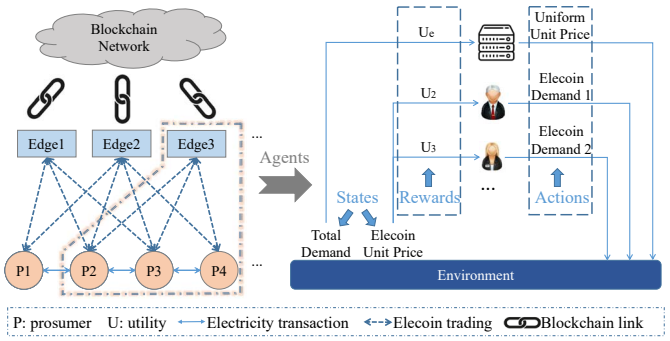


Fig. 2: Architecture of multi-agent reinforcement learning algorithm in elecoins trading process.

Next, the edge server updates its current average policy $\bar{\pi}_e(s_e^t, p)$ and current unit elecoin price policy $\pi_e(s_e^t, p)$, where $\bar{\pi}_e(s_e^t, p)$ is leveraged to judge "win" or "failure" of the current strategy. The average policy can be updated by step 5, where $M(s_e^t)$ is used for recording the occurrence counts of states observed by the agent. Through interactions with the environment and other agents, the edge server can update its current price policy $\pi_e(s_e^t, p)$ to maximize the cumulative reward, which is given by step 6, where

$$\Phi_{s_e^t,p} = \begin{cases} -\min\left(\pi_e(s_e^t,p), \frac{\varepsilon_e}{|\mathcal{A}_e|-1}\right), & \Lambda, \\ \sum_{p'\neq p}\pi_e\left(s_e^t,p'\right), \frac{\varepsilon_e}{|\mathcal{A}_e|-1}\right), & \text{otherwise.} \end{cases}$$

where $\Lambda : p \neq \arg\max_{p' \in \mathcal{A}_e} Q_e(s_e^t, p')$, $\mid \mathcal{A}_e \mid$ is the number of the edge server's action set. Specifically, the edge server would choose its learning parameter ε_e from ε_e^{win} and ε_e^{lose}. Under the condition of win, ε_e^{win} is applied to update the policy cautiously. Otherwise, ε_e^{lose} is used to learn fast under the "failure" condition, i.e.,

$$\varepsilon_e = \begin{cases} \varepsilon_e^{win}, & \Lambda', \\ \varepsilon_e^{lose}, & \text{otherwise,} \end{cases}$$

where Λ' indicates the condition that $\sum_{p\in\mathcal{A}_e} \pi_e\left(s_e^t,p\right)Q_e\left(s_e^t,p\right) > \sum_{p\in\mathcal{A}_e} \bar{\pi}_e\left(s_e^t,p\right)Q_e\left(s_e^t,p\right)$.

The computational complexity mainly focuses on the strategy update. We assume the number of elements in state space and action space denoted as S and A, respectively. The complexity of the agent is $O(S^2 \times A)$.

Similar to Algorithm 1, we can obtain the purchase policies of prosumers using DMA-UBRL algorithm.

B. Smart Contract based Electricity Sharing

Once obtaining elecoins from edge servers through the Stackelberg game, prosumers are able to purchase electricity from other prosumers with adequate energy reserves. Considering the credit crisis among prosumers, we construct an electricity blockchain and ETSC to ensure trading executed reliably and prosumers' accounts dealt accurately.

ETSC is realized by four parts, i.e., Registration Module, Transaction Module, Electricity Clearing Module and Quire Module. As shown in Algorithm 2, in Registration Module,

Algorithm 1. DMA-UBRL Algorithm for the Edge Server.

Input: σ_e, ϕ_e, ε_e^{win}, ε_e^{lose}.
Initialization: $t = 1$, $Q_e(s_e^t, p^t) = 0$, $\pi_e(s_e^t, p^t) = \frac{1}{|\mathcal{A}_e|}$ $\bar{\pi}_e(s_e^t, p^t) = \frac{1}{|\mathcal{A}_e|}, \varepsilon_e^{win} < \varepsilon_e^{lose}, M(s_e^t) = 0$.
for $t = 1, 2, 3, \cdots$ **do**
1. Observe state s_e^t.
2. Select action p^t at random with the probability policy $\pi_e(s_e^t, p^t)$.
3. Observe next reward R_e and state s_e^{t+1}.
4. Update $Q_e(s_e^t, p^t)$:
$Q_e(s_e^t, p^t) \leftarrow (1-\sigma_e)Q_e(s_e^t, p^t) + \sigma_e \cdot (R_e + \gamma_e \max_{p\in\mathcal{A}_e} Q_e(s_e^{t+1}, p))$.
5. Update average policy $\bar{\pi}_e(s_e^t, p)$:
$M(s_e^t) = M(s_e^t) + 1$
$\bar{\pi}_e(s_e^t, p) \leftarrow \bar{\pi}_e(s_e^t, p) + \frac{1}{M(s_e^t)}(\pi_e(s_e^t, p) - \bar{\pi}_e(s_e^t, p))$, $\forall p \in A_e$.
6. Update current strategy $\pi_e(s_e^t, p)$:
$\pi_e(s_e^t, p) \leftarrow \pi_e(s_e^t, p) + \Phi_{s_e^t,p}$, $\forall p \in A_e$

end for

it is necessary to initialize the whole blockchain system. By $constructor()$ in the smart contract, the blockchain registers the selected edge servers as an administrator of the trading system, and initializes their accounts information and electricity price. When prosumers intend to participate in trading, they are required to provide their names, addresses, generation intentions and trading intentions. After registration, prosumers' accounts will be initialized, including their available electricity, as well as total generated electricity and used electricity.

In Transaction Module, considering the cold start issue, prosumers desired to purchase electricity could acquire elecoins from edge servers by $buyToken()$function. Prosumers are able to select retailers, while prosumers choosing to become the retailers can also put on their commodity information. After that, prosumers confirm the exact quantity of electricity y_i and place the order. Then $checkQulification()$ function in the smart contract will be called to verify the accounts authority and ensure the balance of elecoins and electricity adequate. Finally, $trade()$ function accomplishes the modification of corresponding accounts information, including available electricity as well as elecoins of prosumers and retailers. In Electricity Clearing Module, the modification of total generated electricity and total used electricity will be executed in $generation()$ function and $consumption()$ function, according to data from prosumers' smart meters. In Quire Module, we provide six interfaces for prosumers to inquire their accounts' information.

IV. Performance Evaluation and System Implementation

A. Parameter Setting

In the experiment, we consider the edge server's action space as $A_p \in [10, 19]$ and prosumers' action space as $A_x \in [20, 50]$. For parameters in DMA-UBRL algorithm, we set the maximum episode number as 2000. The learning rate is set as $\sigma = 0.8$. The discount factor $\gamma = 0.1$. In addition, we set the

Algorithm 2. Electricity Trading Smart Contract (ETSC).

Registration Module

System Initialization $\rightarrow$ $constructor()$

$administrator, totalElecoins,$

$unitPowerPrice, edgeServersAccounts.$

Prosumers Input $\rightarrow$ $empower()$

$userName, userAddr, genRight, transRight.$

$availPower = avaElecoins = 0.$

$totalGenPower = totalUsePower = 0.$

Transaction Module

for $i = 1, 2, 3, \cdots$ **do**

1. Buys elecoins amount x_i $\rightarrow$ $buyToken()$
2. Selects retailers.
3. Confirms quantity of electricity y_i.
4. Accounts verification $\rightarrow$ $checkQulification()$
5. Accounts clearing $\rightarrow$ $trade()$

Prosumer: $avaPower + y_i$, $Elecoins - y_i * Q$.

Retailer: $avaPower - y_i$, $Elecoins + y_i * Q$.

end for

Electricity Clearing Module

Smart Meter$\rightarrow$ $generation()$, $consumption()$

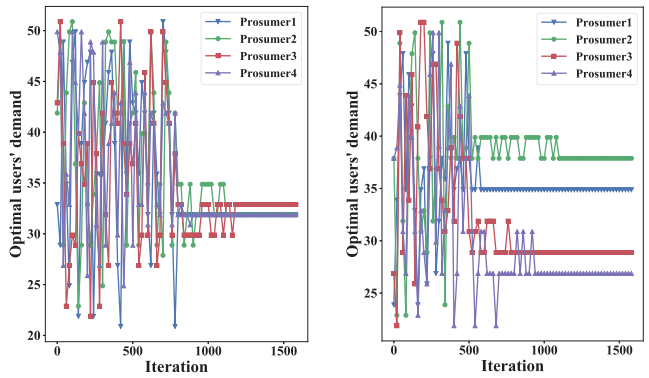
Quire Module

learning parameters $\epsilon^{win} = 0.0025$ and $\epsilon^{lose} = 0.01$ [11]. For parameters in the trading process, we consider the fixed block reward $Rc = 200$, the variable reward factor $r = 3$ and block size $s = 200$. The other default parameters are set as follows: $N = 8, M = 1, \lambda = 0.01, l = 1, k = 8, Q = 0.6, \eta = 0.3$.

B. Simulation Results

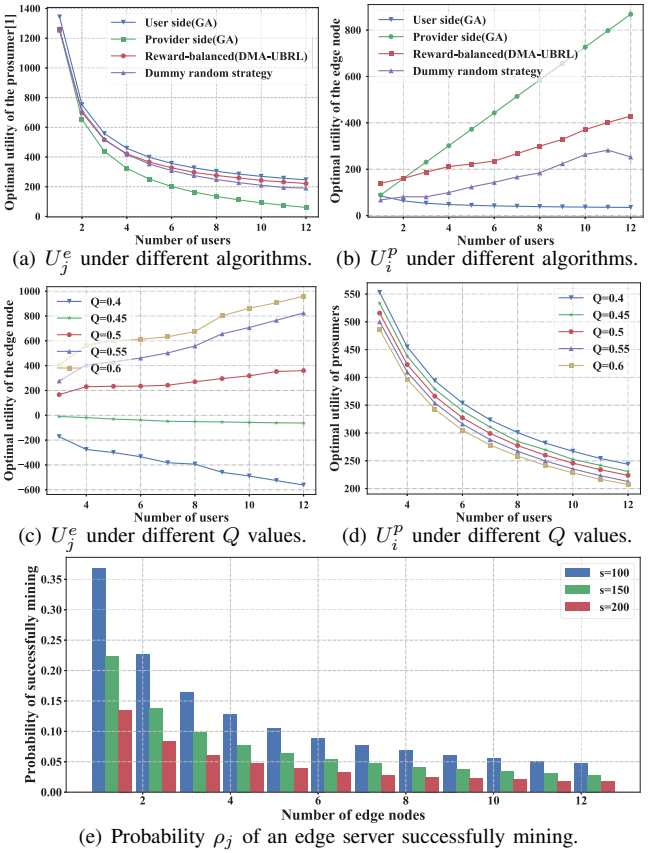
1) Convergence Performance: As shown in Fig. 3, we obtain the convergence performance of DMA-UBRL algorithm under the unified pricing mechanism and the discriminated pricing mechanism. To clearly present the tendency, we consider one edge server and four prosumers in this experiment. Under the unified pricing mechanism, prosumers' optimal demands are quite similar. The tiny distinction among prosumers is caused by diverse intentions for electricity service. In fact, edge server's pricing for unit elecoin obviously affects prosumers' optimal demands as shown in Fig. 3(b). The higher the price of unit elecoin is, the fewer demands of prosumers will be.

2) Economic Analysis: Fig. 4(a) and Fig. 4(b) present the balanced utility of prosumers and the leader edge server. As the number of prosumers increases, the red lines show the utilities tendency of both parties using DMA-UBRL algorithm. We use Genetic Algorithm (GA) to optimize prosumers' utility or the edge server's utility as comparing conditions, which are presented by blue lines and green lines respectively. The yellow lines represent dummy random policies of the edge server and prosumers. Whether considering prosumers' utility only or considering the edge server's utility only leads to poor performance of the opposite side. Our proposed multi-agent algorithm can reach the utility-balanced equilibrium, thereby jointly optimize the utilities of both parties. Besides, the increase of prosumers' quantity causes the decline of



(a) Unified pricing mechanism. (b) Discriminated pricing mechanism.

Fig. 3: Convergence performance of prosumers under different pricing mechanisms with DMA-UBRL algorithm.



(a) U_j^e under different algorithms. (b) U_i^p under different algorithms.

(c) U_j^e under different Q values. (d) U_i^p under different Q values.

(e) Probability ρ_j of an edge server successfully mining.

Fig. 4: Performance of the utility-balanced trading.

prosumers' utility. The reason is that multiple prosumers compete for elecoins from the same edge server, which rises the unit elecoin price rises and affects the quality of trading. On the other hand, as the number of prosumers rises, the utility of the edge server grows unsteadily, due to the fact that different prosumers have diverse intentions for elecoins and electricity.

In Fig. 4(c) and Fig. 4(d), the uniform unit electricity price Q between prosumers can also affect the utilities of the edge server and prosumers. For the edge server, within some range, the utility increases as the unit electricity price rises. The

reason is that the prosumers need to purchase more elecoins for their certain demands. Furthermore, different demands of each prosumer result in the unsmooth curve. Additionally, we find when $Q <= 0.45$, the utility will gradually decline. It is because the utilities of the edge server cannot offset the operating cost. On the other hand, prosumers' utility declines with the unit electricity price rising because prosumers purchase less electricity using an equivalent amount of elecoins.

In order to estimate the potential revenue of edge servers in the system, we present the probability of successfully mining under different block sizes in Fig. 4(e). As the number of edge servers increases, each edge server accounts for less computing power proportion to obtain the priority to record transactions, which means more difficult to successfully mine.

C. Realistic System Implementation

In order to demonstrate our system, we implement a prototype and demonstrate it with several shortcuts.

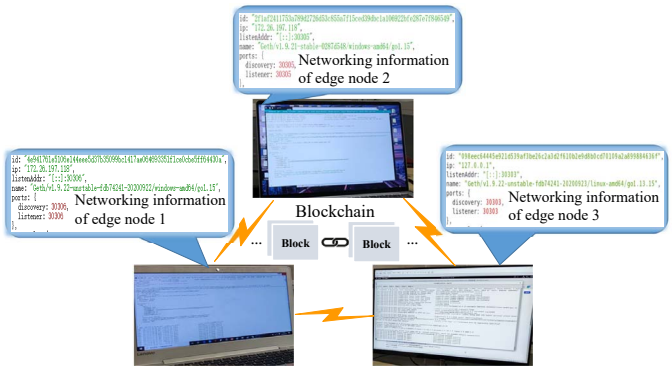


Fig. 5: Connection information of edge nodes.

1) Private Blockchain Construction: We build a local electricity blockchain based on three edge servers using Geth in Ethereum. As shown in Fig. 5, the devices in our implementation are I) A laptop with 8 GB RAM, Intel i7-7500U CPU. II) A laptop with 16 GB RAM, Intel i5-10210U CPU. III) A virtual machine with 2GB RAM and 30GB SCSI. IV) A router TP-LINK TL-WR842N providing wireless access.

2) Deployment and Presentation: Initially, we design ETSC using solidity language. In order to facilitate prosumers to conveniently operate their accounts and operate trading, a website interface[2] is designed for prosumers to call the function using MetaMask. In addition, the strict authentication of identity and calling mode have been set to prevent some malicious attacks caused by permission problems.

By using the comprehensive blockchain development framework Truffle, we compile and deploy our smart contract on the local blockchain simulator Ganache for security test. After that, we redeploy the smart contract on the build blockchain on Geth. Additionally, prosumers could interact with the private blockchain for information, such as the current block and transaction details, through the electricity blockchain explorer.

[2]**http://carolqiu.site/app/home.html**

V. Conclusion

In this article, we propose a distributed P2P electricity trading system aided by blockchain, which ensures the reliability between trading entities. In order to satisfy the balanced utilities for the edge server and prosumers, we formulate the elecoins purchasing problem as a hierarchical Stackelberg game. In this mechanism, the NE solution is given with the proof of existence and uniqueness. Then we design a DMA-UBRL algorithm to solve the Nash equilibrium. Finally, to further deal with privacy crisis and trading security, we factually design the blockchain-based electricity trading system using smart contract and blockchain explorer on Ethereum.

References

[1] M. B. Mollah, J. Zhao, D. Niyato, K.-Y. Lam, X. zhong Zhang, A. Ghias, L. Koh, and L. Yang, "Blockchain for future smart grid: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 8, pp. 18–43, 2021.

[2] M. Sabounchi and J. Wei, "Towards resilient networked microgrids: Blockchain-enabled peer-to-peer electricity trading mechanism," *2017 IEEE Conference on Energy Internet and Energy System Integration (EI2)*, pp. 1–5, 2017.

[3] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, pp. 840–852, 2018.

[4] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[5] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Transactions on Industrial Informatics*, vol. 13, pp. 3154–3164, 2017.

[6] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 3690–3700, 2018.

[7] R. Deng, Z. Yang, M. Chow, and J. Chen, "A survey on demand response in smart grids: Mathematical models and approaches," *IEEE Transactions on Industrial Informatics*, vol. 11, pp. 570–582, 2015.

[8] S. Maharjan, Q. Zhu, Y. Zhang, S. Gjessing, and T. Başar, "Dependable demand response management in the smart grid: A stackelberg game approach," *IEEE Transactions on Smart Grid*, vol. 4, pp. 120–132, 2013.

[9] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Optimal pricing-based edge computing resource management in mobile blockchain," *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2018.

[10] J. Lee, J. Guo, J. Choi, and M. Zukerman, "Distributed energy trading in microgrids: A game-theoretic model and its equilibrium analysis," *IEEE Transactions on Industrial Electronics*, vol. 62, pp. 3524–3533, 2015.

[11] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 15, pp. 3602–3609, 2019.

[12] B. Gu, X. Yang, Z. Lin, W. Hu, M. Alazab, and R. Kharel, "Multiagent actor-critic network-based incentive mechanism for mobile crowdsensing in industrial systems," *IEEE Transactions on Industrial Informatics*, vol. 17, pp. 6182–6191, 2021.

[13] N. Liu, L. He, X. Yu, and L. Ma, "Multiparty energy management for grid-connected microgrids with heat- and electricity-coupled demand response," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 1887–1897, 2018.