BC-MCSDT: A Blockchain-based Trusted Mobile Crowdsensing Data Trading Framework

Weiwei Hu[1], Bo Gu[1,2] Jinming Li[1], Zhen Qin[1]
[1]School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou 510275, China
[2]Guangdong Provincial Key Laboratory of Fire Science and Intelligent Emergency Technology, Guangzhou 510006, China
E-mail: huww3@mail2.sysu.edu.cn, gubo@mail.sysu.edu.cn, lijm89@mail2.sysu.edu.cn, qinzh23@mail2.sysu.edu.cn

2022 IEEE Symposium on Computers and Communications (ISCC) | 978-1-6654-9792-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISCC55528.2022.9912779

***Abstract*—Mobile crowdsensing (MCS) is a new sensing paradigm that relies on the crowd's sensing capabilities to aggregate data. Unlike traditional MCS systems, where sensing data are traded via a third-party platform, we propose a blockchain-based data trading framework to ensure the security of data transactions in the MCS system. In particular, the interactions between selling mobile users (SMUs) and buying mobile users (BMUs) are modeled as a Stackelberg game. Then, the optimal unit price and the amount of sensing time purchased from SMUs are solved by two smart contracts. Notably, the SMUs are compensated according to not only the amount of sensing time but also their historical reputation to encourage SMUs to contribute high-quality data. Furthermore, the blockchain technology guarantees that the reputations of each SMU are recorded in a traceable manner. Experimental results confirm that the proposed mechanism achieves near-optimal social welfare while protecting the security of data transaction.**

***Index Terms*—Mobile crowdsensing, blockchain, incentive mechanism, Stackelberg game**

I. Introduction

With the expansion of the Internet of Things (IoT), mobile crowdsensing (MCS) is considered as an efficient approach to enable fine-grained data collection over a large-scale area. In contrast to traditional sensing approaches, which require deploying a large number of sensors, MCS leverages the mobilities and sensing capabilities of the 'crowd' so that large-scale data can be gathered in a cost-efficient manner. Despite its advantages, several challenges remain to be addressed to implement a successful MCS system.

1) Lack of efficient incentive mechanisms. Mobile users who provide sensing data in MCS systems are called selling mobile users (SMUs). Due to the additional energy consumption and sensing efforts, self-interested SMUs are not willing to contribute high-quality sensing data or even participate in sensing activities unless they are fully compensated. Consequently, efficient incentive mechanisms are essential for high-quality data collection in MCS systems.

Recently, numerous studies have designed incentive mechanisms for quality-aware crowdsensing [1], [2]. In [1], Han $et\ al.$ proposed an ex-ante pricing scheme that posts a price to only the target SMUs to minimize the total payment within the restriction of data quality. However, they do not take into account SMU selfishness in the sense that each SMU aims to maximize its own benefits. Game theory is a powerful tool to analyze competitions between two parties with conflicting interests. Some researchers have applied game theory to design incentive mechanisms in MCS systems. In [3], Zhang $et\ al.$ used a Stackelberg game to analyze the conflicting interests between buying mobile users (BMUs) and SMUs and designed an incentive mechanism to obtain the unique Stackelberg equilibrium while simultaneously ensuring the benefits of the SMUs. Although game theory-based methods can address the selfishness issue, they require substantial amounts of information exchange (e.g., SMU's data quality, sensing cost, utility functions), which makes the signaling overhead unbearable.

2) Lack of accurate data quality evaluation mechanisms. The quality of sensing data provided by SMUs is highly dependent on their sensing ability and effort. Without an accurate data quality evaluation mechanism, SMUs may strategically use low-quality devices or reduce sensing effort to minimize their sensing cost. As a result, accurate data quality evaluation is of great significance for allocating sensing tasks to SMUs with high sensing ability and effort.

Existing works on data quality evaluation can be classified as instant quality-based methods and reputation-based methods. In particular, instant quality-based methods evaluate the sensing quality on the basis of the latest committed data. In [4], an expectation–maximization (EM) algorithm was considered to evaluate each SMU's real-time effort for sensing. However, deriving an objective evaluation of the quality of the latest committed data is usually time-consuming. To address this problem, reputation-based methods consider both the current and historical performance of SMUs. In [5], Xu $et\ al.$ obtained each SMU's reputation by considering the data quality in the task just completed and its historical performance. Nevertheless, the abovementioned methods evaluate the data quality of SMUs based on objective parameters such as utilities, which do not accurately reflect the true benefits of BMUs in pratical situations.

3) Lack of distributed data trading frameworks. A typical MCS system relies on a centralized sensing platform (CSP) to aggregate and analyze the sensing data. Such a centralized architecture is vulnerable to distributed denial-of-service (DDoS) attacks [6]. Furthermore, the CSP, which acts

This work was supported in part by the National Science Foundation of China (NSFC) under Grand U20A20175 and in part by the National Key R&D Program of China under Grant 2020YFB1713800. *(Corresponding author: Bo Gu.)*

as a broker between BMUs and SMUs, is not fully trusted. For example, malicious CSPs may steal or tamper with the sensing data transmitted between BMUs and SMUs through man-in-the-middle attacks [7]. Therefore, it is urgent to design a distributed and trusted framework for MCS systems.

Fortunately, the emergence of blockchain, a distributed ledger technology, enables transactions among untrusted network entities. Recently, blockchain has been extensively applied in many fields, for instance, electricity trading, edge computing and cloud computing [8]–[10]. In [8], Kang *et al.* proposed an electricity trading framework based on blockchain that enables decentralized electricity trading between electric vehicles. In [9], Guo *et al.* applied blockchain to edge computing and realized trusted authentication for IoT devices. In [10], a novel blockchain-based mobile device cloud framework was proposed to manage user registration, task release, penalizing and rewarding. Only a few works have focused on integrating blockchain into MCS to enable distributed data transaction.

To address the abovementioned challenges, in this paper, we first propose a decentralized architecture that investigates the potential of consortium blockchain to ensure the security and privacy of data transactions in a MCS system. Then, we formulate the task allocation and data pricing as a Stackelberg game to incentivize SMUs to provide high-quality sensing data. We elaborate two autoexecuting smart contracts (SCs): optimization-based task allocation (OBTA) deployed on the SBSs and deep reinforcement learning-based data pricing (DRLBDP) deployed on the SMUs. Furthermore, we design a reputation evaluation mechanism: the reputations of SMUs are updated based on the data quality evaluated by BMUs and recorded in the blockchain to prevent malicious tampering. Our contributions can be summarized as follows:

- Novel Blockchain-Based MCS Framework: We develop a blockchain-based mobile crowdsensing data trading (BC-MCSDT) architecture that enables distributed data transactions between SMUs and BMUs. The block structure of consortium blockchain is elaborated to implement traceable and tamper-resistant data storage.
- Game-Theoretical Analysis: The interactions between BMUs and SMUs, as well as the competition among SMUs, are modeled as a Stackelberg game, while the SBSs are used to replace the BMUs for gaming. The Nash equilibrium is then derived by iteratively solving a convex optimization problem and learning a pricing policy.
- OBTA: The OBTA encourages SMUs to provide high-quality sensing data by allocating tasks to SMUs according to their reputations. Then, the reputation of each SMU is recorded in the consortium blockchain to prevent malicious tempering.
- DRLBDP: The DRLBDP guides SMUs to learn a good pricing policy while interacting with the environment. Intuitively, the SMU can increase its price if the price is competitive in the previous epoch and vice versa. Notably, each SMU can determine its price without requiring any price information about other SMUs. The signaling overhead can therefore be reduced substantially.

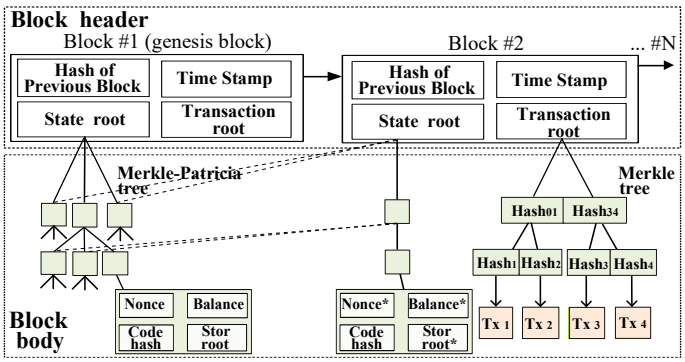


Fig. 1: Blockchain structure for BC-MCSDT

II. System Model

A. System Architecture

There are three participants in our BC-MCSDT system: SMUs, SBSs and BMUs.

1) SMUs are the data producers. Upon receiving a task request, each SMU executes the DRLBDP to calculate its selling price, performs the sensing task and obtains the corresponding reward.

2) SBSs are the authorized nodes. Each SBS is equipped with communication, storage and computation resources and is mainly responsible for i) forwarding task requests to SMUs; ii) collecting the selling price from each SMU and executing the OBTA to determine the amount of sensing time to purchase from each SMU; and iii) validating and recording transactions into the blockchain.

3) BMUs are the sensing task initiators. Each BMU sends a request by indicating its monetary budget and the type of data to be collected. Upon receiving the sensing data, each BMU evaluates the data qualities of SMUs and sends the evaluations to the nearest SBS.

B. Blockchain Architecture

Fig.1 shows the blockchain structure of the BC-MCSDT system. Each block in the blockchain is formed from a block header and a block body. The block header records the abstract data of a block, including the hash of the previous block, time stamp, state root and transaction root. Concretely, with the exception of the genesis block, each block records the hash of the previous block so that all blocks in the blockchain are connected sequentially. The time stamp is a characteristic of the block generation time. The state root and transaction root are the root hashes of the state tree and transaction tree, respectively. The block body is organized as a state tree and a transaction tree. The state tree is a Merkle-Patricia tree [11] that records the account information as key-value pairs. The account data include the balance, codehash (i.e., hash value of SCs), nonce (i.e., number of SCs executions) and storage root (i.e., the root of contract data). The codehashes of SMU accounts and SBS accounts are the hashes of DRLBDP and OBTA, respectively. The transaction tree is a Merkle tree [11] that records the hashes of transactions over a period of time.

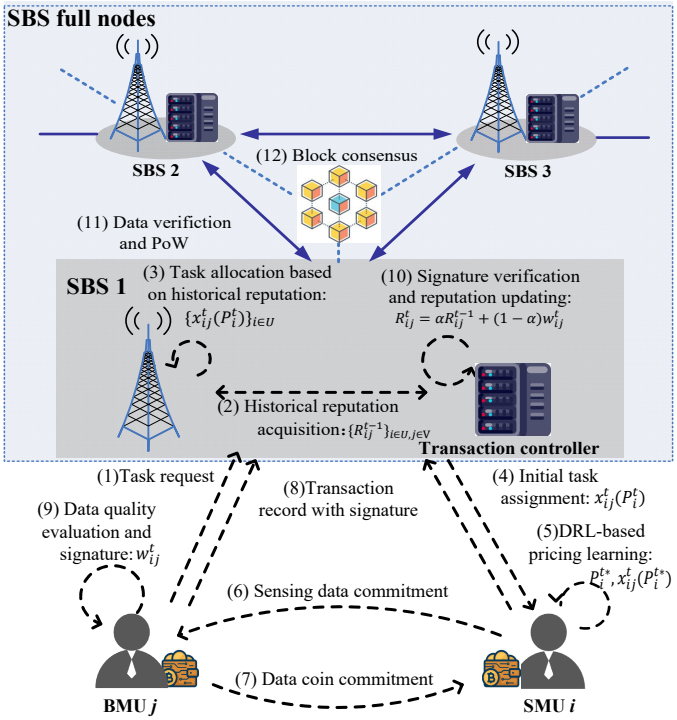


Fig. 2: The Sensing Process of BC-MCSDT

All the transaction data are stored in the local of SMUs and BMUs.

In blockchain systems, the transaction data can only be recorded in blocks after being verified by multiple parties, which is called block consensus. Unlike traditional blockchain, where all nodes participate in reaching the consensus, consortium blockchain categorizes all nodes into full nodes and light nodes, and only full nodes are responsible for block consensus. In this paper, SBSs and BMUs act as full nodes and light nodes, respectively. The full nodes maintain the complete blockchain and are responsible for collecting, auditing and storing transaction data, while light nodes store only a blockheader chain and are responsible for verifying transaction data. In this was, the consensus efficiency of generating a new block can be improved substantially, and the storage cost of light nodes can be reduced.

III. Problem Formulation

We consider a BC-MCSDT system where the sensing time is divided into time slots. In time slot t, the numbers of SMUs and BMUs are expressed as U^t and V^t, respectively. Additionally, the set of SMUs is denoted by $\mathcal{U}^t = \{1, 2, \ldots, U^t\}$, and the set of BMUs is denoted by $\mathcal{V}^t = \{1, 2, \ldots, V^t\}$. Formally, each SMU $i \in \mathcal{U}^t$ performs sensing tasks within the constraints of the limited available time budget τ_i^t and each BMU $j \in \mathcal{V}^t$ pays for SMUs within the constraints of a limited available monetary budget β_j^t. To conserve the data security and privacy of each MU (i.e., SMU i or BMU j), we use an elliptic curve encryption algorithm for system initialization. First, each MU generates a local account {i.e., pk, sk, $Waddr$} and sends the information {i.e., pk, $Waddr$} to the nearest SBS for registration. $Waddr$ is the wallet address used to access the data coin account, and pk represents the public key, which is unidirectionally generated by the secret key sk via the elliptic encryption algorithm. The SBS uploads account information to the consortium blockchain. $Waddrs$ and pks of other participants can be obtained by requesting the nearest SBS. Fig. 2 demonstrates the sensing process of the BC-MSCDT system.

Data Request and Task Completion (Step 1-4): First, BMU j initiates a task request to the nearest SBS, including the plaintext u_j^t (i.e., PoIs and monetary budget β_j^t) and signature after hash $Sign_{sk_j}(H(u_j^t))$. The SBS decrypts and judges the request according to the result $Vrfy_{pk_j}(u_j^t, Sign_{sk_j}(H(u_j^t)))$. If the request is verified, the SBS records the u_j^t to the transaction controller and broadcasts the request to other SBSs for verification. Based on the monetary budget β_j^t of BMU j, time budget τ_i of SMU i and accumulated reputation R_{ij}^{t-1}, the smart contract OBTA automatically calculates the allocation function $x_{ij}^t(p_i^t)$.

Data Coin's Paying and Earning (Step 5-8): SMU i observes the historical allocation and learns the biding price p_i^{t*} according to the DRLBDP. Furthermore, SMU i sends the sensing data encrypted with pk_j to BMU j. After receiving the sensing data, BMU j checks the transaction and pays for the data coin. The data coin is transfered from the $Waddr_j$ of BMU j to the $Waddr_i$ of SMU i according to the given data quantity x_{ij}^t and the unit price p_i^{t*}. Both BMU j and SMU i confirm the transaction and generate transaction records and then sign and send the records to SBSs for verification.

Data Evaluation and Reputation Update (Step 9-10): BMU j evaluates the task completion quality w_{ij}^t and then sends plaintext w_{ij}^t and signature $Sign_{sk_j}(H(w_j^t))$ to the nearest SBS. The verified w_{ij}^t is used to update the reputation R_{ij}^t of SMU i for task j. The data of TA-SC (i.e., $\{\tau_i^t, \beta_j^t, p_i^{t*}, w_{ij}^t, R_{ij}^t\}$) are stored in the storage roots of SBS accounts, while the data of TA-SC (i.e., $\{p_i^{t*}, x_{ij}^t\}$) are stored in the storage roots of SMU accounts.

Data verification and Proof-of-Work (Step 11): Each SBS initiates a verification request with the transaction data and its signature to other SBSs. Then, the transaction records are structured into blocks and continue to spread to the network. Once all the transaction records are authenticated, each SBS spends its effort to search for a valid proof-of-work (PoW) solution in the form of $H(param + nonce) < target$ [12], where $param$ represents the data related to block information, $nonce$ denotes a random nonce, and $target$ indicates the target difficulty, which is adjusted by finding a specific $nonce$.

Implementation of Block Consensus (Step 12): The first SBS with a valid PoW (i.e.,$nonce$) becomes the leader of block consensus. The leader packs and broadcasts transaction data to other SBSs (i.e., followers) for verification. Then, each follower broadcasts signed audited reports to other followers for mutual signature review. After completing the data audit and signature review processes, the followers generate and send audit reports to the leader, including data audit records, mutual audit records and signature records. Finally, the leader calculates statistics on the audit reports and sends them to all

followers for storage on the consortium blockchain.

IV. Stackelberg Game

A Stackelberg game is a noncooperative game that includes a leader and several followers, and each player attempts to maximize their own benefits. In general, a leader gives its strategy first; then, followers give the optimal response by observing the actions of the leader. In this paper, the responses of the leader and each follower are derived according to the OBTA and DRLBDP, respectively. Generally, a Stackelberg game consists of the following three parts:

- **Participants**: SBSs and SMUs are the participants, where the SBS is the leader and SMUs are the followers.
- **Strategies**: The strategy of SMUs involves determining the price of a unit quantity of their sensing time; the strategy of the SBSs involves choosing the quantity of sensing time to purchase from SMUs.
- **Payoff**: The payoff functions of SMUs and SBSs correspond to Eqs.(2) and (4), respectively.

The reputation of each SMU i is updated according to its historical data quality; at time slot $t-1$, it is defined as follows

$$R_{ij}^{t-1} = \alpha R_{ij}^{t-2} + (1-\alpha)\omega_{ij}^{t-1} \tag{1}$$

where α is the learning rate of reputation and ω_{ij}^{t-1} is the data quality evaluation of SMU i for sensing task j.

At time slot t, for each task j, the amount of sensing time purchased from each SMU i is a function of its bidding price, denoted by $x_{ij}^t\left(p_i^t\right)$. The payoff of all SMUs is given by

$$\psi\left(\boldsymbol{p}^t, \boldsymbol{x}^t\right) = \sum_{i\in\mathcal{U}^t} \psi_i\left(p_i^t, \boldsymbol{x}_i^t\right) = \sum_{i\in\mathcal{U}^t} p_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(p_i^t\right) \tag{2}$$

where $\boldsymbol{p}^t = \{p_i^t\}_{i\in\mathcal{U}^t}$ and p_i^t is the unit price of SMU i, $\boldsymbol{x} = \{\boldsymbol{x}_i^t\}_{i\in\mathcal{U}^t}$ and $\boldsymbol{x}_i^t = \{x_{ij}^t\}_{j\in\mathcal{V}^t}$.

The payoff of all BMUs is given by

$$\begin{aligned}\phi\left(\boldsymbol{x}^t, \boldsymbol{p}^t\right) &= \sum_{j\in\mathcal{V}^t} \phi_j\left(\boldsymbol{x}_j^t, \boldsymbol{p}^t\right) \\ &= \sum_{j\in\mathcal{V}^t} \log\left(\sum_{i\in\mathcal{U}^t} R_{ij}^{t-1} x_{ij}^t\right) - \sum_{i\in\mathcal{U}^t} p_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(p_i^t\right)\end{aligned} \tag{3}$$

where $\boldsymbol{x}_j^t = \{x_{ij}^t\}_{i\in\mathcal{U}^t}$, and $\log(\cdot)$ is a monotonically increasing utility function of the reputation-weighted sensing time.

The social welfare (i.e., sum payoff of SBSs), which is the sum payoff of all BMUs and all SMUs, is given by

$$\begin{aligned}\phi\left(\boldsymbol{x}^t, \boldsymbol{p}^t\right) &= \psi\left(\boldsymbol{p}^t, \boldsymbol{x}^t\right) + \phi\left(\boldsymbol{x}^t, \boldsymbol{p}^t\right) \\ &= \sum_{j\in\mathcal{V}^t} \log\left(\sum_{i\in\mathcal{U}^t} R_{ij}^{t-1} x_{ij}^t\left(p_i^t\right)\right)\end{aligned} \tag{4}$$

Definition 1: Letting $\boldsymbol{x}_i^{t*} = \{x_{ij}^{t}{}^*\}_{j\in\mathcal{V}^t}$, $\boldsymbol{x}^{t*} = \{x_{ij}^{t}{}^*\}_{i\in\mathcal{U}^t, j\in\mathcal{V}^t}$ and $\boldsymbol{p}^{t*} = \{p_i^{t*}\}_{i\in\mathcal{U}^t}$, the point $(\boldsymbol{x}^{t*}, \boldsymbol{p}^{t*})$ is a Nash equilibrium if the following equations are satisfied

$$\psi_i\left(p_i^{t*}, \boldsymbol{x}_i^{t*}\right) \geq \psi_i\left(p_i^t, \boldsymbol{x}_i^{t*}\right), \forall p_i^t \neq p_i^{t*}, \forall i \in \mathcal{U}^t \tag{5}$$

and

$$\phi\left(\boldsymbol{x}^{t*}, \boldsymbol{p}^{t*}\right) \geq \phi\left(\boldsymbol{x}^t, \boldsymbol{p}^{t*}\right), \forall \boldsymbol{x}^t \neq \boldsymbol{x}^{t*} \tag{6}$$

V. OBTA-Based Social Welfare Optimization

In this section, we show how the OBTA works to determine the quantity of sensing time to purchase from SMUs. Given a unit price $\boldsymbol{p}^t = \{p_i\}_{i\in\mathcal{U}^t}$, the OBTA is elaborated to optimize the social welfare

$$\begin{aligned}\max_{\boldsymbol{x}} \quad & \sum_{j\in\mathcal{V}^t} \log\left(\sum_{i\in\mathcal{U}^t} R_{ij}^{t-1} x_{ij}^t\right) \\ \text{s.t.} \quad & \sum_{j\in\mathcal{V}^t} x_{ij}^t \leq \tau_i^t, \ \forall i \in \mathcal{U}^t \\ & \sum_{i\in\mathcal{U}^t} p_i^t x_{ij}^t \leq \beta_j^t, \ \forall j \in \mathcal{V}^t\end{aligned} \tag{7}$$

Theorem 1. *According to* [13], the optimal solution to OBTA can be expressed as follows

$$x_{ij}^t\left(\boldsymbol{p}^t\right) = \begin{cases} \frac{\beta_j^t R_{ij}^{t-1}}{p_i^t \sum_{i\in\mathcal{U}^t} R_{ij}^{t-1}}, & \text{if I} \geq 0 \\ \frac{\tau_i^t R_{ij}^{t-1}}{\sum_{j\in\mathcal{V}^t} R_{ij}^{t-1}}, & \text{otherwise} \end{cases} \quad \forall i \in \mathcal{U}^t, \forall j \in \mathcal{V}^t \tag{8}$$

where $I = \sum_{j\in\mathcal{V}^t} \log\left(\sum_{i\in\mathcal{U}^t} \frac{\beta_j^t R_{ij}^{t-1}{}^2}{p_i^t \sum_{i\in\mathcal{U}^t} R_{ij}^{t-1}}\right) - \sum_{j\in\mathcal{V}^t} \log\left(\sum_{i\in\mathcal{U}^t} \frac{\tau_i^t R_{ij}^{t-1}{}^2}{\sum_{j\in\mathcal{V}^t} R_{ij}^{t-1}}\right)$.

VI. DRLBDP-Based SMU Payoff Optimization

This section explains how the DRLBDP works to obtain the best response of SMUs for the strategy taken by the OBTA.

A. Overview of Reinforcement Learning

In RL, each agent can take appropriate actions based on the current state and trial-and-error experience, with the aim of maximizing the long-term payoff. Specifically, at time slot t, the agent first determines an action a_t by observing the current environment state s_t. Next, the agent receives a reward r_{t+1}, and the environment transitions to state s_{t+1}. Consequently, multiple experience tuples $\{(s_t, a_t, r_{t+1}, s_{t+1}), ...\}$ are generated by interacting with the environment. Then, the discounted rewards can be obtained by the agent updating their policy [14].

As a classic policy-based algorithm, the policy gradient (PG) algorithms select actions by maximizing an objective function. The PG of a policy can be presented as

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = E_{s\sim p^\pi, a\sim\pi_{\boldsymbol{\theta}}}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s) Q^\pi(s,a)]. \tag{9}$$

On this basis, the deterministic policy gradient (DPG) algorithm [15] is transformed into a deterministic policy $\mu_{\boldsymbol{\theta}}$, in which the gradient can be expressed as

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = E_{s\sim\mathcal{D}}[\nabla_{\boldsymbol{\theta}} \mu_{\boldsymbol{\theta}}(a|s) \nabla_a Q^\mu(s,a)|_{a=\mu_{\boldsymbol{\theta}}(s)}] \tag{10}$$

The deep deterministic policy gradient (DDPG) [16], which combines DPG and deep learning, includes two networks (i.e., actor and critic). Specifically, the actor network decides the deterministic action a given a state s, and the critic network obtains the estimated Q-value based on state s and action a.

Algorithm 1 Training process of DRLBDP

Initialize:

Initialize critic networks $\{Q_i(\boldsymbol{o}, \boldsymbol{a}|\boldsymbol{\theta}^{Q_i})\}_{i\in\mathcal{U}}$ and actor networks $\{\mu_i(o|\boldsymbol{\theta}^{\mu_i})\}_{i\in\mathcal{U}}$ with parameters $\{\boldsymbol{\theta}^{Q_i}\}_{i\in\mathcal{U}}$ and $\{\boldsymbol{\theta}^{\mu_i}\}_{i\in\mathcal{U}}$

Initialize target networks $\{Q_i'\}_{i\in\mathcal{U}}$ and $\{\mu_i'\}_{i\in\mathcal{U}}$ with parameters $\boldsymbol{\theta}^{Q_i} \leftarrow \theta^{Q_i}$, $\boldsymbol{\theta}^{\mu_i} \leftarrow \boldsymbol{\theta}^{\mu_i}$

Initialize the experience replay buffer $\mathcal{D}$

for each episode t **do**

Receive initial observation state from the nearest SBS.

for each agent $i \in \mathcal{U}^t$ **do**

Select an action a_i^t according to Eq.(13);

Execute the action a_i^t, then observe reward r_i^t and next state o_i^{t+1} from the nearest SBS;

Store this experience $\left(o_i^t, a_i^t, r_i^t, o_i^{t+1}\right)$ in $\mathcal{D}$;

end for

for each agent $i \in \mathcal{U}^t$ **do**

Sample a minibatch of experience from $\mathcal{D}$;

Update parameters $Q_i(\boldsymbol{o}, \boldsymbol{a}|\boldsymbol{\theta}^{Q_i})$ based on Eq.(15);

Update parameters $\mu_i(o|\boldsymbol{\theta}^{\mu_i})$ according to Eq.(17);

Update the target networks's parameters:

$$\boldsymbol{\theta}^{Q_i'} \leftarrow \tau\boldsymbol{\theta}^{Q_i} + (1-\tau)\boldsymbol{\theta}^{Q_i'}$$

$$\boldsymbol{\theta}^{\mu_i'} \leftarrow \tau\boldsymbol{\theta}^{\mu_i} + (1-\tau)\boldsymbol{\theta}^{\mu_i'}$$

end for

end for

In a multiagent scenario, constant changes in each agent's policy can lead to an unstable environment, so the strategies learned for each agent in this environment are meaningless. Therefore, a multiagent DDPG (MADDPG) [17] algorithm, which integrates the DDPG algorithm into a multiagent architecture, is proposed with the following advantage:

- **Centralized training, distributed execution**: Data transaction experiences are shared among agents to train the critic and actor in the training process, while each agent uses only its local observations to output the action in the execution process.

B. Proposed Smart Contract DRLBDP

We propose a smart contract DRLBDP to determine the optimal price for the agents according to the multiagent DDPG algorithm. At time slot t, each SMU i observes the task assignment results from the environment and then chooses a unit price to maximize its immediate benefits.

$$\begin{aligned} &\max_{p_i^t} \; p_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(\boldsymbol{p}^t\right) - c_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(\boldsymbol{p}^t\right) \\ &\text{s.t.} \quad p_i^t \geq 0 \end{aligned} \tag{11}$$

where c_i^t is SMU i's unit sensing cost.

State: We consider the previous L time slots to provide the agent better opportunities to learn the changes in the environment. At time slot t, the state feature space observed by SMU i is

$$o_i^t = \{p_i^{t-1}, \{x_{ij}^{t-1}\}_{j\in\mathcal{V}^{t-1}}, \cdots, p_i^{t-L}, \{x_{ij}^{t-L}\}_{j\in\mathcal{V}^{t-L}}\}, \forall i \in \mathcal{U}^t \tag{12}$$

Action: After observing the state features, each SMU i determines the unit price (i.e., p_i^t), and the action of the output for each SMU i is calculated as

$$a_i^t = \mu_i\left(o_i^t|\boldsymbol{\theta}^{\mu_i}\right) \tag{13}$$

Reward: At time slot t, the reward function of SMU i is defined as

$$r_i^t = \; p_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(\boldsymbol{p}^t\right) - c_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(\boldsymbol{p}^t\right) \tag{14}$$

where SMU i receives a penalty when the sensing cost exceeds the payment received (i.e., $c_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(\boldsymbol{p}^t\right) > p_i^t \sum_{j\in\mathcal{V}^t} x_{ij}^t\left(\boldsymbol{p}^t\right)$).

The smart contract DRLBDP is composed of a critic network and an actor network. In the actor network, the input is the observation of the state features space o_i^t, and the output is the action a_i^t. In the critic network, the inputs are the state features space o_i^t and action a_i^t, and the output is the estimated value of the current state.

Critic Network: The loss function of the critic network can be expressed as

$$\mathcal{L}_i = \frac{1}{T}\sum_t^T (y_i^t - Q_i(\boldsymbol{o}^t, \boldsymbol{a}^t|\boldsymbol{\theta}^{Q_i}))^2 \tag{15}$$

where $\boldsymbol{a}^t = \{a_i^t\}_{i\in\mathcal{U}^t}$, $\boldsymbol{o}^t = \{o_i^t\}_{i\in\mathcal{U}^t}$, and y_i^t is the value of the target action, which can be expressed as

$$y_i^t = r_i^t + \gamma Q_i'(\boldsymbol{o}^{t+1}, \boldsymbol{a}'^{(t+1)}|\boldsymbol{\theta}^{Q_i'}) \tag{16}$$

Actor Network: The expected reward gradient of SMU i with deterministic policies $\mu^{\boldsymbol{\theta}_i}$ can be presented as

$$\nabla_{\boldsymbol{\theta}^{\mu_i}} J \approx \frac{1}{T}\sum_t^T \nabla_{\boldsymbol{\theta}^{\mu_i}} a_i \nabla_{a_i} Q_i(\boldsymbol{o}^t, \boldsymbol{a}^t|\boldsymbol{\theta}^{Q_i})|_{a_i=\mu_i(o_i^j|\boldsymbol{\theta}^{\mu_i})} \tag{17}$$

where $\boldsymbol{a}_{N\setminus i}^t = \{a_1^t, \cdots, a_{i-1}^t, a_{i+1}^t, \cdots, a_N^t\}$.

Algorithm 1 illustrates the implementation of DRLBDP.

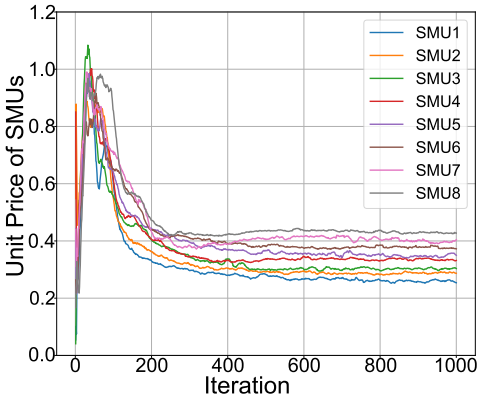
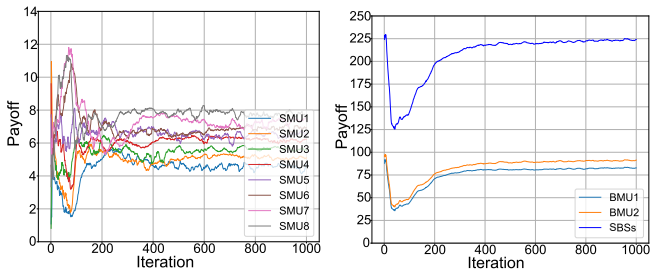


Fig. 3: Unit Prices of SMUs ($\beta_j = 20, \tau_i = 25$)



(a) (b)

Fig. 4: Payoff ($\beta_j = 20, \tau_i = 25$). (a) Payoff of SMUs. (b) Payoff of BMUs and SBSs.

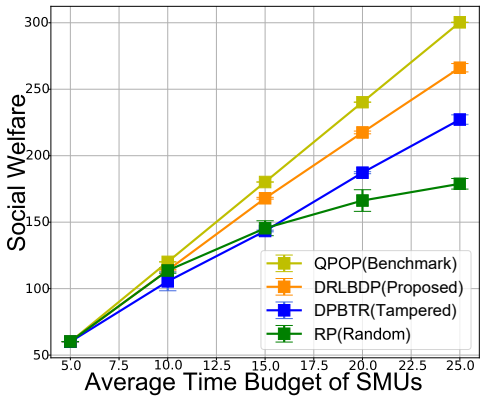


Fig. 5: Different time budgets of SMUs ($\beta_j = 20$)

VII. Numerical Evaluation

A. Simulation Setup

We consider a BC-MCSDT system consisting of 2 BMUs and 8 SMUs. The problem of payoff maximization is restrained by the monetary budget of the BMUs (i.e., $\beta_j \in$[20,25]) and the time budget of the SMUs (i.e., $\tau_i \in$[5,25]) simultaneously. For simplicity, the average data qualities (ADQs) provided by the SMUs are set to $\{\omega_{i1} = 1 + i/10\}_{i\in\mathcal{U}}$ and $\{\omega_{i2} = 1.1 + i/10\}_{i\in\mathcal{U}}$. The cost of each SMU for unit sensing time is set to 0.05. The reputations of the SMUs are determined by their data qualities. We set α to 0.9 and the initial reputation equal to ADQ. To explore the impact of the data quality dispersion on DRLBDP, we consider a standard deviation (i.e., $\sigma = 0.05$). The aim of self-interested SMUs is to maximize the immediate payoff, so we set the discount factor γ of long-term payoff to 0.

B. Behaviors of Proposed Algorithm

We first assess the convergence of the smart contract DRLBDP. Fig. 3 illustrates the unit prices of SMUs versus the number of iterations. The number of iterations required for convergence is approximately 400, which is acceptable. Furthermore, the agents can effectively learn to match their prices and data qualities. This result is not surprising because once the price does not match quality, the agents are penalized according to the ratio of their mismatch.

Fig. 4(a) illustrates the payoff of SMUs versus the number of iterations. Initially, SMUs of different data qualities consistently increase unit prices to obtain higher payoff, which leads to substantial differences in terms of the payoff among SMUs, as shown in Fig. 3. Then, the payoff of each SMU converges with the convergence of the unit price, as shown in Fig. 4(a).

Fig. 4(b) illustrates the payoff of BMUs and SBSs versus the number of iterations. SBSs and BMUs regain relatively stable payoffs after the initial sharp fluctuation due to the competition among SMUs.

C. Comparison with Other Algorithms

We compare the DRLBDP with three algorithms: average quality proportional optimal pricing (AQPOP), data pricing based on tampered reputation (DPBTR), random price (RP).

- **AQPOP**: SBSs have the ADQ knowledge of SMUs and that the unit price of each SMU is proportional to its ADQ. Furthermore, the monetary budget of BMUs and the time budget of SMUs are depleted at the same time.
- **DPBTR**: The reputation of SMU 1 and SMU 8 are tampered to $R_{1j}^{t\,\prime} = R_{1j}^{t} + 1$ and $R_{8j}^{t\,\prime} = R_{8j}^{t} - 1$.
- **RP**: The unit prices of each SMU are randomly generated within $(0, 1)$.

Fig. 5 compares the four algorithms with different time budget range of each SMU. First, compared with QPOP, the DRLBDP achieves near-optimal performance in the case of data quality fluctuations and a lack of prior knowledge of ADQs. In addition, DRLBDP outperforms DPBTR in terms of social welfare. That is because the amount of sensing time mismatches the real data quality in DPBTR, which leads to a loss of BMUs' benefits. Furthermore, DRLBDP outperforms RP regarding social welfare given an adequate average time budget. RP cannot make full use of SMUs' sensing resources, but our DRLBDP encourages each SMU to dynamically adjust its price to realize a resource-efficient BC-MCSDT system.

VIII. Conclusion

In this paper, we first present a BC-MCSDT system to introduce a trusted decentralized task allocation framework of MCS. Then, we design a reputation-based incentive mechanism to encourage SMUs to contribute high-quality sensing data. Without the prior knowledge of SMUs perceived data and the actions simultaneously taken by other SMUs, the proposed algorithm achieves a resource-efficient BC-MCSDT system. Furthermore, we compare the proposed algorithm with other three algorithms. Simulation results show that our algorithm achieves a near-optimal social welfare while protecting the security of data transaction.

References

[1] K. Han, H. Huang, and J. Luo, "Quality-aware pricing for mobile crowdsensing," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1728–1741, 2018.

[2] X. Tian, W. Zhang, Y. Yang *et al.*, "Toward a quality-aware online pricing mechanism for crowdsensed wireless fingerprints," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 5953–5964, 2018.

[3] L. Zhang, J. Tan, Y.-C. Liang *et al.*, "Deep reinforcement learning for modulation and coding scheme selection in cognitive hetnets," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[4] D. Peng, F. Wu, and G. Chen, "Data quality guided incentive mechanism design for crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 2, pp. 307–319, 2017.
[5] C. Xu, Y. Si, L. Zhu *et al.*, "Pay as how you behave: A truthful incentive mechanism for mobile crowdsensing," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 053–10 063, 2019.
[6] A. Yaar, A. Perrig, and D. Song, "Stackpi: New packet marking and filtering mechanisms for ddos and ip spoofing defense," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1853–1863, 2006.
[7] M. Arafeh, M. El Barachi, A. Mourad *et al.*, "A blockchain based architecture for the detection of fake sensing in mobile crowdsensing," in *2019 4th International Conference on Smart and Sustainable Technologies (SpliTech)*, 2019, pp. 1–6.
[8] J. Kang, R. Yu, X. Huang *et al.*, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3154–3164, 2017.
[9] S. Guo, X. Hu, S. Guo *et al.*, "Blockchain meets edge computing: A distributed and trusted authentication system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1972–1983, 2020.
[10] M. Wang, C. Xu, X. Chen *et al.*, "Bc-mobile device cloud: A blockchain-based decentralized truthful framework for mobile device cloud," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1208–1219, 2021.
[11] J. Zhu, Q. Li, C. Wang *et al.*, "Enabling generic, verifiable, and secure data search in cloud services," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 8, pp. 1721–1735, 2018.
[12] M. Keshk, B. Turnbull, N. Moustafa *et al.*, "A privacy-preserving-framework-based blockchain and deep learning for protecting smart power networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5110–5118, 2020.
[13] 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, no. 9, pp. 6182–6191, 2021.
[14] B. Gu, M. Alazab, Z. Lin, X. Zhang, and J. Huang, "Ai-enabled task offloading for improving quality of computational experience in ultra dense networks," *ACM Trans. Internet Technol.*, vol. 22, no. 3, mar 2022. [Online]. Available: https://doi.org/10.1145/3491217
[15] Volodymyr, Mnih, Koray *et al.*, "Human-level control through deep reinforcement learning." *Nature*, 2015.
[16] Z. Gu, C. She, W. Hardjawana *et al.*, "Knowledge-assisted deep reinforcement learning in 5g scheduler design: From theoretical framework to implementation," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 7, pp. 2014–2028, 2021.
[17] H. Peng and X. Shen, "Multi-agent reinforcement learning based resource management in mec- and uav-assisted vehicular networks," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 131–141, 2021.