Trustworthy and Efficient Crowdsensed Data Trading on Sharding Blockchain

En Wang[1], Jiatong Cai[1], Yongjian Yang[1], Wenbin Liu[1], Hengzhi Wang[1], Bo Yang[1*], Jie Wu[2], *Fellow, IEEE*

***Abstract*—With the development of communications, networking, and information technology, Crowdsensed Data Trading (CDT) becomes a novel data trading paradigm. In CDT, the data requesters publish crowdsensing tasks with specific data requirements, and then workers complete these tasks, upload the data and obtain corresponding rewards. To efficiently deal with data trading, most of the existing CDT systems assume a trusted centralized platform. However, we argue that the platform may collude with workers or requesters to trick others for achieving more benefits. For example, according to the workers' uploaded data, the platform can modify the reward functions by colluding with the requester. Similarly, the platform might collude with workers to let them know the reward function, then workers could forge data. Meanwhile, requesters and workers may also be malicious. For example, requesters may post tasks but fail to pay and workers can upload wrong data to mislead the system. To solve the above problems, we combine the Crowdsensed Data Trading system with intelligent Blockchain (CDT-B), which contains a smart contract called CDToken. As a credible third-party, the CDToken is used to record the requesters' reward function and workers' data uploading function to avoid targeted trick. At the same time, we not only design a Data Uploading and Preprocessing (DUP) mechanism in CDToken to collect and process the workers' sensed data, but also propose a Grouping Truth Discovery (GTD) to evaluate their data quality for determining the payments. Moreover, to hold a large number of requesters and workers in CDT-B, we propose a Layered Sharding blockchain based on Membership Degree (LSMD) to solve the blockchain inefficiency problem. Finally, we deploy CDToken to an experimental environment based on Ethereum and demonstrate its efficient performance and practicability.**

***Index Terms*—Intelligent blockchain, crowdsensed data trading, truth discovery, sharding**

I. Introduction

WITH the rapid development of future communications, networking, and information technology, people's demand for data is increasing. In recent years, many data trading systems have emerged, such as Qlik, CitizenMe, and DataExchange [1]. Data requesters can search and purchase the data that they need in the system. However, most data providers are research institutions or companies in reality, and they may not share data due to privacy or profit. If the data requesters collect the data by themselves, it will incur a large cost. Luckily, crowdsensing [2], [3] has been proposed as a new perception paradigm by leveraging workers' mobility and diverse sensing devices to collect data. Moreover, crowdsensing can be naturally combined with data trading, called Crowdsensed Data Trading (CDT), where data requesters can publish some sensing tasks with specific data requirements and workers can use their mobile devices to sense and share data for achieving rewards [4], [5]. As can be seen, with the future communications and networking, CDT will replace the existing data collection method and become a low-cost, high-efficiency data trading paradigm.

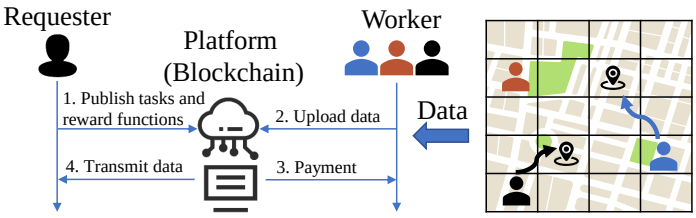


Fig. 1. Crowdsensed data trading system: combining intelligent blockchain with future networking and communications

Generally, a CDT [6] system includes a platform, data requesters, and workers. As shown in Fig. 1, the requester first publishes a task with sensed locations and the reward function to the platform. Then workers interested in the task move to the task location to complete the task and upload sensed data. Next, the platform pays workers a certain reward. Finally, the platform transmits the data to the requester. There have been several works to design CDT systems. Some of them [7], [8], [9], [10], [11] study the worker selection mechanism in CDT, where the platform selects a worker group based on the worker's historical information. Also, some works [12], [13], [14], [15] design the payment mechanism in CDT, in which the platform will calculate the quality of workers' data to pay rewards.

Most existing CDT systems assume a trusted centralized platform. However, we argue that the platform may be untrusted and that it can collude with workers or requesters to trick others for achieving more benefits. For example, the platform may collude with the requester to change its reward function according to the workers' uploaded data. Similarly, platforms might collude with workers to let them know the reward function, then workers could forge data. Moreover,

[1]En Wang, Jiatong Cai, Yongjian Yang, Wenbin Liu, Hengzhi Wang and Bo Yang are with the Department of Computer Science and Technology and Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, Jilin 130012, China. (E-mail: wangen@jlu.edu.cn; caijj20@mails.jlu.edu.cn; yyj@jlu.edu.cn; liuwenbin@jlu.edu.cn; wanghz17@mails.jlu.edu.cn; ybo@jlu.edu.cn (Corresponding Author))

[2]Jie Wu is with the School of Computer and Information Sciences, Temple University, USA. (E-mail: jiewu@temple.edu)

This work is supported in part by National Key R&D Program of China under Grant Nos. 2021ZD0112501 and 2021ZD0112502, and National Natural Science Foundation of China under Grant Nos. 62102161, 61772230 and 61972450, and China Postdoctoral Science Foundation under Grant Nos.2021T140261 and 2021M701389, and CCF-Baidu Open Fund (No.2021PP15002000).

requesters and workers can also be malicious. For instance, requesters may post tasks but fail to pay and workers can upload wrong data to mislead the system. The malicious workers may also register multiple accounts to complete tasks, causing unfair payment and low accuracy of sensed data. Therefore, how to deal with untrusted centralized platforms and the influence of malicious workers and requesters in the CDT system is our first challenge.

Fortunately, blockchain [16] can achieve decentralization, and smart contracts [17] can be regarded as a trusted platform in CDT systems. Blockchain has the nature of anonymity, tamper resistance, and transparency. Nodes on the blockchain can directly trade without relying on a third-party platform. There exist some special complex programs deployed on the blockchain, called smart contracts, which can automatically execute operations according to trading conditions and enforce the participants to fulfill their obligations. Due to the immutability of smart contracts, malicious requesters cannot refuse the payment and the data trading process is more trustworthy. After the requester publishes tasks, the related smart contracts cannot be modified and the workers cannot see the entire smart contract. To deal with the influence of malicious workers, we design the DUP mechanism to group malicious workers and propose the GTD algorithm to weaken the influence of malicious workers' data on the estimated truth.

Although some works [18], [19], [20], [21], [22], [23] use blockchain technology to achieve the decentralization of the CDT systems, due to a large number of requesters and workers in CDT systems, the low efficiency of the blockchain reduces the performance of the system. For example, the Bitcoin blockchain system can only process about 7 transactions per second [23]. The main reason is that every node needs to verify and store all transactions. However, as far as we know, the existing works mostly ignore the low efficiency caused by the blockchain, which seriously affects the transaction throughput and system performance of the CDT system. So this is the second challenge we need to deal with.

To achieve high efficiency, sharding [24], [25], [26], [27], [28] is the most promising solution to blockchain scalability. Most existing sharding schemes are considered as complete sharding, where the shards are completely isolated, and each node belongs to only one shard. However, according to relevant research statistics, more than 96% of transactions in the sharding system are cross-shard [26]. Moreover, most of the current complete sharding is random sharding, which will increase the ratio of cross-shard transactions. Therefore, inspired by Pyramid [29], we propose a layered sharding method based on nodes' history information, where nodes can belong to multiple shards.

To sum up, we combine the Crowdsensed Data Trading system with intelligent Blockchain (CDT-B), which includes a smart contract called CDToken that can be regarded as a credible third party. We first utilize the CDToken to record the requesters' reward function and workers' data uploading function to avoid making the targeted trick. At the same time, we not only design a Data Uploading and Preprocessing (DUP) mechanism in CDToken to collect and process the workers' sensed data, but also propose a Grouping Truth Discovery (GTD) to evaluate their data quality for determining the payments. Moreover, considering that there exist a large number of requesters and workers in a CDT-B system, we propose a Layered Sharding blockchain based on Membership Degree (LSMD) to solve the inefficiency problem caused by the blockchain. As far as we know, this is the first work that combines sharding blockchain and CDT system.

The main contributions of this paper are listed below:

- We propose a decentralization CDT-B system with a smart contract CDToken, where the blockchain achieves trustworthy data trading and CDToken avoids platform collusion and malicious requesters.
- We design DUP mechanism, GTD algorithm, and payment mechanism in CDToken. DUP mechanism contains a data uploading mechanism and data preprocessing algorithms, which can handle the influence of malicious workers. GTD algorithm can evaluate the truth of the data and the payment mechanism pays workers according to the quality.
- Due to the low efficiency of the blockchain, we propose LSMD, a novel layered sharding blockchain. Through nodes' historical information, nodes are allocated according to their membership degree for different shards.
- We implement a CDT-B prototype and deploy CDToken to an experimental environment based on Ethereum, and conduct extensive simulations to prove the remarkable performance and practicality of CDToken.

The remainder of this paper is organized as follows. Related work is discussed in Section II. The system model and problem formulation are proposed in Section III. The details of CDT-B are shown in Section IV. Simulation results are presented in Section V. Lastly, the conclusion is summarized in Section VI.

II. RELATED WORK

Crowdsensed Data Trading. CDT is a novel form of data trading, which allows requesters to hire workers to collect data and obtain suitable rewards. CDT systems are designed to deal with the difficult problem of data acquisition, but this also brings a series of challenges. CMAB-HS [7] is proposed to tackle the problem of quality unknown seller selection and incentive strategy design. DPDT [8] is proposed to preserve the identity privacy of consumers and the task privacy of workers. With the development of CDT system, more researchers have also noticed this field and proposed a series of works.

To efficiently deal with data trading, many existing works study the worker selection mechanism in CDT. Sun et al. [9] propose a trustworthy and cost-effective cell selection (TCECS) framework that takes cell heterogeneity and malicious participants into consideration simultaneously. Wang et al. [10] proposes a novel multi-task allocation framework named MTasker. Cheng et al. [11] design an effective grid-based forecasting method to estimate the spatial distribution of workers/tasks in the future and then use the predictions to assign workers to tasks. Moreover, some works propose the payment mechanism in CDT, in which the platform will calculate the quality of workers' data to pay rewards. Song et al. [12] introduce perceived quality into the design of the

incentive mechanism. Duan et al. [13] design a distributed auction framework and propose two distributed auction schemes, CPAS, and TPAS, to achieve budget balance and high computational efficiency. Meanwhile, to resolve the conflicts between multiple data sources of heterogeneous data types, Li et al. [14] use an optimization framework to model this problem. Cai et al. [15] propose a novel framework for efficient data trading in IoT systems throughout the data collection and data processing phases. Although these works make the CDT system more robust, they all rely on a hypothetical trusted third-party platform. However, the platform may collude with workers or requesters to trick others for achieving more benefits in reality. So our CDT-B system intends to use blockchain to achieve the decentralization of the CDT system, we also consider malicious users in the system and the problems caused by the low efficiency of the blockchain.

Blockchain in CDT. As an emerging distributed ledger technology, blockchain has been widely studied and applied to CDT systems. Blockchain can be regarded as a platform in CDT that achieves decentralization. For example, An et al. [18] design a crowdsensing quality control model based on a two-consensus blockchain. Zheng et al. [19] design a blockchain-based decentralized data trading platform, on which data providers can better control data trading. But they do not consider the impact of malicious users on the system. Dai et al. [20] and Cai et al. [21] allow the requester to purchase a statistical result calculated by some blockchain nodes, which protect data privacy by SGX and additive secret sharing, respectively. But they do not apply to our scenario, where truth discovery needs to be calculated on the raw data and then the system pays workers based on their data quality. A blockchain-based crowdsensed data trading system is proposed in [22]. The workers send the sensed data to consumers for truth discovery and truthful rating and the data is encrypted to ensure data security during the truth discovery process. But in our paper, truth discovery is done by CDToken to avoid untruthful consumers' rating and the high computing cost of encryption. Moreover, the existing works do not take into account the impact of blockchain's low efficiency problem on CDT systems, which is an important contribution of our paper.

Truth discovery. Truth discovery is widely used in data quality assessment. Yin et al. [30] first define the truth discovery problem and propose the truthfinder algorithm that utilizes the interdependency between website trustworthiness and fact confidence to find trustable websites and facts. Li et al. [31] propose a framework that different types of distance functions can be plugged into capturing the characteristics of different data types, and the estimation of source reliability is jointly performed across all the data types together. In this paper, we implement GTD in CDToken, where the data uploaded by malicious workers is regarded as a group.

Efficiency solution in blockchain. The main reason for blockchain inefficiency is that every node needs to verify and store alltransactions. Sharding [28] is the most promising blockchain scaling solution to achieve high performance. The core idea of sharding is to divide and conquer, which divides all nodes into different groups, and each group is a shard. The tasks are then grouped and assigned to different shards for parallel processing, thereby improving the performance of the overall blockchain. ELASTICO [24] is the first open and decentralized sharding-based blockchain system, where each shard is responsible for verifying the PBFT-based consensus of a set of transactions and files. Omniledger [25] is the first sharding-based blockchain system to achieve full sharding. The system adopts a client-driven mechanism to upload cross-shard transactions. RapidChain [26] proposes a transfer mechanism in the unspent transaction output-based system. For each cross-shard transaction, the mechanism first transfers all involved UTXOs to the same shard by sub-transactions. Monoxide [27] proposes a relay mechanism in the account/balance-based system. Each cross-shard transaction is split into several sub-transactions including internal transactions and relay transactions. Pyramid [29] allows nodes to belong to multiple shards, which reduces the proportion of cross-shard transactions. Although there are many works on sharding at present, they do not consider CDT and cannot be directly applied to this work.

TABLE I
MAIN NOTATIONS

Notation	Meaning
u_i, t_j	the i-th worker, the j-th task
n, m	the number of tasks and the number of workers
T, T_i	the set of tasks and the set of tasks that w_i deals with
U, U_j	the set of workers and the set of workers who execute t_j
t_{fir}, t_{sec}	the time limits for data uploading
D_i, ED_i	the data and the encrypted data of u_i
d_i^j	the data of worker u_i for task t_j
$A_{i,j}$	the number of tasks either u_i or u_j has done alone
$B_{i,j}$	the number of tasks both u_i and u_j have done
$S_{i,j}$	the similarity between u_i and u_j
G, g_i	the preprocessed data and the i-th group of workers
$\widehat{d_j^k}, \widehat{w_k}$	the result of g_k for task t_j and the weight of g_k
w_i, d_j^*	the weight of u_i and the truth of t_j
λ_j^i, r_j^i	the data quality and reward of u_i for t_j

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model

We first discuss the system model of CDT-B and the main notations are listed in Table I. In the data trading scenario, the data requester hopes to obtain data from the Points of Interest (PoIs) within a certain period. The requester can collect data himself or deploy some fixed sensors, which is not a low-cost method for the requester to ask for high-quality collected data. Therefore, the requester intends to hire a group of workers to complete the task of collecting data and pay them corresponding rewards. Due to the influence of malicious workers and malicious platform, we combine the crowdsensed data trading system with intelligent blockchain.

As shown in Fig. 2, the node that wants to publish the task becomes a requester and then it publish a task with data requirements to the blockchain to form a smart contract, i.e., CDToken, including the content, requirements, budget of the task, and so on. CDToken processes the collected data and then calculates the estimated truth of the task. Meanwhile, the

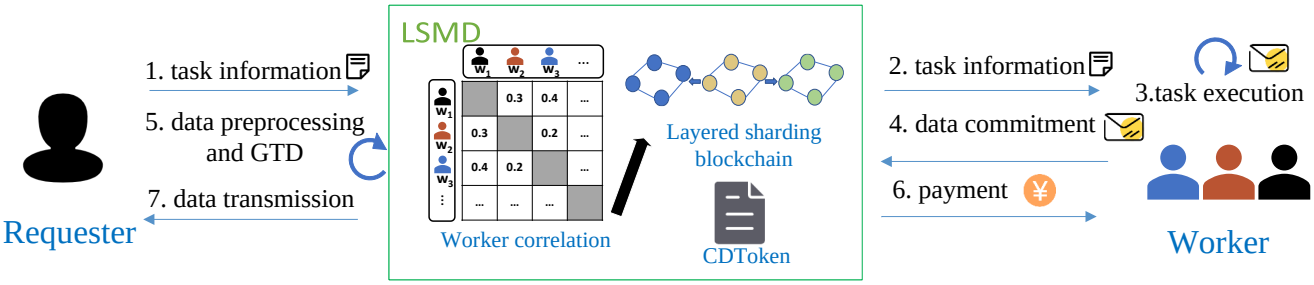


Fig. 2. The model of CDT-B

data quality of each worker is calculated through the truth of the task. Finally, according to the data quality of each worker, the corresponding reward is paid to the worker.

B. Problem Formulation

We consider a CDT-B system consisting of a crowd of n workers $U = \{u_1, u_2, u_3, \ldots, u_n\}$. The requester publishes a sensing tasks set $T = \{t_1, t_2, \ldots, t_m\}$. Each task is to sense a specific object or event in an appointed area. Then a worker u_i performs a set of sensing tasks and uploads the data $D_i = \{(d_i^j, t_i^j) \mid t_j \in T_i\}$, where d_i^j is the sensing data for task t_j in the form of numerical values, e.g., cellular signal strength [32], noise level measurements [33], and Wi-Fi signal strength [34], and t_i^j is the corresponding timestamp. Then worker u_i uploads D_i to the blockchain. Once the task is completed and the sensing data $D = \{D_i\}$ from all workers has been collected, the CDToken on blockchain calculates an aggregated result d_j^* for each task j as an estimate for the truth, which is unknown to either the requester or the workers.

The digital signature can be used to verify identity information and avoid denial, and hash digest is used to protect information from tampering. The requester broadcasts the task information at the start of the task, and the main content is $Task = \{Con, Time, Req, Rule, Baq, B, t_f, t_s, Deposit\}$, where Con stands for the specific content of the task, $Time$ for the time limit of the task, Req for the credit requirements of workers, $Rule$ for the evaluation of data quality, Baq for the basic requirements of the task data, such as data type, data length, etc., Bud for the budget of the task. t_f and t_s are the time limit for data uploading. When the requester publishes the task, it sends the budget of the task in advance. This avoids the requestor's malicious exit during the task. Workers who want to complete the task need to pay $Deposit$ first to avoid the malicious exit during the task.

After the worker u_i completes the task set T_i, they upload the data D_i to CDToken. CDToken preprocesses the data uploaded by workers through DUP, i.e., Data Uploading and Preprocessing, and then obtains the estimated truth through GTD, i.e., Grouping Truth Discovery. Next, CDToken gets the data quality of each worker based on the truth and pays the workers rewards based on their data quality. Finally, the data will be sent to the requester. Due to the low efficiency of the blockchain, we also design a layered sharding blockchain based on membership degree. The nodes of different shards can process transactions in parallel.

IV. DETAILS IN CDT-B SYSTEM

In our CDT-B system, there is no guarantee that workers and requesters are trustworthy. Moreover, workers may not follow the rules of the tasks. For example, workers maliciously withdraw from the task early, or workers can be paid to peek at data uploaded by other workers and upload corresponding data. This results in an inaccurately estimated truth. It also has a side effect on worker incentives. Therefore, we propose DUP including a two-step data uploading mechanism to make the whole data uploading process trustworthy and the data preprocessing algorithm based on Task Set Similarity Calculating (TS-SC) and Accounts Trajectory Similarity Calculation (AT-SC). Moreover, the GTD algorithm and the reward payment algorithm are proposed. Finally, we propose LSMD to solve the low efficiency of blockchain where nodes are assigned to different shards according to membership degree.

A. Data Uploading and Preprocessing

In the data uploading stage, each worker uploads his data to CDToken. Due to the transparency of blockchain, the data uploaded by each worker is publicly visible. To prevent unfairness caused by data leakage, we split the data uploading into two steps.

Step 1: upload encrypted data. The first step is to upload the encrypted data. Worker u_i first calculates an encrypted data ED_i by using the SHA256 Algorithm in Secure Hash Algorithm 2 [35]. SHA256 takes his data D_i and a random number $nonce_i$ as input, $i.e., ED_i = SHA256(D_i, nonce_i)$. Then the worker u_i sends the encrypted data ED_i to CDToken. CDToken uses a function $f - upload()$ to verify that the current time is less than the preset upload time of the task, $i.e., now < t_f$. It also checks whether deposits provided by workers meet requirements, $i.e., msg.value > Deposit$. Finally, $f-upload()$ stores the encrypted data and the deposits against the worker's accounts.

Step 2: upload real data. The second step is to upload unencrypted data, and then verify through the encrypted data. First, the worker u_i sends D_i and random numbers $nonce_i$ to CDToken. Then $s-upload()$ first checks whether the current time meets the requirements of the task, $i.e., t_f < now < t_s$. Then $s-upload()$ checks the consistency of the current data and the encrypted data, $i.e., SHA256(D_i, nonce_i) = ED_i$. If worker u_i passes the time check and consistency check, $s-upload()$ will record his data. Otherwise, worker u_i is dishonest and his $Deposit$ will be deducted.

TABLE II
EXAMPLE SHOWING MALICIOUS WORKERS

Worker	Task1	Task2	Task3	Task4
1	9.4	5	7.4	3.51
2	9.8	5	7.8	3.2
3	9.5	7	7.8	3.36
4'	20	20	20	20
4"	20	20	20	20
4"'	20	20	20	20
TD without malicious workers	9.65	5.74	7.78	3.36
TD with malicious workers	11.49	10.09	10.64	9.72

TABLE III
EXAMPLE SHOWSING IN CROWDSENSING

Worker	Task1	Task2	Task3	Task4
1	10:16:15	10:24:17	10:31:19	10:11:16
2	x	10:26:24	10:34:24	x
3	10:18:34	10:23:41	x	10:13:44
4'	10:23:36	x	10:21:35	10:24:34
4"	10:24:37	x	10:23:18	10:24:19
4"'	10:26:22	x	10:25:23	10:25:28

In practice, the quality of sensing data from different workers is varying and unknown. Therefore, the truth discovery algorithm [36], [37] is used to aggregate data, calculate workers' weights and jointly estimate the truth. Then, the data quality of workers is calculated through the truth. However, due to the anonymity of the blockchain, workers have no identity information. Thus, some malicious workers may register multiple nodes to complete the task. Some workers will also suffer and lose motivation to complete tasks. This may even mislead the system.

First, we provide an example shown in Table II to demonstrate that the existing truth discovery algorithms are vulnerable to malicious workers. Consider a CDT-B system with 4 tasks and 4 workers, one of whom is a malicious worker. Each task measures the temperature at a given location, and each account can upload at most one data for a task. Finally, the system aggregates the worker's data into the temperature at that task location. Suppose Workers 4 is a malicious worker who uses 3 accounts to upload fake data (20), in order to mislead the aggregation results of the system. We used CRH [31], a widely used truth discovery algorithm, to aggregate the uploaded data both with and without the malicious worker. According to the result, we can see that the data uploaded by a malicious worker has a significant impact on the aggregated results of the tasks. Therefore, it shows that the existing truth discovery algorithms are susceptible to influence when workers are unreliable.

To deal with the influence of malicious workers, the core idea is to identify malicious workers. Since a malicious worker may register multiple accounts to mislead the system, we use accounts instead of workers for the sake of illustration. We find two characteristics of malicious workers:

- First of all, a malicious worker might want to manipulate the aggregated results of multiple task data, so they should upload data for each task using different accounts, such as the example in Table II. Therefore, the data uploaded by accounts with high task sets similarity is likely to come from malicious workers.
- Secondly, the trajectory of a malicious worker's accounts should have a similar pattern. So the data uploaded by accounts with high trajectory similarity is likely to come from malicious workers.

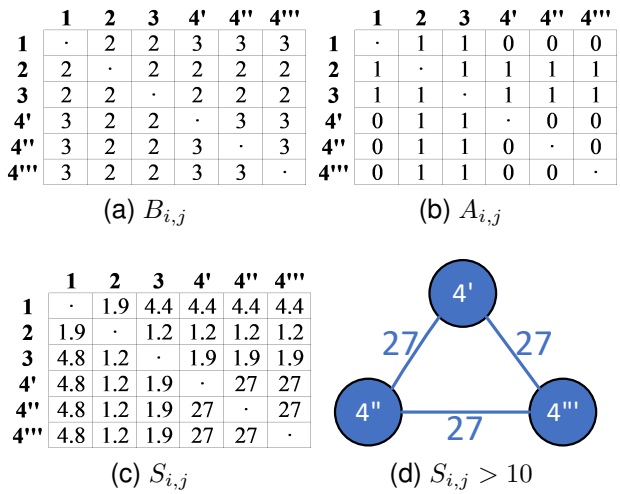


Fig. 3. An example of TS-SC

For the above two characteristics, we group accounts that are likely registered by malicious workers and give their data a lower weight in GTD. Please note that we do not delete the data of these potential malicious accounts because of the possibility of misidentification. Next, we introduce two grouping algorithms.

Task Set Similarity Calculating (TS-SC). Due to the fact that the task sets completed by malicious accounts should have a high degree of similarity, which means they perform similar tasks. Inspired by [38], we design a grouping algorithm based on two accounts' similarity of task sets they have done. The grouping method includes the following steps:

- $B_{i,j}$ represents the number of tasks both account i and account j have done. $A_{i,j}$ represents the number of tasks either account i or account j has done alone. Then the similarity between account i and account j, denoted as $S_{i,j}$, is calculated as:

$$S_{i,j} = \frac{m}{B_{i,j} + A_{i,j}} e^{\frac{B_{i,j}}{A_{i,j}}} \tag{1}$$

  where m is the total number of tasks. The greater the similarity, the more similar the task set of the two accounts are.
- Then an undirected graph is constructed, where nodes represent accounts, and the edge between i and j represents their similarity $S_{i,j}$. Note that only edges that exceed the threshold p are included in TS-SC.
- Subgraphs are discovered using Depth First Search (DFS) algorithm. Each component represents a set of accounts that have done similar tasks. Each subgraph is a group, and the account that is not in any component will be treated as a separate group.

To illustrate the process of this grouping method, we provide an example in Table III, where the values in the table are the timestamps for the corresponding tasks.

Fig. 3 shows the procedure of TS-SC. The adjacency matrix in Fig. 3a shows the number of tasks both i and j have done. The adjacency matrix in Fig. 3b shows the number of tasks either i or j has done alone. Fig. 3c shows the similarity value between i and j. We set the threshold $p = 10$, and an undirected graph is constructed as shown in Fig. 3d. We see that one component is constructed in this example, i.e., $\{4', 4'', 4'''\}$. Account 1, 2, and 3 are not in this component, and each of them becomes a group. The grouping result of this example consists of four groups, i.e., $\{4', 4'', 4'''\}$, $\{1\}$, $\{2\}$, $\{3\}$. Note that TS-SC groups the malicious accounts in the same group, and thus it is effective.

TS-SC can be used in the scenario where accounts have diverse accomplished task sets. To deal with the scenario where most accounts have similar accomplished task sets, we propose the following account grouping method.

Accounts Trajectory Similarity Calculation (AT-SC). As mentioned above, the trajectory of a malicious worker's accounts should be a degree of consistency. Since we get the data $D_i = \{(d_i^j, t_i^j) \mid t_j \in T_i\}$ uploaded by u_i for task t_j as two time series data, namely accomplished task series X_i and timestamp series Y_i, these two time series data can be viewed as the trajectory of an account. Due to malicious workers may register multiple accounts to complete tasks, the trajectories of these accounts should have relatively high similarity.

Since the number of tasks completed by accounts may vary, Dynamic Time Warping (DTW) [39] is a measure of the similarity of two time series of different lengths. Suppose we have two time series data like two workers' accomplished task series, Q and C, whose lengths are n and m, respectively. Suppose we have two time series data like two workers' accomplished task series, Q and C, whose lengths are n and m, respectively. When m is not equal to n, we need to align the two time series data. So we need to construct a matrix grid of $n \times m$, the matrix element (i, j) represents the distance between two points q_i and c_j, $d(q_i, c_j)$, which is generally Euclidian distance, $d(q_i, c_j) = (q_i - c_j)^2$. Each matrix element (i, j) represents the alignment of points q_i and c_j. The idea of the DTW algorithm is to find a path from the bottom left corner of the matrix to the top right corner so that the sum of the elements on the path is the smallest, and we call the sum of the elements on the path length. We define this path as the warping path. The warping path $W = \omega_1, \omega_2, \cdots, \omega_k$ is a contiguous set of matrix elements that defines the DTW distance between Q and C. We define the DTW distance as in [40]:

$$DTW(Q, C) = min\{\frac{\sqrt{\sum_{k=1}^{K} w_k}}{K}\} \tag{2}$$

where K in the denominator is mainly used to compensate for paths of different lengths. Here we define a cumulative distance matching the two series Q and C starting at $(0, 0)$, and at each point, the distances calculated at all previous points add up. After arriving at the end point (n, m), the cumulative distance is the total distance we said above, that is, the similarity between the sequence Q and C. Cumulative distance $\gamma(i, j)$ can be expressed in the following way, the cumulative distance $\gamma(i, j)$ is the current grid point

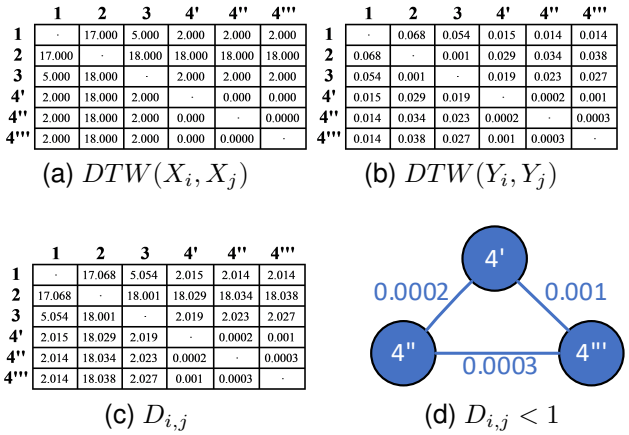


Fig. 4. An example of AT-SC

distance $d(i, j)$, that is, the Euclidian distance of points q_i and c_j and the cumulative distance of the smallest adjacent elements that can reach the point: $\gamma(i, j) = d(q_i, c_j) + min\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\}$.

Based on the DTW distance between the task series and the time series of the accounts, we propose a grouping method that includes the following steps:

- The difference between account i and j, denoted as $D_{i,j}$, is calculated as:

$$D_{i,j} = DTW(X_i, X_j) + DTW(Y_i, Y_j) \tag{3}$$

  Note that the less the difference, the more similar the trajectories of the two accounts.
- Then an undirected graph is constructed, where nodes represent accounts, and the undirected edge between i and j represents their difference value $D_{i,j}$. Note that only edges that are under a threshold φ are included.
- Like TS-SC, subgraphs are discovered using the Depth First Search algorithm. Each component represents a set of accounts with similar trajectories. Each subgraph is a group, and the account that is not in any component will be treated as a separate group.

Next, we still use the example in Table III to illustrate the process of AT-SC. We use two adjacency matrices in Fig. 4a and Fig. 4b to represent the DTW difference between account i and account j according to the task series and timestamp series. The adjacency matrix in Fig. 4c shows the difference value between account i and account j. We set the threshold $\varphi = 1$, and thus an undirected graph is shown in Fig. 4d. One component is constructed in this example, i.e., $\{4', 4'', 4'''\}$. Accounts 1, 2, and 3 are not in the component, and thus each of them is regarded as a group. Therefore, the grouping result includes four groups, i.e., $\{4', 4'', 4'''\}$, $\{1\}$, $\{2\}$, and $\{3\}$. We can see that the method correctly groups all the accounts used by the malicious workers in one group.

Due to different grouping methods, some accounts may be mistakenly classified as malicious worker groups. So we keep two grouping results and process them in the subsequent grouping truth discovery algorithm.

B. Grouping Truth Discovery

At this stage, we first integrate the grouping data. Let $G = \{g_1, g_2, ..., g_l\}$ denotes the account grouping results, where $g_i \cap g_j = \emptyset$ and $\cup_{g_i \in G} g_i = U$. Each group $g_k \in G$ represents a set of accounts likely used by the malicious workers. Let $\widehat{T_k} = \cup_{i \in g_k} T_i$ denotes the set of tasks performed by the accounts in group g_k. Then CDToken groups data as follows. For each task t_j, we first aggregate the data within each group $g_k \in G_j$, where $G_j = \{g_k \mid g_k \in G, t_j \in \widehat{T_k}\}$. The weight calculation method for each account in group g_k is as follows:

$$w_j^i = 1 - \frac{\left| d_j^i - \overline{d_j^k} \right|}{\sum_{l \in g_k} \left| d_j^l - \overline{d_j^k} \right| + \varepsilon} \tag{4}$$

where $\overline{d_j^k}$ is the mean of data uploaded by accounts in g_k. ε is a small constant real number. The reason we need ε is to make sure the equation still makes sense when $\sum_{l \in g_k} \left| d_j^l - \overline{d_j^k} \right| = 0$. The data results for each group are calculated as follows:

$$\widehat{d_j^k} = \frac{\sum_{i \in g_k} w_j^i d_j^i}{\sum_{i \in g_k} w_j^i} \tag{5}$$

The weight of each group g_k is calculated as

$$\widehat{w_k} = 1 - \frac{|g_k|}{|U_j|} \tag{6}$$

where $|g_k|$ denotes the number of accounts in group $|g_k|$, and $|U_j|$ is the number of accounts who upload data for task t_j. Note that using only one data to represent each group reduces the impact of malicious workers.

Finally, the CDToken estimates the truth for each task according to the truth discovery algorithm. A general truth discovery algorithm can be divided into two stages: weight estimation and truth estimation. First, the algorithm randomly guesses the truth of each task, and then iteratively updates the weight and estimated truth of each worker until convergence.

Weight estimation: In this step, each worker's weight w_i is estimated based on the distance between its data and the estimated truth. Let U_j denotes the workers who upload sensing data for task t_j and e_j^* denotes the last round's estimated truth for task t_j. The weight w_i for task t_j is calculated as

$$w_i = 1 - \log\left(\frac{\mathbb{D}(d_j^i, e_j^*)}{\sum_{i \in U_j} \mathbb{D}(d_j^i, e_j^*) + \varepsilon}\right) \tag{7}$$

where $\mathbb{D}()$ is the distance function measuring the difference between the worker's data and the estimation truth.

Truth estimation: In this step, according to the last round's weight of each worker w_i for task t_j, the estimation truth of task t_j is calculated as

$$e_j^* = \frac{\sum_{i \in U_j} w_i d_j^i}{\sum_{i \in U_j} w_i} \tag{8}$$

Although some existing truth discovery algorithms have some differences in weight and truth update, they all follow the same principles: 1) workers whose data are closer to the estimated truth have a higher weight. 2) the results of the aggregation are more dependent on high-weight workers.

Algorithm 1 Grouping Truth Discovery Algorithm

Input: Workers' data D
Output: Estimated truth $\{d_j^* \mid t_j \in T\}$

```
//Account grouping
G ← AG(D)
//Data grouping
for each t_j ∈ T do
    Group data for t_j based on G
    Aggregate data in each group g_k ∈ G_j using (5)
    Calculate the weight ŵ_k of each group using (6)
end for
Get an initialization estimation truth by (9)
repeat
    //Weight estimation
    for each group g_k ∈ G do
        Update weight w_i by w_i = 1 − log(D(d_j^i, d_j) / (Σ_{i∈U_j} D(d_j^i, d_j) + ε))
    end for
    //Truth estimation
    for each t_j ∈ T do
        Update the estimation truth using d_j^* = (Σ_{i∈U_j} w_i d_j^i) / (Σ_{i∈U_j} w_i)
    end for
until convergence criterion is satisfied
return estimation truth {d_j^* | t_j ∈ T}
```

The truth discovery algorithm we designed is summarized in Algorithm 1. Different from the general truth discovery algorithm, we treat the accounts in one group as a whole, and thus we use d_j^* for group g_k. In addition, instead of randomly initializing the estimated truth for each task, we initialize the estimated truth of each task t_j as follows

$$d_j^* = \frac{\sum_{g_k \in G_j} \widehat{w_k} \widehat{d_j^k}}{\sum_{g_k \in G_j} \widehat{w_k}} \tag{9}$$

Let's assume that the ground truth obtained is $\{d_j^* \mid t_j \in T\}$. We denote the distance between any two data d_j^i and d_j^k by $d(d_j^i, d_j^k)$. The distance measurement function $d()$ measures the similarity between different data. It could be their Euclidean distance, cosine distance, or any other specified similarity distance. A smaller distance usually indicates higher similarity, and vice versa.

The quality of each worker i's data is measured based on its deviation v_j^i from the ground truth, shown in Equation (10). Obviously, data with higher quality is in closer proximity to the ground truth than lower quality ones, which results in a smaller deviation v_j^i.

$$v_j^i = d(d_j^i, d_j^*) \tag{10}$$

Let η_i be the deviation ratio, i.e., $\eta_i = \frac{\sum_{k \in U_j} v_j^k}{v_j^i + \varepsilon}$. We calculate the worker i's data quality based on the following equation:

$$q_j^i = \frac{\eta_i}{\sum_{k \in U_j} \eta_k} \tag{11}$$

where ε is a small constant real number. We note that q_j^i is a real number within $(0, 1)$ and $\sum_{i \in U_j} q_j^i = 1$.

Our proposed quality estimation method handles numerical data values only. The intention of our design can be applied to more general crowdsensing scenarios, where the ground truth can be calculated by aggregating high-quality data, and the distance to the truth reflects the accuracy of each worker's contributed data.

C. Reward Payment

Since we have used two grouping methods before, we can get two data qualities for each worker, $q^i_{j,t}$ and $q^i_{j,a}$. So the quality score of each worker for task j can be expressed as:

$$\lambda^i_j = \gamma * q^i_{j,t} + \beta * q^i_{j,a} \tag{12}$$

where γ and β are the weight of the influencing factors of grouping and $\gamma+\beta = 1$. Considering that workers with higher quality scores should get higher rewards, the reward for each worker is:

$$r^i_j = \lambda^i_j * B_j \tag{13}$$

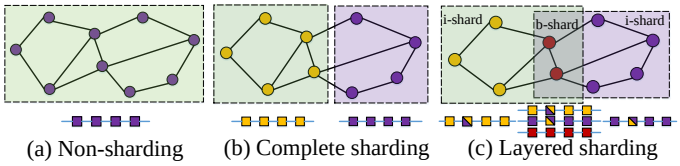
where B_j is the budget of the task j and $\sum_{i\in U_j} \lambda^i_j = 1$.

Under our proposed model, truthfulness can be guaranteed naturally. Workers only upload data that they sensed, and workers cannot get other information when completing tasks and uploading data. The data quality of each worker is related to the data of other workers, so workers cannot manipulate their data to obtain higher rewards. To gain their desirable payment, the best way is to improve the accuracy of their data as much as possible.

D. Layered Sharding Blockchain Based on Membership Degree

Since the consensus mechanism of the blockchain requires the participation of all nodes in the entire network, each node needs to verify and store all transactions, and each consensus message needs to be broadcast in the entire blockchain network like Fig. 5a. This leads to the low efficiency problem of blockchain. However, as far as we know, most of the existing related studies have neglected the low efficiency caused by the blockchain. As shown in Fig. 5b, the current common sharding is called complete sharding, where the shards are completely isolated, and each node belongs to one shard. The nodes in the same shard are responsible for the consensus within the shard, including verification and storage. Pyramid [29] proposes a layered sharding blockchain where nodes can belong to multiple shards. In Fig. 5c, due to red nodes storing the records of both shards, red nodes verify the transaction sent by yellow nodes to purple nodes. The i-shards include nodes that are only responsible for processing internal transactions. The b-shards can handle cross-shard transactions by connecting multiple i-shards.

The sharding structure is randomly generated in Pyramid, and the number of shards is also random. However, in an actual CDT system, this randomization method will reduce the throughput of the system. In crowdsensing, due to the limitations of workers' devices, the tasks completed by each worker are often similar. For example, a worker's temperature sensor is more accurate, so he is likely to often complete tasks



(a) Non-sharding (b) Complete sharding (c) Layered sharding

Fig. 5. Different blockchain systems

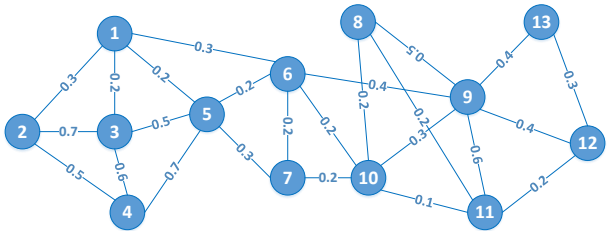


Fig. 6. An example of weighted undirected graph example

related to temperature measurement. Then according to this logic, each worker will frequently transact with some specific people. Then in the transaction verification stage, a shard is formed by nodes that have a high probability of frequent transactions, which will increase the throughput of the system.

Due to the transparency of the blockchain, we can obtain historical data of transactions between nodes. In this way, we get a matrix F of transaction frequency between nodes, where $f_{i,j}$ represents the number of transactions between node i and node j. The matrix F can be regarded as the adjacency matrix of the weighted undirected graph Fig. 6 composed of nodes.

In reality, there are interactions between individuals, and the weighted network can reflect the strength of the interaction between individuals. Next, we propose a Layered Sharding based on Membership Degree (LSMD) shown in Algorithm 2. In the initial stage of sharding, each node can be regarded as an independent shard. We define the weight $s(x)$ of node x as the sum of the weights of edges between all nodes connected to this node:

$$s(x) = \sum_{y\in\tau(x)} \omega_{x.y} \tag{14}$$

The higher the weight of the node, the greater the local influence of the node in the network. And it plays an important role in the sharding process. If the weight of a node is greater than the weights of all its neighbors, this node is a core node.

We first look for core nodes, that is, those nodes with the greatest local influence. We regard each core node as a shard. Next, we expand these shards. Nodes have different affiliations to different shards. We use the membership degree to indicate the degree of membership of the node to the shard and set the membership threshold θ as the basis for sharding. We define the membership degree of node i to shard S_k as:

$$m(S_k, i) = \frac{\sum_{j\in\tau(i)} \omega_{i.j}}{S(i)} \tag{15}$$

Generally speaking, when a node is connected to multiple shards, the membership degrees of the node to multiple shards are calculated and then compared with the threshold θ. When the membership degree of the node is greater than θ, the node is added to the shard. If the membership degree of a node and

TABLE IV
THE MEMBERSHIP DEGREE OF NODES AND SHARDS

Core node	Neighbor node	Membership degree
3	1	0
3	2	0.73
3	4	0.35
3	5	0.16
9	8	0.94
9	10	0.27
9	11	0.92
9	12	0.65
9	13	1

Algorithm 2 Layered Sharding Blockchain Based On Membership Degree

Input: Network = (V, E, W)
Output: Sharing structure S

Set θ //Set initial membership degree threshold
//Get core nodes
Go through every $i \in V$ do:
for each neighbor u of i:
  if $\forall S(u) \leq S(i)$ **then**
    $S = S \cup \{i\}$; //Extend the core node set
//Extend core nodes
repeat
  Go through every $k \in S$ do:
  for each neighbor i of S_k:
    Compute $m(S_k, i)$; //Compute the membership degree
    if $m(S_k, i) > \theta$ **then**
      S_k add i; //Extend the node into the set of core node
until all nodes are visited
return sharing structure S

multiple shards are greater than θ, then this node belongs to these shards at the same time. That means the node is in the b-shard in the layered sharding blockchain.

Next, to facilitate the expansion of shards, we normalize membership degree as:

$$M(S_k, i) = \frac{m(S_k, i) - min}{max - min} \tag{16}$$

where max represents the maximum value of $m(S_k, i)$, and min represents the minimum value of $m(S_k, i)$. The larger the value of $M(S_k, i)$, the greater the possibility that node i belongs to the sharding S_k. Conversely, the smaller the $M(S_k, i)$, the less likely it is that node i belongs to the sharding S_k. If all neighbor nodes of node i are in the sharding S_k, then $M(S_k, i) = 1$, if $M(S_k, i) > \theta$, then node i belongs to the sharding S_k.

We calculate the membership degree of nodes and shards in Table IV. It can be seen from the table, that the algorithm finds two shards formed by core nodes, namely shards $\{3\}$ and $\{9\}$. When we assume that $\theta = 0.4$, in the first round we expand the shards into $\{2, 3\}$ and $\{8, 9, 11, 12, 13\}$, and finally all nodes are divided into $\{1, 2, 3, 4, 5, 6, 7\}$ and $\{6, 7, 8, 9, 10, 11, 12, 13\}$, so $\{6, 7\}$ is the overlapping part, which is the b-shard nodes.

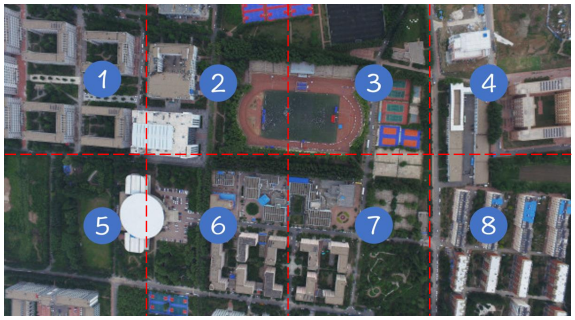


Fig. 7. PoIs for measurement

V. PERFORMANCE EVALUATION

Since there is no public dataset with malicious behaviors and ground truth for crowdsensing, we evaluate our framework through real experiments rather than large-scale simulations. In this section, we first describe our experimental setup. We also introduce some performance metrics for comparison. Then we implement some baselines to compare the performance of the CDT-B system.

A. Experimental Settings

To analyze the performance of CDT-B in this paper, we implement the CDT-B prototype and deploy CDToken to an experimental environment based on Ethereum. In our experiment, we consider a CDT-B system in which the tasks are measuring the Wi-Fi strength, decibel, and light intensity at 8 Points of Interest (PoIs) as shown in Fig. 7. We recruit 50 volunteers in our system, including 40 legal workers and 10 malicious workers. Each legal worker uses one account to perform tasks and each malicious worker has 3 accounts. Malicious workers can use one or more accounts to perform tasks. Note that each account is only allowed to upload one sensor data at one PoI. Therefore, a malicious worker can upload at most 3 data for one task using 3 accounts. We assume that two malicious workers upload fake data. Please note that although there are only 10 malicious workers in our experiment, the experimental results can still represent the scenario when there are malicious workers in the CDT-B system since the percentage of malicious accounts is close to the percentage of legal accounts. We collect Wi-Fi strength, decibel, and light intensity in each PoI multiple times, and calculate the average value as the ground truth. To represent the activity of each account, we define

$$\alpha_i = \frac{|T_i|}{m} \tag{17}$$

where $|T_i|$ is the number of tasks performed by worker i and m is the total number of tasks. In our experiment, each account has to perform at least two tasks and each account performs the task according to his or her preference. It is not hard to see that activity is a good indicator of the contribution of accounts to the system. However, more active malicious workers can cause more damage.

For sharding, the account/balance model is used to represent the state of the ledger, and each node has its account and

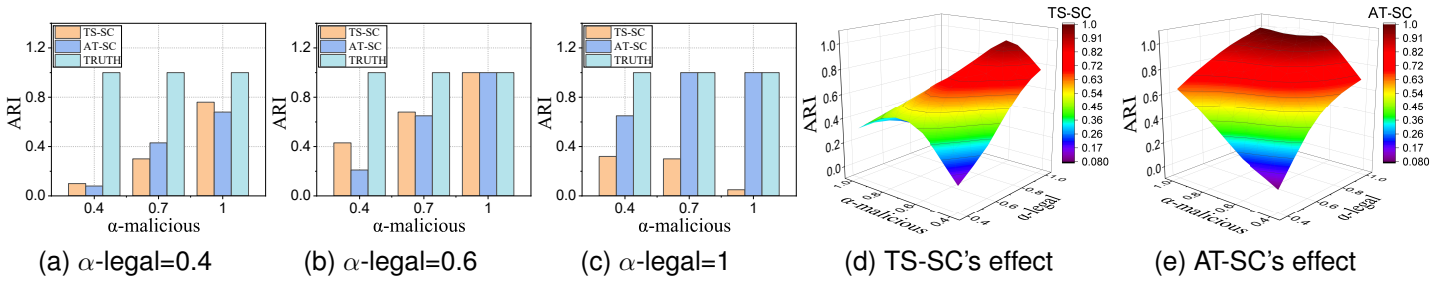


Fig. 8. ARI comparison with different α-legal

balance. The nodes in the sharding blockchain are connected by a partially synchronized peer-to-peer network [41], and messages sent by the nodes can reach any other node through an exponentially increasing timeout. Similar to most running blockchain test platforms, in our test platform, the bandwidth of all connections between nodes is set to 30Mbps, and the link delay is 2 milliseconds. In a consensus round, each node can verify up to 4096 transactions. We implement a prototype of the sharding blockchain based on the degree of membership for performance evaluation. For comparison, we also implement non-sharding and complete sharding prototypes.

B. Performance Metrics & Baselines

The following metrics are used to analyze the performance of the CDT-B system.

- Adjusted rand index (ARI) [42]: This is a widely used criterion for evaluating clustering performance. ARI ranges from -1 to 1 and the larger the value, the better the clustering performance. ARI reflects the degree of overlap between the two groups.
- Average running time: This value is the running time of different truth discovery algorithms.
- Mean absolute error (MAE): This value is a measure of the accuracy of the truth obtained by the truth discovery algorithm. The specific calculation is as follows.

$$MAE = \frac{1}{m}\sum_{j=1}^{m}\left|d_j^* - gt_j\right| \tag{18}$$

where m is the number of tasks, and d_j^* and gt_j are the estimated truth and ground truth for task t_j, respectively. The lower the MAE value, the higher accuracy for the data aggregation.

- Transaction per second: This value is the transaction throughput in transactions per second for LSDM and other sharding blockchain systems.
- Confirmation latency: This value is the confirmation latency of transactions for LSDM and other sharding blockchain systems.
- Extended modularity (EQ): Since we cannot predict the structure of the shards, we use modularity to evaluate the quality of the results. Moreover, our shards overlap, and the traditional modularity cannot be accurately measured. So we use EQ to evaluate the quality of the sharding structure [43]. EQ is calculated as follows:

$$EQ = \frac{1}{2m}\sum_{k=1}^{K}\sum_{v_i,v_j\in C_k}[A_{ij} - \frac{d_i d_j}{2m}]\frac{1}{O_i O_j} \tag{19}$$

where m is the total number of edges of the network, K is the number of shards, d_i is the degree of node v_i, O_i is the number of shards to which node v_i belongs, and A is the adjacency matrix of the network. If there is an edge between v_i and v_j, then $A_{ij} = 1$; otherwise, $A_{ij} = 0$. The larger the value of EQ, the higher the quality of the sharding structure.

To analyze the performance of GTD proposed in this paper, we compare it with two truth discovery algorithms, one is based on CRH [36] and the other based on the truthfinder [30]. Due to the different experimental environments, we adjust the algorithm according to the core idea to adapt to our model.

To compare the performance of sharding based on membership degree, we also implement three other sharding methods:

- Spectral clustering [44] is an algorithm evolved from graph theory. This is a widely used clustering algorithm. Compared with the traditional K-Means algorithm, spectral clustering is more adaptable to data distribution.
- COPRA [45], a label delivery method. The main idea is that the label of a node is affected by surrounding nodes. This algorithm can be regarded as an improved algorithm of the RAK algorithm. The biggest improvement of the COPRA algorithm to the RAK algorithm is that it can discover overlapping communities.
- Random like Pyramid [29], in which nodes are randomly allocated to the shards. Nodes in b-shard could handle cross-shard transactions.

We test the EQ of these sharding methods on four widely used partitioned datasets:

- *Dolphin* data set [46] is obtained by D. Lusseau and others through long-term observation records of the lifestyle of 62 dolphins in a community living off Doubtful Sound. The nodes represent dolphins, and the edges represent the frequency of contact between two dolphins.
- *Football* data set [47] contains the network of American football games between Division IA colleges during regular season Fall 2000, as compiled by M. Girvan and M. Newman. The nodes have values that indicate to which conferences they belong.
- *Karate* data set [48] is obtained by research scholar Zachary by observing the relationships between members

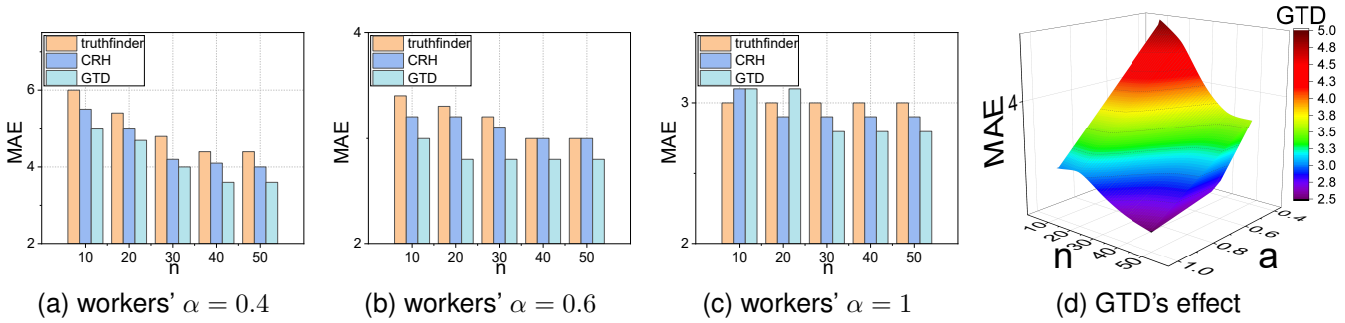


Fig. 9. MAE comparison without malicious workers

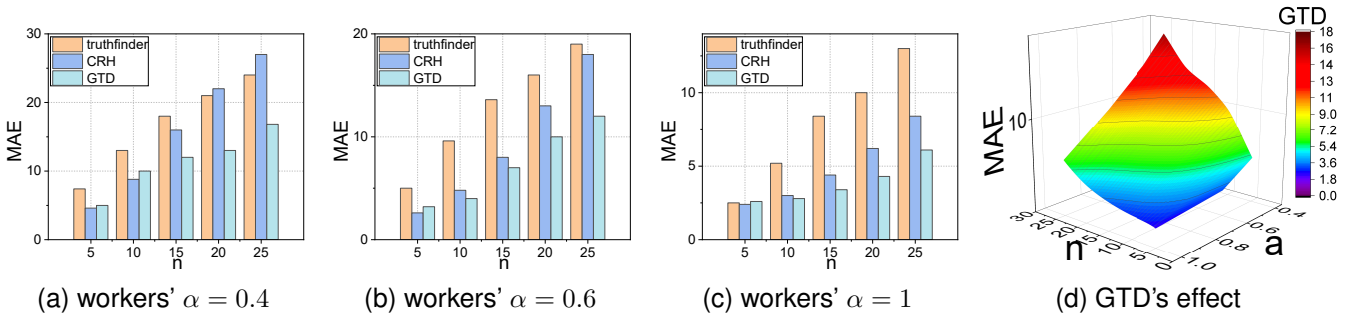


Fig. 10. MAE comparison with malicious workers

of karate clubs in American universities and describes the relationships between members in a small social network.

- *Polbooks* data set [49] is a book about American politics. The node represents books about American politics sold on Amazon's online bookstore. Edges represent frequent co-purchasing of books by the same buyers, as indicated by the "customers who bought this book also bought these other books" feature on Amazon.

C. Simulation Results and Analysis

1) DUP: In this section, we first analyze the performance of TS-SC and AT-SC and then compare the performance of grouping methods according to ARI. Fig. 8 shows the ARI of grouping methods and the truth under different settings. In each setting, the activities of legal workers are fixed, and that of malicious workers varies. We set the activity in three grades, $\alpha = 0.4$, 0.6, and 1 respectively. In Fig. 8, we can see that ARI will increase with the increase of the activity of malicious workers in the two grouping methods. This is because more information like accomplished task sets and timestamps is used to distinguish between these malicious workers. As shown in Fig. 8d, since TS-SC distinguishes workers by comparing the similarity of task sets completed by workers, when workers' activity $\alpha = 1$, the similarity of task sets among workers is very high, so TS-SC performance deteriorates. However, in Fig. 8e, AT-SC not only compares the similarity of task sets but also compares the timestamp similarity of workers' completion of tasks. So ARI in AT-SC increases as worker activity increases. Overall, TS-SC performs better when workers' activities are low. AT-SC performs better when workers are more active.

2) GTD: We now use MAE as a metric to measure the accuracy of the proposed method and the two methods mentioned above. We analyze the MAE of the GTD algorithm in the CDT-B system with or without malicious workers. As can be seen in Fig. 9, no malicious workers are added under different settings. In each setting, we fix the activeness of workers and vary the number of workers. We see that the MAE values of the three methods decrease with the increase of the worker numbers from Fig. 9d. The reason is that as more data is obtained from legal workers, these methods will be more accurate in estimating aggregate results. Also, the MAE decreases when the workers' activity increases. This is because that, with the increase in workers' activity, more task data can be obtained, making the accuracy of the algorithms higher. As shown in Fig. 9, the three methods have similar performance under different settings.

Fig. 10 shows the MAE of the proposed method and the two methods mentioned above with the addition of malicious workers in different settings. In each setting, we still fix the activeness of workers and vary the number of malicious workers. We can see that the MAE values of the three methods increase with the increase in the number of malicious workers. This is because, with more data from malicious workers, it is harder to guarantee the accuracy of the aggregation results. We also see that the MAE values of the three methods decrease with the increase in the activity of workers. The reason is that as more data is obtained from workers, it is harder for malicious workers to manipulate the aggregated results. As shown in Fig. 10c, the MAE of truthfinder is still larger even with the high activeness of workers. But the MAE of our proposed method is always lower than CRH and truthfinder no matter the number of malicious workers. This is because that our method can diminish the impact of malicious behaviors by grouping data from the suspicious workers. We can see from Fig. 10d that MAE grows larger as the number of malicious

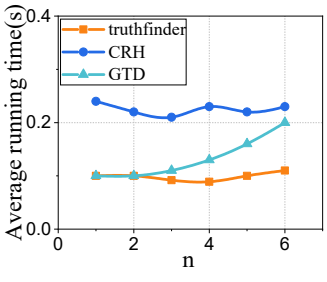


Fig. 11. Time comparison

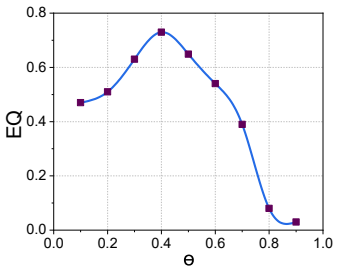


Fig. 12. EQ with different θ

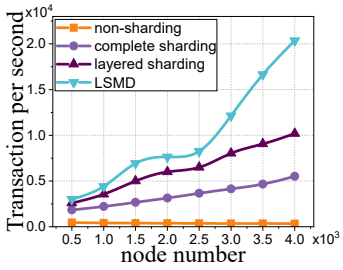


Fig. 13. Transaction per second with varying node number

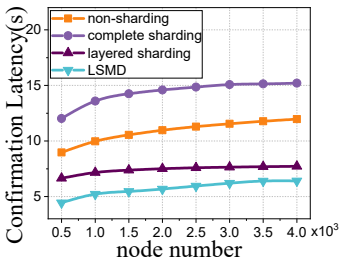


Fig. 14. Confirmation lantency with varying node number

workers increases. At the same time, the MAE decreases with the increase in worker activity, because the system obtains more data to make the estimated truth more accurate.

As shown in Fig. 11, among the three methods, the running time of CRH is the longest. The running time of truthfinder is the shortest and keeps at the millisecond level. The running time of GTD is slightly longer than truthfinder and GTD's running time increases more as the number of workers increases.

3) LSMD: For LSMD, we use artificial synthetic network data sets and first compared the effects of different θ on the results. From Fig. 12, we can see the change of the EQ value of the entire slice structure as theta changes. When $\theta = 0.4$, the EQ value is the highest. When θ becomes larger and larger, the number of nodes that have not joined any shard will increase, forming an isolated node. This makes the structure of sharding worse, and the EQ value drops significantly. So in the next experiment we set $\theta = 0.4$.

We measure the transaction throughput of LSDM and other sharding blockchain systems with different numbers of nodes, and the results are shown in Fig. 13. We can see that as the number of nodes in the network grows, the performance of non-sharding has been poor. The transaction per second of complete sharding and layered sharding are getting larger and larger, and the transaction per second of LSMD is the highest. This is because the other three methods do not consider the impact of historical information on the fragmentation structure, which leads to excessive randomness of the fragmentation process. Our sharding method significantly improves the transaction per second of the system, and to a certain extent overcomes the overhead caused by using the blockchain.

Then we study the comparison latency in LSDM and other blockchain systems with different numbers of nodes. As shown in Fig. 14, the latency increases as the node number increases. As the node number increases, the number of shards also increases, which causes more cross-shard transactions. In

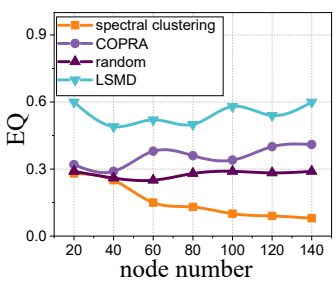


Fig. 15. EQ comparison with varying node number

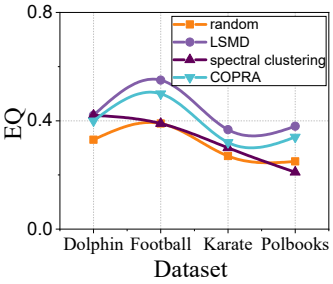


Fig. 16. EQ comparison on different data sets

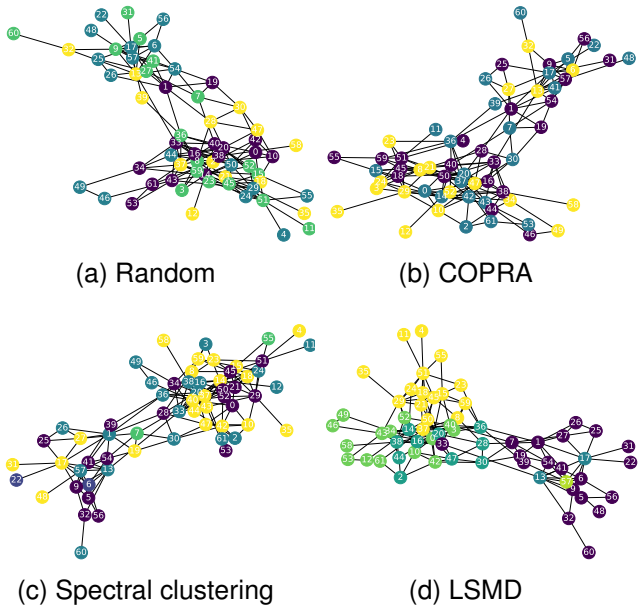


Fig. 17. Different sharding structures on Dolphin data set

complete sharding, cross-shard transactions are divided into multiple sub-transactions, so confirmation latency increases. The confirmation latency of LSMD and layered sharding is relatively close because their sharding structure is similar.

We evaluate the EQ of LSDM and the other three sharding algorithms under different node number settings on the artificial data sets, and the results are shown in Fig. 15. We find that the performance of spectral clustering is poor because the algorithm does not work well when facing overlapping shards. COPRA performs slightly better than random, but due to the randomness of the COPRA algorithm, the sharding results are unstable. LSMD has the highest EQ.

Then we measure the EQ of these several sharding algorithms on real-world datasets shown in Fig. 16. We find that the performance of spectral clustering and random is relatively poor, and the performance of LSMD and COPRA is relatively close. The LSMD algorithm has the highest EQ value, that is, the sharding structure has a better performance.

In addition, to express sharding structure more intuitively, we take the *Dolphin* data set as an example. As shown in Fig. 17, since random and COPRA have certain randomness, the sharding structure of the algorithm is not good enough. Meanwhile, as the shard number needs to be set in advance in the spectral clustering algorithm, the algorithm performance

depends on the shard number. In our experiment, the shard number is consistent with the first two algorithms. As can be seen from Fig. 17d, the sharding structure effect of the LSMD algorithm achieves a better performance.

VI. CONCLUTION

In this paper, we combine the Crowdsensed Data Trading system with intelligent Blockchain (CDT-B), which includes a smart contract called CDToken that can be regarded as a third party. We first utilize the CDToken to record the requesters' reward function and workers' data uploading function to avoid making the targeted trick. At the same time, we not only design a Data Uploading and Preprocessing (DUP) mechanism in CDToken to collect and process the workers' sensed data, but also propose a Grouping Truth Discovery (GTD) to evaluate their data quality for determining the payments. Moreover, considering that there exist a large number of requesters and workers in a CDT-B system, we propose a Layered Sharding blockchain based on Membership Degree (LSMD) to solve the inefficiency problem caused by the blockchain. Finally, we deploy CDToken to an experimental environment based on Ethereum and demonstrate its remarkable trustworthiness and efficiency. In future work, we will consider the dynamic sharding blockchain to adapt to the real-time changes of the CDT-B systems.

REFERENCES

[1] T. Jung, X.-Y. Li, W. Huang, J. Qian, L. Chen, J. Han, J. Hou, and C. Su, "Accounttrade: Accountable protocols for big data trading against dishonest consumers," in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[2] Y. Hu, J. Wang, B. Wu, and S. Helal, "Rl-recruiter+: Mobility-predictability-aware participant selection learning for from-scratch mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2021.

[3] J. Wang, L. Wang, and Y. Song, "Crowd-machine hybrid urban sensing and computing," *Computer*, vol. 54, no. 4, pp. 26–34, 2021.

[4] W. Liu, Y. Yang, E. Wang, H. Wang, Z. Wang, and J. Wu, "Dynamic online user recruitment with (non-) submodular utility in mobile crowdsensing," *IEEE/ACM Transactions on Networking*, vol. 29, no. 5, pp. 2156–2169, 2021.

[5] C. Xiang, Y. Zhou, H. Dai, Y. Qu, S. He, C. Chen, and P. Yang, "Reusing delivery drones for urban crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–17, 2022 (Early Access).

[6] "Thingspeak," https://thingspeak.com.

[7] B. An, M. Xiao, A. Liu, X. Xie, and X. Zhou, "Crowdsensing data trading based on combinatorial multi-armed bandit and stackelberg game," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 253–264.

[8] G. Gao, M. Xiao, J. Wu, S. Zhang, L. Huang, and G. Xiao, "Dpdt: A differentially private crowd-sensed data trading mechanism," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 751–762, 2019.

[9] P. Sun, Z. Wang, L. Wu, H. Shao, H. Qi, and Z. Wang, "Trustworthy and cost-effective cell selection for sparse mobile crowdsensing systems," *IEEE Transactions on Vehicular Technology*, 2021.

[10] J. Wang, Y. Wang, D. Zhang, F. Wang, H. Xiong, C. Chen, Q. Lv, and Z. Qiu, "Multi-task allocation in mobile crowd sensing with individual task quality assurance," *IEEE Transactions on Mobile Computing*, vol. 17, no. 9, pp. 2101–2113, 2018.

[11] P. Cheng, X. Lian, L. Chen, and C. Shahabi, "Prediction-based task assignment in spatial crowdsourcing," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 997–1008.

[12] B. Song, H. Shah-Mansouri, and V. W. Wong, "Quality of sensing aware budget feasible mechanism for mobile crowdsensing," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3619–3631, 2017.

[13] Z. Duan, W. Li, and Z. Cai, "Distributed auctions for task assignment and scheduling in mobile crowdsensing systems," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 635–644.

[14] Y. Li, Q. Li, J. Gao, L. Su, B. Zhao, W. Fan, and J. Han, "Conflicts to harmony: A framework for resolving conflicts in heterogeneous data by truth discovery," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 1986–1999, 2016.

[15] Z. Cai, X. Zheng, and J. Wang, "Efficient data trading for stable and privacy preserving histograms in internet of things," in *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, 2021, pp. 1–10.

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

[17] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, 2014.

[18] J. An, D. Liang, X. Gui, H. Yang, R. Gui, and X. He, "Crowdsensing quality control and grading evaluation based on a two-consensus blockchain," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4711–4718, 2018.

[19] S. Zheng, L. Pan, D. Hu, M. Li, and Y. Fan, "A blockchain-based trading platform for big data," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2020, pp. 991–996.

[20] W. Dai, C. Dai, K.-K. R. Choo, C. Cui, D. Zou, and H. Jin, "Sdte: A secure blockchain-based data trading ecosystem," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 725–737, 2019.

[21] C. Cai, Y. Zheng, Y. Du, Z. Qin, and C. Wang, "Towards private, robust, and verifiable crowdsensing systems via public blockchains," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[22] B. An, M. Xiao, A. Liu, Y. Xu, X. Zhang, and Q. Li, "Secure crowdsensed data trading based on blockchain," *IEEE Transactions on Mobile Computing*, 2021.

[23] C. Li, P. Li, D. Zhou, W. Xu, F. Long, and A. Yao, "Scaling nakamoto consensus to thousands of transactions per second," *arXiv preprint arXiv:1805.03870*, 2018.

[24] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 17–30.

[25] E. Kokoris-Kogias, P. Jovanovic, L. Gasser, N. Gailly, E. Syta, and B. Ford, "Omniledger: A secure, scale-out, decentralized ledger via sharding," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 583–598.

[26] M. Zamani, M. Movahedi, and M. Raykova, "Rapidchain: Scaling blockchain via full sharding," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 931–948.

[27] J. Wang and H. Wang, "Monoxide: Scale out blockchains with asynchronous consensus zones," in *16th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 19)*, 2019, pp. 95–112.

[28] H. Dang, T. T. A. Dinh, D. Loghin, E.-C. Chang, Q. Lin, and B. C. Ooi, "Towards scaling blockchain systems via sharding," in *Proceedings of the 2019 international conference on management of data*, 2019, pp. 123–140.

[29] Z. Hong, S. Guo, P. Li, and W. Chen, "Pyramid: A layered sharding blockchain system," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[30] X. Yin, J. Han, and S. Y. Philip, "Truth discovery with multiple conflicting information providers on the web," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 6, pp. 796–808, 2008.

[31] Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han, "Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 1187–1198.

[32] "Opensignal," http://opensignal.com.

[33] R. K. Rana, C. T. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-phone: an end-to-end participatory urban noise mapping system," in *Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks*, 2010, pp. 105–116.

[34] "Wifi map," https://www.wifimap.io.

[35] H. Gilbert and H. Handschuh, "Security analysis of sha-256 and sisters," in *International workshop on selected areas in cryptography*. Springer, 2003, pp. 175–193.

[36] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han, "A survey on truth discovery," *ACM Sigkdd Explorations Newsletter*, vol. 17, no. 2, pp. 1–16, 2016.

[37] C. Miao, W. Jiang, L. Su, Y. Li, S. Guo, Z. Qin, H. Xiao, J. Gao, and K. Ren, "Privacy-preserving truth discovery in crowd sensing systems,"

ACM Transactions on Sensor Networks (TOSN), vol. 15, no. 1, pp. 1–32, 2019.
[38] J. Lin, D. Yang, K. Wu, J. Tang, and G. Xue, "A sybil-resistant truth discovery framework for mobile crowdsensing," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 871–880.
[39] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16. Seattle, WA, USA:, 1994, pp. 359–370.
[40] C. A. Ratanamahatana and E. Keogh, "Making time-series classification more accurate using learned constraints," in *Proceedings of the 2004 SIAM international conference on data mining*. SIAM, 2004, pp. 11–22.
[41] J. Han and Y. Liu, "Rumor riding: Anonymizing unstructured peer-to-peer systems," in *Proceedings of the 2006 IEEE International Conference on Network Protocols*. IEEE, 2006, pp. 22–31.
[42] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.
[43] Y. Li, J. He, Y. Wu, and R. Lv, "Overlapping community discovery method based on two expansions of seeds," *Symmetry*, vol. 13, no. 1, p. 18, 2021.
[44] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.
[45] S. Gregory, "Finding overlapping communities in networks by label propagation," *New journal of Physics*, vol. 12, no. 10, p. 103018, 2010.
[46] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behavioral Ecology and Sociobiology*, vol. 54, no. 4, pp. 396–405, 2003.
[47] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
[48] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.
[49] V. Krebs, "unpublished," http://www.orgnet.com.

En Wang received his B.E. degree in software engineering from Jilin University, Changchun, in 2011, his M.E. degree in computer science and technology from Jilin University, Changchun, in 2013, and his Ph.D. in computer science and technology from Jilin University, Changchun, in 2016. He is currently a Professor in the Department of Computer Science and Technology at Jilin University, Changchun. He is also a visiting scholar in the Department of Computer and Information Sciences at Temple University in Philadelphia. His current research focuses on the efficient utilization of network resources, scheduling and drop strategy in terms of buffer-management, energy-efficient communication between human-carried devices, and mobile crowdsensing.

Jiatong Cai received his bachelor's degree in software engineering from Jilin University, Changchun, China, in 2020. Currently, he is studying for the master's degree in computer science and technology from Jilin University, Changchun, China. His current research focuses on mobile crowdsensing and blockchain.

Yongjian Yang received his B.E. degree in automatization from Jilin University of Technology, Changchun, Jilin, China in 1983; his M.E. degree in computer communication from Beijing University of Post and Telecommunications, Beijing, China in 1991; and his Ph.D. in software and theory of computer from Jilin University, Changchun, Jilin, China in 2005. He is currently a professor and a PhD supervisor at Jilin University, Director of Key lab under the Ministry of Information Industry, and Standing Director of the Communication Academy. His research interests include: network intelligence management, wireless mobile communication and services, and wireless mobile communication.

Wenbin Liu received the B.S. degree in physics and the Ph.D. degree in computer science and technology from Jilin University, China, in 2012 and 2020, where he is currently a Postdoctoral Researcher in Dingxin Scholar Program with the College of Computer Science and Technology. He was also a visiting Ph.D. student in the Wireless Networks and Multimedia Services Department, Telecom SudParis/Institut Mines-Telecom, France. His research interests include Mobile CrowdSensing, Mobile Computing, and Ubiquitous Computing.

Hengzhi Wang received his bachelor's degree in software engineering from Jilin University, Changchun, China, in 2017. Currently, he is studying for the doctor's degree in computer science and technology from Jilin University, Changchun, China. His current research focuses on mobile crowdsensing, privacy preserving in mobile computing.

Bo Yang is currently a professor in the College of Computer Science and Technology, Jilin University. He is also the director of the Key Laboratory of Symbolic Computation and Knowledge Engineering, Ministry of Education, China. His current research interests are in the areas of data mining, complex network analysis, self-organized and self-adaptive multi-agent systems, with applications to knowledge engineering and intelligent health informatics.

Jie Wu is the Director of the Center for Networked Computing and Laura H.Carnell professor at Temple University. He also serves as the Director of International Affairs at College of Science and Technology. He served as Chair of 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. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Mobile Computing, IEEE Transactions on Service Computing, Journal of Parallel and Distributed Computing, and Journal of Computer Science and Technology. Dr. Wu was general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, ACM MobiHoc 2014, ICPP 2016, and IEEE CNS 2016, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. 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 CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.