CrowdBLPS: A Blockchain-Based Location-Privacy-Preserving Mobile Crowdsensing System

Shihong Zou, *Member, IEEE*, Jinwen Xi, Honggang Wang, *Member, IEEE*, and Guoai Xu

***Abstract*—With the popularization of intelligent terminals, especially current trends, such as "Industrie 4.0" and the Internet of Things, mobile crowdsensing is becoming one of the promising applications built on smart devices in mobile networks. However, the existing mobile crowdsensing models are mostly based on a centralized platform, which is not fully trusted in reality and results in the existence of fraud and other security problems. Furthermore, the data quality collected through crowdsensing is varied, and the location privacy is difficult to guarantee, especially at the worker selection stage. To solve these two problems, an effective blockchain-based location-privacy-preserving crowdsensing model, CrowdBLPS, is proposed in this article. First, the idea of a blockchain is introduced into this model. The decentralized structure and the consensus approach are applied to realize the nonrepudiation and nontampering of information. Second, to improve the data sensing quality and protect worker privacy, a two-stage approach, including the preregistration stage and the final selection stage, is proposed. Finally, we further implement a prototype on the Ethereum public testing network, and the experimental results show the feasibility, availability, and reliability of CrowdBLPS.**

***Index Terms*—Blockchain, Industrie 4.0, Internet of Things (IoT), location privacy preserving, mobile crowdsensing (MCS), reliability.**

Manuscript received August 29, 2019; revised November 24, 2019; accepted December 2, 2019. Date of publication December 6, 2019; date of current version February 28, 2020. This work was supported in part by the National Key R&D Program of China under Grant 2018YFB0803600 and in part by the Beijing University of Posts and Telecommunications Excellent Ph.D. Students Foundation under Grant CX2019118. Paper no. TII-19-3979. *(Corresponding author: Jinwen Xi.)*

S. Zou and G. Xu are with the National Engineering Laboratory of Mobile Network Security and the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100088, China (e-mail: zoush@bupt.edu.cn; xga@bupt.edu.cn).

J. Xi is with the National Engineering Laboratory of Mobile Network Security and the School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100088, China, on leave from the Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: jwxi@bupt.edu.cn).

H. Wang is with the Department of Electrical and Computer Engineering, University of Massachusetts Dartmouth, Dartmouth, MA 02747 USA (e-mail: hwang1@umassd.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2019.2957791

I. Introduction

With the rapid popularization of wireless sensor devices, the Internet of Things (IoT) technology has further deepened our acknowledgment and understanding of the physical world. As a more important part of the traditional IoT, the use of fixed-location sensing methods similar to a wireless sensor network has become the major approach of collecting data [1]. However, it is difficult for this traditional data collection method to meet the diversity of data sensing requirements in most real-world scenarios. Developing a new data collection approach that differs from the traditional one is an urgent goal. In recent years, a series of mobile terminal devices, such as smartphones and Pads, have become an indispensable part of people's daily lives. These terminal devices have certain sensing, computing, and storage capabilities, which can replace the traditional fixed IoT sensors to collect more sensing data in real-life scenarios. Therefore, based on high-performance IoT sensors embedded in mobile terminals and the collection of massive sensing data, mobile crowdsensing (MCS) [2], [3] is proposed as a novel IoT sensing paradigm.

MCS refers to collecting relevant data of the users or the surrounding environment by using the sensors embedded in mobile terminals and the mobility of mobile users, and transmitting data by the existing communication infrastructure (e.g., cellular 4G, wireless WiFi, etc.) [4]. By consolidating the data collected from mobile users in each subarea, the data requester could easily obtain a "big picture" of the target area from the MCS applications without incurring a high deployment cost.

Due to the properties of MCS, it has gradually become an efficient way of perceiving the environment (e.g., air quality, noise, and temperature), traffic (road condition and parking [5]), and other information in real time [6]. Similarly, MCS on the Internet of Vehicles applies vehicle nodes loaded with sensors and centralized data processing centers for real-time traffic management. The goal of sensing is to collect objects, events, or phenomena at a certain time and location by collecting relevant data of the area.

With the rise of MCS techniques, a large number of crowdsensing systems and applications have emerged. However, developing a practical system out of the basic principle entails substantial challenges. First, the implementation of an automatic crowdsensing service relies on a reliable and tamper-resistant transaction management mechanism. However, a central

crowdsensing system may be vulnerable to malicious attacks, so it is difficult for it to avoid repudiation and tempering of information [7], [8]. Second, the tasks always need to be accomplished collaboratively by heterogeneous workers in an untrusted environment; unknowing and distrust on the identities and abilities of others damages the motivation of users to participate in a crowdsensing process. An efficient and automatic arrangement of participants and transactions is, therefore, necessary [9]. Third, in the crowdsensing process, some sensitive information, such as user identity privacy and working information, may be passed to the service platform for processing. A protection mechanism is required to minimize the risk of privacy leakage [10].

To tackle the above challenges while meeting the above requirements, we introduced the idea of a blockchain into MCS to develop a decentralized crowdsensing system and then proposed a two-stage approach to deal with the issues about data quality and location privacy preserving [11]. The blockchain acts as a tamper-resistant distributed ledger, sharing and storing data among a large number of nodes, and helps keep the security and privacy of MCS [12]. Specifically, the distributed ledger technology is of particular relevance to the distributed topology of MCS. Additionally, through, respectively, storing crowdsensing transactions on a large number of nodes, the blockchain answers the challenge of single point of failure and data tampering. Moreover, besides transaction data, requester's information, task information, and working information can also be recorded in the blockchain. These records help build trust between participants in an untrusted environment. For implementing the interaction between users in the crowdsensing process, a smart contract, which contains a set of agreed rules indicating the cooperation serving steps of nodes, can be utilized [13]. The anonymity of the blockchain also ensures that the user's identity privacy is not compromised.

Our proposed crowdsensing system, CrowdBLPS, applies the blockchain-based decentralized framework to replace the traditional centralized framework, which can avoid repudiation and tempering of information and other issues from the centralized system. Following the idea of the smart contract, all the operational steps in the sensing process can be performed automatically by triggering smart contracts, which can ensure that multiple users can collaborate to complete tasks in mutually untrusted environments. For example, at the worker selection stage in a coordinated MCS setup, smart contracts [14]–[16] deployed on the blockchain are responsible for the assignment of sensing tasks with the workers' cloaked locations rather than their exact locations. This article mainly focuses on how to apply the idea of the blockchain to develop a decentralized crowdsensing system and proposes optimization schemes to meet the requirements of location privacy preserving without affecting data sensing quality, achieving a tradeoff between data quality, and privacy preserving.

A. Main Contributions

This article addresses the data quality and location-privacy-preserving problem by designing a decentralized system based on a blockchain in MCS, which is named *CrowdBLPS*. The main contributions are described as follows.

1) We introduce the idea of blockchain to propose a novel decentralized MCS system $CrowdBLPS$. Different from the traditional crowdsensing systems, the introduction of the blockchain in the MCS system can build trust based on cryptography technologies and the consensus mechanism and avoid repudiation and tempering of information and other security issues.
2) Following the idea of a smart contract, we propose a two-stage approach, including the preregistration stage and the final selection stage, to achieve the purpose of data quality control while meeting the requirements of location privacy preserving in the blockchain-based crowdsensing system.
3) For the optimization approach at each stage, we set the formal optimization goals and further demonstrate that the optimization problems at each stage are NP-hard.
4) We implement the proposed solution to verify its feasibility and stability through a software prototype based on the Ethereum public testing network and the real-world dataset. The experimental results indicate that the proposed system, $CrowdBLPS$, is secure and reliable.

B. Article Organization

The rest of this article is organized as follows. We review and summarize the existing works in Section II and present a comprehensive definition in Section III. We present the proposed approaches and efficient algorithms in Section IV. Then, we provide a detailed analysis of our proposed system performance and simulation results in Section V. Finally, Section VI concludes this article.

II. Related Work

With the beginning of the fourth industrial revolution, namely Industry 4.0, intelligence had been already introduced in industries, thanks to embedded systems, which enables communication and cooperation among devices and stand-alone systems, so that a higher level of intelligence can be provided to industrial processes [17], [18]. Furthermore, communication can be used by objects that are already smart, such as smartphones, to augment their intelligence, thanks to information collected by distributed objects, whether it is objective, coming from measures collected by sensors, or subjective, provided by humans [19]. This concept is based on crowdsensing and crowdsourcing, where a large amount of data gathered by multiple nodes are processed to get more information [20].

To guarantee the security and reliability, traditional MCS systems mainly focus on the worker selection stage toward the industrial IoT, which can be categorized into two major approaches: global worker selection mechanism (GWSM) and subarea worker selection mechanism.

1) *GWSM:* This means selecting workers for the working area included in the MCS task. Assuming that each worker receives the same incentives, the system selects a group of workers from volunteer users and requests all workers to participate in all perceptual cycles of the task area. Existing work [21], [22] studied worker selection problems and challenges in participatory sensing tasks and proposed a

worker search framework based on the regional coverage to maximize the spatial coverage by selecting a defined number of workers.

2) *Subarea worker selection mechanism:* This means selecting workers for the divided subareas in each sensing cycle. Guo *et al.* [23] proposed the task model *TaskMe* for image tasks in crowdsensing with a dynamic budget and location-based social networking to perform worker selection. They combined multiple face quality measurements and multiple payments in the enhanced reverse auction scheme to improve the quality of crowdsensing data. Recently, An *et al.* [24] have proposed a node matching method based on the idea of matching degree calculation, improving the quality of the sensing data acquired from the workers.

Research on location privacy preserving for mobile location services in MCS has been widely published in related literature [25]. Hua *et al.* [26] proposed a location perturbation method based on the strict mathematical disturbance data, similar to the concept of differential privacy [27]. A differential privacy algorithm [28] has been recently applied to privacy preserving of anonymous locations or trajectory data. However, the differential privacy algorithm is not suitable for location-oriented services, because in mobile group perception, in order to ensure task coverage, it may be necessary to select workers for each location, even if these locations are uncertain.

Blockchain was invented by Nakamoto in 2008 to serve as the public transaction ledger of the Bitcoin cryptocurrency [29]. The ledger records a continuously growing list of transaction records, called blocks, which are linked by the cryptographic hash of the previous block. A blockchain is typically managed by a peer-to-peer network collectively following a predefined consensus protocol [30]. The public blockchain is transparent and open to everybody without permission. A smart contract is a tiny executable program that is stored inside a blockchain. Once certain conditions are triggered, the program can run automatically. Smart contracts permit trusted transactions and agreements to be carried out among disparate anonymous parties without the need for a trusted third party.

Traditional data quality control and location privacy preserving in above MCS require a trusted central platform. However, we cannot fully guarantee the security and reliability of a centralized platform in reality. To solve the problem, we proposed a novel blockchain-based system model $CrowdBLPS$ in MCS.

$CrowdBLPS$ runs over a public blockchain, where anyone can join and participate and does not require any participating node to buy assets for trading. It realizes nonrepudiation and ensures that the information in crowdsensing cannot be tampered with, avoiding harm from malicious behaviors, such as plagiarism and fraud. Besides, we propose a two-stage worker selection approach, including the preregistration stage and the selection stage in $CrowdBLPS$, to improve the quality of the sensing data acquired from the workers. $CrowdBLPS$, as interdisciplinary research work, representing the novel system realizes effective data quality control and privacy preserving in MCS based on blockchain.

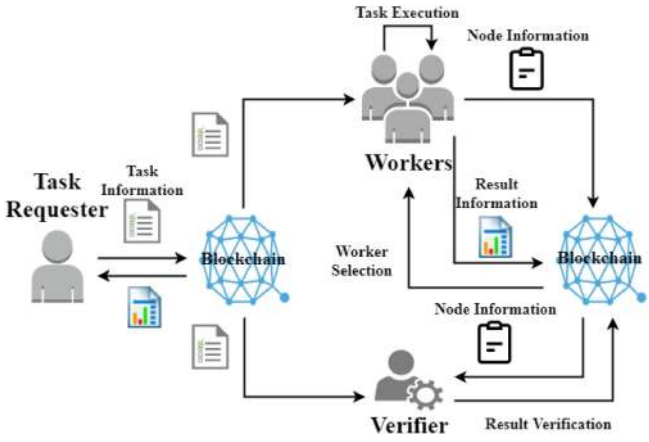

Fig. 1. Blockchain-based crowdsensing system model CrowdBLPS.

III. Blockchain-Based Crowdsensing System

Although the centralized platform plays a very important role in crowdsensing, however, a completely trusted centralized platform does not exist in reality. However, we proposed a blockchain-based crowdsensing system model to achieve decentralization and nonrepudiation in a crowdsensing scenario.

A. Model Overview

The participating modes in the blockchain-based crowdsensing system model, including task requester, workers, and verifiers (miners), manage blockchain by storing and verifying transactions and blocks. Each node can become a task requester or worker according to the actual situation. As shown in Fig. 1, the node that wants to publish a task becomes a task requester and broadcasts the task information through blockchain. The nodes that want to do the task upload their working information and submit a deposit to sign a contract with the task requester. After completing the process of preregistration, the smart contract will be automatically triggered to select suitable workers from the preregistering worker set for the specific task. The deposit will be sent back to the workers who fail to preregister or be selected. After completing the task, the workers upload the result metadata, and the actual data will be encrypted and stored in the distributed database, waiting for the evaluation of the task requester. Finally, the evaluation result will be submitted; if qualified, the workers will get the reward. How to design an appropriate evaluation mechanism in the decentralized crowdsensing framework is an important issue, and we will extend this work in the future.

B. Basic Idea and Initial Setting

As shown in Fig. 1, we present the decentralized system model $CrowdBLPS$, which includes three main roles, a task requester, workers, and a verifier, who can interact with each other in a peer-to-peer network. For better understanding of the worker selection process, notations and their corresponding descriptions are given in Table I. And we give some formal definitions for the corresponding optimization problem.

TABLE I
IMPORTANT NOTATION DEFINITION AND DESCRIPTION

No	Parameter	Description
1	$\mathcal{R}$	A task requester
2	$\mathcal{W}$	A participating worker
3	$\mathcal{V}$	A verifier
4	t_j	Divided sub-target j
5	n	Number of participating workers
6	m	Number of divided sub-targets
7	b_i	Travel distance budget of worker i
8	l_i	Location of the worker i
9	a_i	Cloaked area of the worker i
10	k_j	Coverage worker number of target j
11	g	Task coverage goal
12	$d_{i,j}$	Euclidean distance between i and j
13	$\mathbf{x}$	Result matrix at the first stage
14	$\mathbf{y}$	Result matrix at the second stage
15	$\mathbf{u}$	Area coverage matrix
16	S	Total area of working scope

Definition 1 (Task requester): A task requester $\mathcal{R}$ can anonymously create its corresponding identity and account on the blockchain to post tasks and conduct transactions. Each account contains some properties related to the requester, such as tokens and reputation. $\mathcal{R}$ can publish the task to the blockchain for selecting suitable workers.

Definition 2 (Worker): Similarly, a worker $\mathcal{W}_i$ is an anonymously created user on the blockchain, and the corresponding account will include the reputation and the acceptable travel budget b_i for completing the task. Before receiving the tasks, $\mathcal{W}$ needs to submit relevant working information to the blockchain and prestore a certain deposit for worker selection.

Definition 3 (Verifier): A verifier $\mathcal{V}$ can participate in the verification and consensus process, who is the miner node that is selected through the proof of work. $\mathcal{V}$ is responsible for managing the transaction information on the blockchain.

Definition 4 (Cloaked area): For protecting the true location, a cloaked area generated for $\mathcal{W}_i$ is described as $\langle a_i, f_i \rangle$, in which a_i is a spatial anonymous area based on the true location of $\mathcal{W}_i$ in our proposed algorithm, and then, f_i is the probability density function.

Definition 5 (Target subarea): To maximum the task coverage, we divide the area required by the task into multiple subareas. If the working area of the worker is circular, the dividing interval is an integral multiple of the radius of the worker's working coverage area. Similarly, the whole target is divided into multiple subtargets. Thus, the full coverage of the target area is approximately transformed to achieve the full coverage of subtargets in the subareas. We define the target subarea as t_j and the area as s_j.

If a worker meets working conditions, he can be selected to complete the task. Next, we describe the definitions, including what is a task, how to perform worker selection, what is the task coverage, and what is the task cost.

Definition 6 (Task): The main content in the task information contains $\{Con, Sco, Time, Req_t, Req_w, Bud\}$, in which *Con* denotes the specific content of the task, *Sco* denotes the working scope of the task, *Time* denotes the time limit of the task, Req_t and Req_w, respectively, denote the requirements of the task data and workers, and *Bud* denotes the budget for workers. In addition to the specific content, the task requester must attach its digital signature and public key and then generates a hash digest of all information, forming the broadcast task information

$$Task = \{Con, Sco, Time, Req_t, Req_w, Bud, Sig, pk\}. \quad (1)$$

Definition 7 (Worker selection): Whether a worker is selected can be described as a mapping of worker i to subarea j, which is represented by a matrix $\mathbf{x}$; if $x_{i,j} = 1$, then the worker $i \in N$ is selected for subarea $j \in M$.

Definition 8 (Task coverage): The subarea coverage is defined as the ratio of the working scope to the total task working area, which could be described as $(\sum_{i \in k_j} x_{i,j} s_{i,j})/s_j$, where $x_{i,j} s_{i,j}$ is the working scope and s_j is the total area. Thus, the aggregation of all subareas coverage constitutes the overall task coverage (*TU*), which is shown as follows:

$$TU = \sum_{j \in M} \frac{\sum_{i \in N} x_{i,j} s_{i,j}}{s_j}. \quad (2)$$

We define a coverage goal g for task coverage and the value range is $g \in [0, 1]$.

Definition 9 (Task cost): In this article, we define the Euclidean distance as the task cost model, which is shown in $\mathbf{d}$. Thus, the aggregation of all sensing costs constitutes the overall task cost (*TC*), which is shown as follows:

$$TC = \sum_{j \in M} \sum_{i \in N} x_{i,j} d_{i,j}. \quad (3)$$

Finally, the optimization problem in the preregistration stage and the worker selection stage could be defined as follows.

Definition 10 (Preregistration): After $\mathcal{R}$ published a task, the workers who want to do the task need to register by posting their working information and paying a certain deposit for signing a contract. During the process of preregistration, the local working scope of workers is considered to filter the workers in the same working region to prevent oversaturation. At the same time, global task coverage is also considered. Therefore, we set the control parameters $\mathcal{P}$ and $\mathcal{Q}$ to improve the worker quality in the process of preregistration. $\mathcal{P}$ is the number threshold (the maximum workers in a subregion) of workers in the same subregion and $\mathcal{Q}$ is the multiple of the task coverage goal. All contracted workers may not all appear in the selected set of workers. The deposit of the unselected workers will be sent back.

Definition 11 (WSMC: Worker selection mechanism with cloaked locations): The second-stage selection approach using the idea of spatial location privacy preserving and the optimization approach based on a greedy algorithm is defined as *WSMC*, increasing the data quality by improving the quality of workers and task scope coverage in the blockchain-based crowdsensing model. *WSMC* includes the selection phase (*local worker selection mechanism (LWSM)* and *GWSM*) and the fine-tuning phase. The first phase guarantees to be optimal at the system end. The second phase guarantees to be optimal at the worker end.

Definition 12 (LWSM): Given sets of workers and subareas, each subarea may contain many mobile workers, by the smart contract, and the *LWSM* aims to achieve the local coverage optimization with the minimum task cost by selecting the qualified

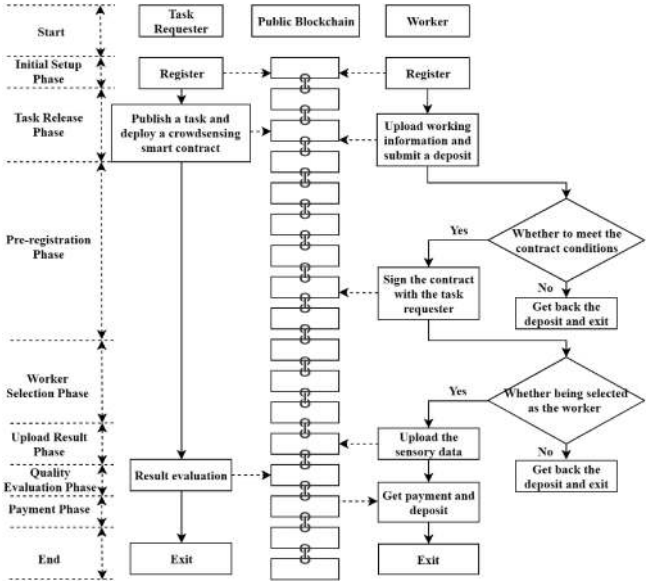

Fig. 2. Execution process of CrowdBLPS.

workers in the subareas to work using their exact locations

$$\max \quad \sum_{j\in M} \frac{\sum_{i\in N} x_{i,j}s_{i,j}}{s_j}$$

$$\text{s.t.} \quad \forall i \in N, \sum_{j\in M} x_{i,j}d_{i,j} \le b_i$$

$$d_{i,j} \ge 0, \quad i = 1,\ldots,n; j = 1,\ldots,m. \tag{4}$$

Definition 13 (GWSM): For the local worker selection result based on the set of divided subareas in a task, we also considered the impact of selected workers on the surrounding areas to adjust the selected workers. We define $\hat{s}_{i,j}$ as the overlapped area with around scopes when $x_{i,j} = 1$, traversing other workers in the subarea; if $\exists x_{m,j} = 1, m \in N$, making $\hat{s}_{m,j} < \hat{s}_{i,j}$, then $x_{m,j} \leftarrow 1$ and $x_{i,j} \leftarrow 0$ (i.e., change the selected worker)

$$\min \quad \sum_{i\in N}\sum_{j\in M} d_{i,j}x_{i,j}$$

$$\text{s.t.} \quad \sum_{j\in M} \frac{x_{i,j}s_{i,j}}{s_j} \ge gS$$

$$\forall i \in N, \sum_{j\in M} x_{i,j}d_{i,j} \le b_i$$

$$d_{i,j} \ge 0, \quad i = 1,\ldots,n; j = 1,\ldots,m. \tag{5}$$

On the basis of local optimization, the result of global worker selection optimization is closer to the actual coverage goal.

C. Phase Description of CrowdBLPS

To better detail the crowdsensing communication process, we have defined seven phases in $CrowdBLPS$ as follows (shown in Fig. 2).

Algorithm 1: Smart Contract Creation in the Public Blockchain.

Input: ID_w—the *ID* of a worker, ID_r—the *ID* of a requester, $Pool_w$—the worker pool, *Time*—the due time of a task.

Output: *Status*—the status of task.

Initializing the value in $Task \longleftarrow ID_t$, $Owner \longleftarrow ID_r$, $Status \longleftarrow Available$, $Reject \longleftarrow False$, and $legal \longleftarrow False$;
$Rewards \longleftarrow FreezeReward(ID_r)$;
if $Rewards < R_{t_i}$ **then**
  $ContractCreate \longleftarrow Failure$;
  return *ContractCreate*
end if
if $Validation(ID_w) = True$ **then**
  $legal \longleftarrow True$;
  $Deposit \longleftarrow FreezeReward(ID_w)$;
  Sign a contract and publish it to the blockchain;
  $Status \longleftarrow UnAvailable$;
end if
if $legal = False$ **then**
  $Reject \longleftarrow True$;
  return *Reject*
end if
$SensoryData \longleftarrow DataUploading()$;
if $CostTime > Time$ **then**
  $Status \longleftarrow Failure$;
  $ID_r \longleftarrow Transfer(Rewards, Deposit, Owner)$;
  return *Status*
end if
if $Evaluation(SensoryData)$ is *Appropriate* **then**
  $Status \longleftarrow Completed$;
  $ID_w \longleftarrow Transfer(Rewards, Deposit, ID_w)$;
else
  $Status \longleftarrow Available$;
  $ID_r \longleftarrow Transfer(Deposit, Owner)$;
end if
return *Status*

1) Initial Setup Phase: $CrowdBLPS$ generates public–private key pairs for the mobile users in crowdsensing, and the private key is saved by the users themselves. The private key will be used to sign in the process of signature.

2) Task Release Phase: The task requester $\mathcal{R}$ broadcasts the specific task content with its signature and public key to the blockchain, which is used to verify the trueness and effectiveness. Simultaneously, the crowdsensing contract, including the task information and worker's execution requirements, will be posted by the task requester to the public blockchain in the form of a transaction, and any worker who meets the contract conditions can sign the contract. To ensure fair trade, the requester creates a smart contract shown in Algorithm 1, which contains the information about a requester, workers, and the task and runs automatically in the public blockchain according to a predefined protocol.

3) Preregistration Phase: After receiving the broadcasting task information, the workers who want to do the task could initiate a transaction containing the working information and a certain deposit to sign the contract. The deposit is used to prevent fraud and will be sent back if the worker fails in preregistration. If the worker succeeds in preregistration, the worker will participate in the process of final selection. Due to the setting of $\mathcal{P}$ and $\mathcal{Q}$, the final selected worker set will be a subset of the preregistering worker set.

4) Worker Selection Phase: Once the process of preregistration is finished, the smart contract that conducts final worker selection will be automatically triggered. If the worker is selected as the final worker, the corresponding deposit will be sent back to the worker. To guarantee the data quality and protect the location privacy, *WSMC* is used to select suitable workers from the preregistering worker set.

5) Upload Result Phase: After completing the sensing task, the worker needs to use the digital signature and public key to upload the sensory result and wait for the result evaluation from the task requester. Considering the storage space problem of the blockchain, we only upload the metadata, and the actual data will be stored in the distributed database. Furthermore, due to the openness and transparency of the blockchain, it is necessary to encrypt the result information with the task requester's public key to prevent plagiarism.

6) Quality Evaluation Phase: When the task requester receives the result information, it evaluates the quality of the data through the quality evaluation method. In our article, the task requester quantifies and normalizes the sensing data and then divides it into two sets, which are qualified and unqualified, according to Req_t in the task information. These two sets reflect the satisfaction levels of the results to Req_t of the task.

7) Payment Phase: If the uploaded data are qualified, the smart contract automatically pays the workers, along with the deposit.

D. Preregistration Stage Optimization

For being selected for completing tasks, any workers could upload their working information and pay a certain deposit to sign a contract with the task requester. As a result, the number of workers signing contracts may be greater than the number of workers actually needed. However, the concentration of workers in hot-spots may result in overcoverage of subregions and failure to reach the coverage goal of the task area. Therefore, we define the control parameters $\mathcal{P}$ and $\mathcal{Q}$ to prevent overcoverage of hot-spots' subregions and guarantee the coverage goal of the whole task area. $\mathcal{P}$ is the maximum number of workers required for a working subregion. According to the time of registration, if the number of workers who choose to work in a certain area a_i exceeds $\mathcal{P}$, then the later workers who choose to work in a_i cannot sign the contract successfully. $\mathcal{Q}$ is the multiple of the task coverage goal, which is set to increase the success rate of worker selection

$$\forall j \in M, \sum_{i \in N} x_{i,j} \leq \mathcal{P}_j, \sum_{i \in N} x_{i,j} s_{i,j} \geq \mathcal{Q} g S. \tag{6}$$

E. Selection Stage Optimization

In *WSMC*, the mobile workers' exact locations are preserved, and the distance metric $\mathbf{d}$ describing mappings between workers and subareas is unavailable to the blockchain. The location uncertainty caused by location privacy preserving has an impact on task cost estimations and subsequently worker selection. Thus, we proposed a two-step optimization solution *WSMC* in the selection stage, WSMC$_s$ and WSMC$_f$. First, WSMC$_s$ is to combine the *LWSM* with the *GWSM* based on uncertain locations. Second, WSMC$_f$ is to fine tune the selection results completed by smart contracts on the blockchain. Next, we will describe each stage in detail and standardize the task objectives.

1) First-Step Optimization (Selection, WSMC$_s$): The optimization objective of the first step is to reduce the impact of location uncertainty from a spatial cloaking algorithm on the worker–target (subarea) pairs, ensuring the coverage quality and preserving the location privacy. If the exact locations of workers are available, the objective would be presented as shown in (4). However, considering location privacy preserving with cloaked areas, it is necessary for us to estimate distances.

2) Second-Step Optimization (Fine-Tune, WSMC$_f$): The second-step optimization will be carried out by the workers themselves on their mobile devices. The smart contract will be triggered after the worker receives the task information, since each worker knows where he is, they can use the exact location to fine tune the results from the first step and choose whether to accept the assigned tasks. If rejecting, to reduce overhead, the system model will only reselect workers within the subarea that the user rejected.

The optimization objective of the second step is shown as

$$\begin{aligned} \min \quad & \sum_{i \in N} \sum_{j \in M} d_{i,j} y_{i,j} \\ \text{s.t.} \quad & |\mathbf{y}_i - \mathbf{x}_i| < \epsilon \\ & \sum_{j \in M} \frac{y_{i,j} s_{i,j}}{s_j} \geq \sum_{j \in M} \frac{x_{i,j} s_{i,j}}{s_j} \\ & \forall i \in N, \sum_{j \in M} y_{i,j} d_{i,j} \leq b_i \\ & d_{i,j} \geq 0, \quad i = 1, \ldots, n; j = 1, \ldots, m \end{aligned} \tag{7}$$

in which for each worker w_i, $\mathbf{x}_i$ and $\mathbf{y}_i$ are the corresponding vectors at different stages, $\mathbf{d}_i$ denotes the travel distance vector, and b_i denotes the worker's travel budget. In the first constraint, $|\mathbf{y}_i - \mathbf{x}_i|$ denotes the Hamming distance between $\mathbf{x}_i$ and $\mathbf{y}_i$, and the small threshold ϵ is used to maintain the results of the first stage. The second constraint represents that the coverage contribution of w_i is at least equal to that in the first stage.

F. Problem Complexity Analysis

In this section, we demonstrate that our proposed two-step approaches *WSMC$_s$* and *WSMC$_f$* are presented NP-hard, by using the minimum set cover problem.

Definition 14 (Minimum set cover problem): Given an arbitrary collection S, a collection $\mathcal{C}$ includes the subsets of S (assuming that $\bigcup_{C\in\mathcal{C}} C = S$) to find the minimum subcollection $\mathcal{A} \in \mathcal{C}$ of S to cover S ($\bigcup_{A\in\mathcal{A}} A = S$).

Theorem 1: $WSMC_s$ is an NP-hard problem.

Proof: We show that the minimum set cover problem for our problem is reduced by a polynomial to prove that $WSMC_s$ is NP-hard. ■

First, consider a set $S = \{w_1, \ldots, w_n, w_\emptyset, t_1, \ldots, t_m, t_\emptyset\}$ and $\mathcal{C}$, which is the subset of S, i.e., $\mathcal{C} = \{\{w_i, t_j\} : w_i \in S, t_j \in S\}$. Then, consider $k > 0$ and $c : \mathcal{C} \to \mathcal{R}_+$, c (the cost function), and $c(\{w_\emptyset, t_j\}) = D$, where $t_j \neq t_\emptyset$ and $D > \sum_{i\in N, j\in M} c(\{w_i, t_j\})$.

Let $W = \{w_i : i = 1, \ldots, n\}$ denote a set of mobile workers and $T = \{t_j : j = 1, \ldots, m\}$ denote a set of targets in the subareas. A distance between t_j and w_i is equal to $d_{i,j} = c(\{w_i, t_j\})$. Assume that $\mathcal{X}_{\text{opt}}$ denotes the optimal solution of $WSMC_s$ with full coverage and minimum travel cost (when $k = 1$ and $g = 100\%$). We define $\mathcal{C}_{\text{opt}} \subset \mathcal{C}$ from $\mathcal{X}_{\text{opt}}$, where t_j is assigned to w_i in $\mathcal{X}_{\text{opt}}$, i.e., $x_{i,j} = 1$; then, $\{w_i, t_j\} \in \mathcal{C}_{\text{opt}}$. If w_i has no target of the subarea assigned to it in $\mathcal{X}_{\text{opt}}$, then $\{w_i, t_\emptyset\} \in \mathcal{C}_{\text{opt}}$. If workers have been assigned to all subareas and each worker has at least one subtarget, i.e., $\mathcal{X}_{\text{opt}}$, then $\{p_\emptyset, t_\emptyset\} \in \mathcal{C}_{\text{opt}}$.

We will prove that any other solution would not be better than $\mathcal{C}_{\text{opt}}$ to cover set S with the minimal cost and full coverage. Assume the existing $\mathcal{C}'$ that covers S with a lower cost. All elements in $\mathcal{C}'$ represent assignment pairs of workers to subtargets, defining an optimization solution $\mathcal{X}'$ for $WSMC_s$. Since $\mathcal{C}'$ has a lower cost than $\mathcal{C}_{\text{opt}}$, $\mathcal{X}'$ has a lower cost than $\mathcal{X}_{\text{opt}}$, which is contradictory, with $\mathcal{X}_{\text{opt}}$ being the optimal solution of $WSMC_s$.

Similarly, $WSMC_f$ can also be proved to be NP-hard (see Definition 13).

IV. Core Algorithm Details in the Blockchain-Based Crowdsensing Model

In this section, we present the algorithm details about preregistration and worker selection in $CrowdBLPS$, which is able to work well in the blockchain-based crowdsensing system. Additionally, we will present the construction of the blockchain and consensus algorithm in detail.

A. First Stage: Preregistration

In order to increase the success rate of task completion and sensory data quality, we added the preregistration stage before the worker selection stage, where the working area of workers and the global task coverage goal are considered. The working information of every worker could contain his/her preferred working scope, and each worker could sign a contract with the task requester if he/she wants to complete the task. Due to the differences in the geographic area, most contracted workers may only appear in certain interested subareas, while some remote subareas may not have workers covered, which are unable to reach the task coverage goal. Data quality also declines due to duplication of hot-spot area data and lack of data in remote areas. Therefore, we proposed the preregistration control algorithm (Algorithm 2) based on the control parameters $\mathcal{P}$ and $\mathcal{Q}$ to guarantee the data quality.

Algorithm 2: Preregistration.

Input: r_i—the preferred working region of worker $\mathcal{W}_i$, x_i—the selected worker matrix, s—the coverage area of worker $\mathcal{W}$, S—the task coverage area, g—the task coverage goal, Num_{R_i}—the number of workers in subregion R_i, $\mathcal{P}$—the worker threshold in a subregion, $\mathcal{Q}$—the multiple of task coverage goal.

Output: y—the worker contracted result.

```
if ∑_{k∈N} x_k s_k < Q g S then
    if Num_{R_i} < P and r_i ∈ R_i then
        Num_{R_i} += 1;
        x_i ⟵ 1;
        Sign a contract between the worker and the task
        requester on the public blockchain;
        return True;
    else
        return False;
    end if
else
    return Completed;
end if
```

B. Second Stage: Worker Selection

In the second stage, we apply two steps to conduct worker selection, including $WSMC_s$ and $WSMC_f$. $WSMC_s$ represents the preliminary selection of workers and $WSMC_f$ represents the fine-tuning process of selection results by workers themselves. Next, we will present the selection algorithms in details.

1) First-Step Stage $WSMC_s$: In the first step of worker selection, we first present two effective methods to solve the uncertainty problem caused by location anonymity in the first stage, and the latter is used in this article. To approximate the optimization objectives for $WSMC_s$, we combine the efficient greedy algorithms proposed in [31] based on the partial set cover problem.

We used the cloaked area to replace the user's exact location in the selection process to receive tasks and utilized a distance-based travel cost model, in which the Euclidean distance is defined as the sensing cost between workers and subtargets. At the same time, the querying algorithms of uncertain spatiotemporal data have been extensively studied; the existing range query, nearest neighbors, top-k, and others are used to propose querying [32].

a) Geometric centroid-point method: As shown in Fig. 3(a), there are a large number of location points evenly distributed in the cloaked area $z \in a$; we calculate the geometric centroid of all points as the expected location of the worker to calculate the expected distance matrix $\hat{\mathbf{d}}$

$$\hat{d}_{i,j} = \text{dis}\left(\int_{z\in a_i} z f_i(z) dz, l_j\right) \tag{8}$$

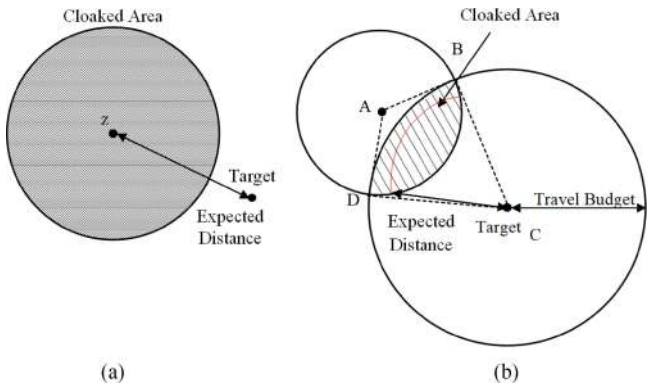

Fig. 3. (a) and (b) Expected distance calculation methods.

where l_j denotes the location of the target j in the subarea and dis denotes the Euclidean distance function.

b) Expected probabilistic method: As shown in Fig. 3(b), for the worker–target pair $\langle i, j\rangle$, we first calculate the probability that the worker i could access to the target in the subarea j as $p_{i,j}$. A simple pruning approach is adapted to shrink the cloaked area a_i to a_i', which is the coincident area between a_i and the circular area centered at target j with traveling radius r_i. Combining f_i, we calculate the probability that a_i' includes the worker i, which is equal to the probability $p_{i,j}$ that the travel scope of worker i includes the target j

$$p_{i,j} = \int_{z \in a_i'} z f_i(z) dz. \tag{9}$$

Finally, based on probability $p_{i,j}$, we calculate the expected distance $\hat{d}_{i,j}$ between the intersection area a_i' and the target as follows:

$$\hat{d}_{i,j} = \frac{\int_{z \in a_i'} \mathrm{dis}(z, l_j) f_i(z) dz}{p_{i,j}}. \tag{10}$$

We perform worker selection by combining the greedy strategy at the first stage, which is usually suboptimal because it makes the best choice at the time at every step. Through algorithm iterations, the most appropriate worker can be selected to work for a subarea, resulting in the most cost-effective worker–target pairs and updating the coverage of subarea targets in real time. The iteration stops (algorithm convergence) once the coverage goal is reached or the worker travel budget is exhausted. For a worker w_i, $i \in N$, and a target t_j, $j \in M$, the cost effectiveness is calculated as follows:

$$\psi_{i,j}^{(1)} = \frac{\hat{d}_{i,j}}{\min(1 - u_j, \frac{1}{k_j}) + \epsilon} \tag{11}$$

where $\hat{d}_{i,j}$ denotes the expected distance and the denominator denotes the expected coverage contributed by w_i. $\mathbf{u}$ denotes the matrix vector of currently covered portions of the subarea targets. The corresponding value of this subarea target in $\mathbf{u}$ will be set to 1 if the subarea target is fully covered ($u_j = \frac{\sum_{i \in N} x_{i,j} s_{i,j}}{s_j}$ and the value range is [0,1]). The expected coverage contributed by worker w_i for target t_j is, hence, $\min(1 - u_j, \frac{1}{k_j})$. Thus, the overcoverage can be prevented, and the small positive value ϵ is added to avoid overflow.

Algorithm 3: Worker Selection.

Input: W—the overall set of mobile workers, T—the overall set of subtargets, **b**—the budget vector, $\hat{d}$—the expected distance matrix, **k**—the coverage-required vector, g—the task coverage goal, p—the access probability matrix, R—the iteration threshold.
Output: **x**—the worker selection matrix, **u**—the target-covered portion vector.

```
Initializing all values to 0 in x, u, TU, and r;
while (TU ≤ gS and r < R) do
  if existing a probable worker–target pair then
    Select suitable worker i for target j with the probability p_{i,j};
    if selected then
      x_{i,j} ⟵ 1;
      TU ⟵ min(1 − u_j, 1/k_j) s_j + TU;
      u_j ⟵ min(1 − u_j, 1/k_j) + u_j;
      b_i ⟵ b_i − d̂_{i,j};
      if u_j = 1 then
        T ⟵ T \ t_j;
      end if
      if b_i = 0 then
        W ⟵ W \ w_i;
      end if
    else
      r ⟵ r + 1;
    end if
  else
    break;
  end if
end while
```

For the first method, the probability is calculated as

$$p_{i,j} = \begin{cases} 1, & \text{if } \hat{d}_{i,j} \le b_i \\ 0, & \text{if } \hat{d}_{i,j} > b_i \end{cases}. \tag{12}$$

Our distance estimation method is based on expected probabilities, and Algorithm 3 is proposed for selecting the most cost-effective pair of worker–target $\langle i, j\rangle$ with probabilities $p_{i,j}$.

We set the upper-bound threshold R (convergence parameter) to stop the algorithm in the expected probabilistic approach, which is serving for experiment purposes and enhanced efficiency. The coverage proportion will be updated in $\mathbf{u}$ at the end of the first step, which is referred to as the expected coverage vector and sent to the workers in the second step.

c) Privacy-preserving analysis: Unlike traditional crowdsensing systems with some true identities in the registration phase, which has the risk of user-sensitive information leakage, $CrowdBLPS$ utilizes the pseudonymous Bitcoin-like addresses to denote task requesters and workers, which enable privacy preserving without submitting true identity to finish

Algorithm 4: Worker Fine Tuning.

Input: w_i—the current worker i and $i \in N$, t—the set of accessible subtargets for w_i, b_i—the travel budgets for w_i, $\mathbf{k}$—the coverage-required vector for subtargets, g—the task coverage goal, $\mathbf{u}$—the target-covered portion vector, R'—the iteration threshold.

Output: $\mathbf{y}_i$—the worker selection matrix for w_i.

```
Initializing all values to 0 in y_i, LTU, STU, and r;
for all subtargets in t do
    u_j ⟵ u_j − x_{i,j}/k_j;
    STU ⟵ STU + x_{i,j}/k_j;
end for
while (LTU ≤ STU and r < R' and b_i > 0) do
    if existing a probable subtarget j in t then
        if selecting j then
            if d_{i,j} < b_i then
                y_{i,j} ⟵ 1;
                LTU ⟵ min(1 − u_j, 1/k_j)s_j + LTU;
                u_j ⟵ min(1 − u_j, 1/k_j) + u_j;
                b_i ⟵ b_i − d_{i,j};
                t ⟵ t \ t_j;
            end if
        else
            r ⟵ r + 1;
        end if
    else
        break;
    end if
end while
```

a crowdsensing task. Additionally, according to the submitted working information, especially the location information, we proposed the location-privacy-preserving approach based on spatial cloaked areas to replace the worker's true location with a corresponding cloaked region for accepting task information, preventing the true locations of workers exposed to the public. Therefore, $CrowdBLPS$ can provide dual protection for identity privacy and location privacy.

d) Time complexity analysis: Assuming the number of workers n, the number of subtargets is m, and the number of continuous sampling points in each cloaked area is s; then, the time complexity of our proposed uncertain distance estimation method is $O(nms)$. For the expected probabilistic method, due to the limitation of the number of iterations R, the time complexity is $O(nmR)$.

2) Second-Step WSMC$_f$: Due to the uncertainty of anonymous locations, the subtargets may not be accessible for the selected worker. Thus, the assigning results need to be fine tuned at the second step using the workers' exact locations while not affecting the overall coverage. However, if each worker simply selects the closest target to save cost, the selected workers will exceed the need of the subarea, which may cause overcovered. Therefore, we proposed additional constraints to limit the overall changes resulting from fine tuning in the second stage.

Algorithm 4 describes the fine-tuned algorithm of worker selection at the second step. Similarly, it iteratively selects the suitable worker w_i for the subtarget with some probability to avoid overcoverage. Different from Algorithm 3, we want to satisfy the first constraint of (7), so any selection changes in $x_{i,j}$ will be penalized. Therefore, the cost-effective score of each selection could be calculated as follows:

$$\psi_{i,j}^{(2)} = \frac{\frac{\hat{d}_{i,j}}{b_i} + 1 - x_{i,j}}{\min(1 - u_j, \frac{1}{k_j}) + \epsilon} \tag{13}$$

which is the ratio of the second-step cost to the expected coverage provided by this worker w_i for the subtarget $t_j \in t$, and the latter is calculated the same as that in the first step. The probabilities that are used to select workers for the task in the second step are also different from those in the first step. For a subtarget j, $p_{i,j}$ is calculated as follows:

$$p_{i,j} = 1 - \frac{\psi_{i,j}^{(2)}}{\max\{\psi_{i,j}^{(2)}\}}. \tag{14}$$

With this probability, our goal is to avoid overcoverage of overall target, but at the same time reduce the chances of costly tasks. Without it, workers will be selected for the subtargets repeatedly until their travel budgets are exhausted. The full budget of all workers may result in excessive coverage of high costs. $\psi_{i,j}^{(2)}$ is used to calculate this probability in favor of a more cost-effective selection. Similar to Algorithm 2, for higher efficiency, we adapted the iteration threshold R' to stop the algorithm.

a) Time complexity analysis: Due to the existence of the upper-bound threshold R', for the current worker with m subtargets, the time complexity will be $O(mR')$.

V. Experimental Results and Analysis

In this section, we will evaluate our proposed two-stage worker selection approaches experimentally using the synthetic datasets. First, we describe the details of experimental settings and then present the experimental results and analysis.

A. Experimental Settings

To analyze the performance of the quality control model $CrowdBLPS$ in this article, we set up an experimental environment based on Ethereum. The software environment is Python 3.5. The hardware environment is 2.60 GHz Core(TM) i7-6700HQ CPU, 20.00-GB, Win10 system of 64bit. The simulation strictly follows the protocols and patterns that may be used by the actual scenario in crowdsensing. Then, we will introduce the used datasets, parameter setting, evaluation metrics, and simulation results.

1) Experimental Datasets: We used the well-known moving objective generator in [33] to generate the synthetic datasets to test the security and reliability of our algorithms under different conditions. Fig. 4 shows the simulation map of Oldenburg.

2) Experimental Parameters: We present the parameter settings for our simulations in Table II, with the default settings highlighted. We set the range of activities for workers to be

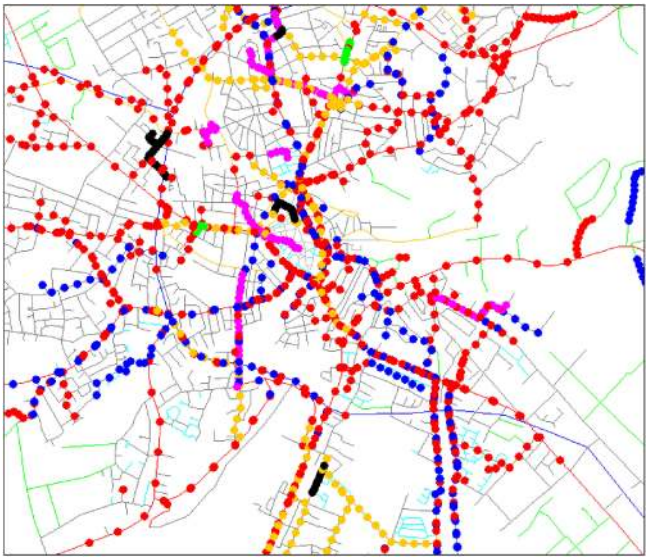

Fig. 4. Workers and regions of a simulation map in Oldenburg of Germany.

TABLE II
PARAMETER SETTINGS

No	Parameter	Value
1	Number of Workers (n)	100, **200**, ..., 900, 1000
2	Number of Sub-Targets (m)	100, **200**, 300, 400, 500
3	Travel Budget (b_i)	50m-**100m**
4	Cloaking Model	**Circular**, Rectangle
5	Cloaking Radius (r)	**12.5%**-37.5% of the map area
6	Coverage Goal (g)	50%-**90%**

around 50–100 m. For simplicity, we experimented with circular areas. The cloaked area for each worker is selected uniformly in the circle with a radius occupying 12.5–37.5% of the map area. The ideal task coverage proportion of each subtarget is 100%, but it is far from being achieved due to the existence of cloaking location, and the range of coverage goal g is from 50% to 90%, with a default value of 90%. The range of R and R' is from 20 to 50. Each experiment is repeated 100 times, and the average is calculated as our final results.

3) Experimental Evaluation Metrics: In the actual sensing process, we first considered the running time of block generation; whether the model can rapidly generate blocks will be a more important factor than the efficiency of the task completion. The running time of generating a block includes the period of Merkle tree generation in the consensus and the new block generation. The running time of the method that measures the proposed worker selection time is also considered to analyze the data.

Then, we considered the success rate and the time cost of the first preregistration stage, which could be affected by the proposed data control parameters $\mathcal{P}$ and $\mathcal{Q}$.

In addition, we proposed the task coverage (*TU*) and the task cost (*TC*) in Section III. In many cases, due to the number of workers and budget constraints, assuming given locations of workers, the expected coverage g may not be achieved. Thus,

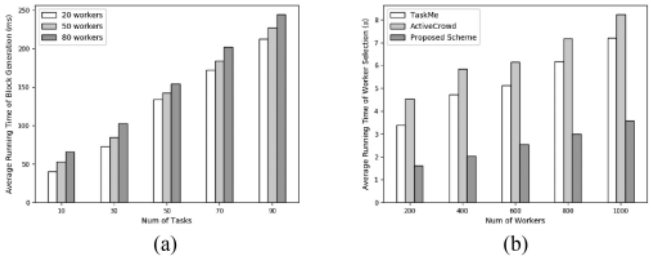

Fig. 5. Average running time. (a) Block generation. (b) Worker selection.

we comprehensively considered the task coverage and cost and proposed new evaluation indicator *PI* (penalized indicator), which is normalized in [0,1] using the min–max method. The smaller the *PI* value, the higher the coverage and the lower the cost, meaning it is better

$$PI = \frac{(g - TU)S}{gS} * \alpha + \frac{TC}{\sum_i b_i} * (1 - \alpha) \tag{15}$$

where α denotes the proportion of task coverage and task cost. Since we considered the importance of task coverage and task overhead, we set α to 50% in our experiments.

4) Experimental Comparisons: We combined different evaluation models and optimization stages (the first/second step of $WSMC_s$ or two-step $WSMC_s$/$WSMC_f$ combination in the second stage) and then proposed several comparison modes as follows.

NPA: It is our two-stage optimization solution without location privacy preserving, which means no privacy constraint, and we defined it as the reference solution. We also assume that the exact locations of workers can be accessible for worker selection.

EPA1: It is the first step of the second-stage method to optimize the local areas.

EPA2: It is the second step of the second-stage method to optimize the local and global areas.

EPA3 (proposed solution): It is our proposed two-stage optimization approach.

B. Experimental Results and Analysis

In this section, we analyze the performance of $CrowdBLPS$ through extensive simulations. We first discuss the running time of block generation and the proposed approach. Then, we mainly discuss the impact of the proposed method on the task coverage and task cost from the following aspects, including the number of workers and subregions, coverage goal, and cloaked radius. The results are presented as follows.

1) Running Time of Block Generation and the Proposed Approach: As shown in Fig. 5(a), with an increase in the number of workers, the average time for generating blocks was increased, but all remain at the millisecond level. The running time of the block generation is mainly affected by the number of workers in the task. The reason is that the larger the number of workers, the larger the Merkle tree in the block.

In order to analyze the performance of the two-stage worker selection approach *WSMC* proposed in this article, we compared

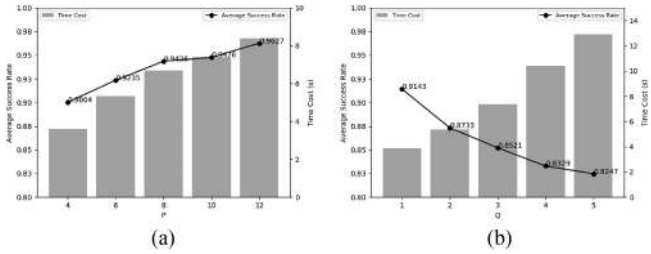

Fig. 6. Average success and time cost for different $\mathcal{P}$ and $\mathcal{Q}$. (a) Increasing $\mathcal{P}$ with $\mathcal{Q} = 2$. (b) Increasing $\mathcal{Q}$ with $\mathcal{P} = 6$.

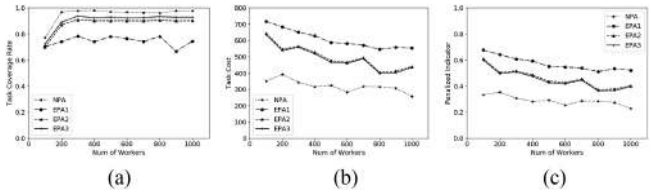

Fig. 7. (a) Task coverage, (b) task cost, and (c) penalized indicator for different workers.

it with other worker selection approaches: one is *TaskMe* [34] and the other is *ActiveCrowd* [35]. Due to different experimental environments, we have retained its core ideas and adapted it to fit our model. Fifty tasks were published to analyze average statistics. As shown in Fig. 5(b), among the three methods, the running time of *ActiveCrowd* is the longest, and the running time of *TaskMe* is slightly shorter than *ActiveCrowd*. The time of our proposed scheme is the shortest, and the magnitude of increase with the number of workers is not as sharp as the other two schemes.

2) Success Rate and Time Cost of Preregistration: The impact on the success rate and time cost of preregistration is shown in Fig. 6. To compare the effect of these two control parameters $\mathcal{P}$ and $\mathcal{Q}$ on the result, we dynamically adjust one of them with the other one fixed. As shown in Fig. 6(a), when $\mathcal{Q} = 2$, the time cost increases with the increase in the value of $\mathcal{P}$, because the maximum threshold $\mathcal{P}$ that accommodates the workers in the subregion increases, and the contract average success rate could reach more than 90%. However, when fixing $\mathcal{P} = 6$ and increasing the value of $\mathcal{Q}$, more workers refuse to sign contracts because of the restriction on the number of workers in the subregions, causing a decline in the average success rate.

3) Impact of the Number of Workers and Subregions: The impact of increasing the number of workers with fixed subregions on task coverage and cost is shown in Fig. 7. Compared to one-step optimization methods (EPA1 and EPA2), our proposed two-step optimization method (EPA3) achieves better result in terms of both task coverage and task cost, which is closer to the result of NPA with no privacy constraint. Since the global optimization of the first stage was taken into account, EPA2 shows a significant improvement than EPA1 in terms of the task coverage rate and efficiency. Based on the fine-tuning optimization in the second stage, EPA3 shows the results closer to the coverage objective. Additional, as shown in Fig. 7(c), increasing

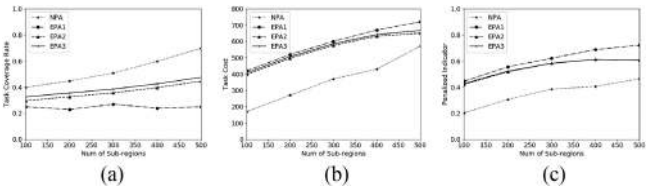

Fig. 8. (a) Task coverage, (b) task cost, and (c) penalized indicator for different subregions.

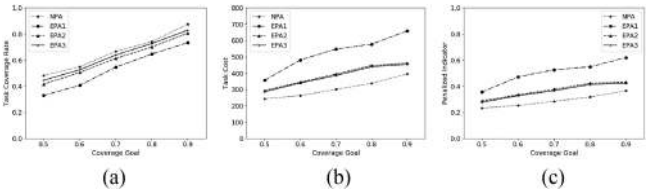

Fig. 9. (a) Task coverage, (b) task cost, and (c) penalized indicator for different coverage goals.

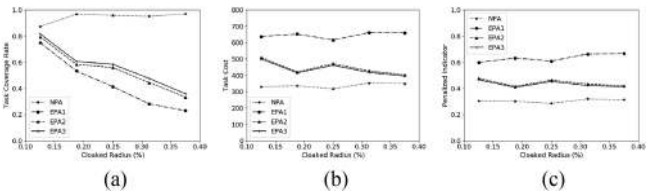

Fig. 10. (a) Task coverage, (b) task cost, and (c) penalized indicator for different cloaked radius percentage.

the number of workers results in a lower penalized indicator, meaning that EPA3 outperforms the other two approaches, i.e., EPA1 and EPA2.

The impact of increasing subregions with a fixed number of workers on task coverage and cost is shown in Fig. 8. Compared to the single-stage optimization methods (EPA1 and EPA2), our proposed two-step optimization method (EPA3) achieves better result in terms of both task coverage and task cost. In addition, EPA1, EPA2, and EPA3 show higher task cost than NPA due to the cloaked location privacy preserving. Similarly, the penalized indicator shown in Fig. 8(c) indicates that EPA3 is more effective and efficient.

4) Impact of Coverage Goal: The impact of increasing coverage goal with a fixed number of workers and subregions on task coverage and cost is shown in Fig. 9. Compared to the one-step optimization methods (EPA1 and EPA2), our proposed two-step optimization method (EPA3) achieves better result in terms of both task coverage and task cost and a penalized indicator.

5) Impact of Cloaking Radius: The impact of increasing cloaked radius with a fixed number of workers and subregions on task coverage and cost is shown in Fig. 10. With the increase in the cloaked radius, except NPA, the task coverage of EPA1, EPA2, and EPA3 is affected to a certain extent; however, EPA3 shows more robustness than the one-step approaches EPA1 and EPA2, indicating that EPA3 is not affected by the cloaked radius as much as the other approaches, while EPA3 outperforms the other approaches for all cloaked sizes.

VI. Conclusion

In this article, we proposed a location-privacy-preserving MCS system $CrowdBLPS$, which integrated the idea of a blockchain into crowdsensing, realizing the decentralization of crowdsensing to avoid the security problems such as repudiation and tempering of information from the traditional centralized crowdsensing system. Following the idea of a smart contract, we proposed a two-stage approach, including the preregistration stage and the final selection stage, based on spatial location privacy preserving and greedy algorithms to protect workers' location privacy and reduce task cost, while achieving the purpose of data quality control in the blockchain-based crowdsensing model. Furthermore, we proved that the optimization problems at each stage were NP-hard. Finally, we implemented the experiments about the average running time of block generation and compared our proposed scheme with the other two schemes and, then, studied the impact of different conditions on the success rate, time, effectiveness, and robustness. The experimental results showed that our proposed approach outperformed the other approaches in terms of operating efficiency, location privacy preserving, and task coverage. In the future, we plan to conduct an in-depth exploration of data quality evaluation to improve data quality and system reliability, making the system model to meet real-world requirements.

Acknowledgment

The authors would like to thank all the reviewers for their helpful comments.

References

[1] K. Abualsaud *et al.*, "A survey on mobile crowd-sensing and its applications in the IoT era," *IEEE Access*, vol. 7, pp. 3855–3881, 2019.

[2] J. Liu, H. Shen, H. S. Narman, W. Chung, and Z. Lin, "A survey of mobile crowdsensing techniques: A critical component for the Internet of Things," *ACM Trans. Cyber-Phys. Syst.*, vol. 2, no. 3, 2018, Art. no. 18

[3] C. Jiang, L. Gao, L. Duan, and J. Huang, "Scalable mobile crowdsensing via peer-to-peer data sharing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 4, pp. 898–912, Apr. 2018.

[4] W. Guo, W. Zhu, Z. Yu, J. Wang, and B. Guo, "A survey of task allocation: Contrastive perspectives from wireless sensor networks and mobile crowdsensing," *IEEE Access*, vol. 7, pp. 78406–78420, 2019.

[5] T. Yan, B. Hoh, D. Ganesan, K. Tracton, T. Iwuchukwu, and J.-S. Lee, "CrowdPark: A crowdsourcing-based parking reservation system for mobile phones," Univ. Massachusetts, Amherst, MA, USA, Tech. Rep., 2011, pp. 1–14.

[6] Y. Liu, L. Kong, and G. Chen, "Data-oriented mobile crowdsensing: A comprehensive survey," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2849–2885, 3Q 2019.

[7] L. Pournajaf, D. A. Garcia-Ulloa, L. Xiong, and V. Sunderam, "Participant privacy in mobile crowd sensing task management: A survey of methods and challenges," *ACM Sigmod Rec.*, vol. 44, no. 4, pp. 23–34, 2016.

[8] S. Basudan, X. Lin, and K. Sankaranarayanan, "A privacy-preserving vehicular crowdsensing-based road surface condition monitoring system using fog computing," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 772–782, Jun. 2017.

[9] M. Thejaswini, P. Rajalakshmi, and U. B. Desai, "Duration of stay based weighted scheduling framework for mobile phone sensor data collection in opportunistic crowd sensing," *Peer-to-Peer Netw. Appl.*, vol. 9, no. 4, pp. 721–730, 2016.

[10] M. Li *et al.*, "CrowdBC: A blockchain-based decentralized framework for crowdsourcing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1251–1266, Jun. 2019.

[11] J. Wang *et al.*, "Learning-assisted optimization in mobile crowd sensing: A survey," *IEEE Trans. Ind. Informat.*, vol. 15, no. 1, pp. 15–22, Jan. 2019.

[12] H. Liu, Y. Zhang, and T. Yang, "Blockchain-enabled security in electric vehicles cloud and edge computing," *IEEE Netw.*, vol. 32, no. 3, pp. 78–83, May/Jun. 2018.

[13] Y. Dai, D. Xu, S. Maharjan, Z. Chen, Q. He, and Y. Zhang, "Blockchain and deep reinforcement learning empowered intelligent 5g beyond," *IEEE Netw.*, vol. 33, no. 3, pp. 10–17, May/Jun. 2019.

[14] B. Zhang, C. H. Liu, J. Tang, Z. Xu, J. Ma, and W. Wang, "Learning-based energy-efficient data collection by unmanned vehicles in smart cities," *IEEE Trans. Ind. Informat.*, vol. 14, no. 4, pp. 1666–1676, Apr. 2018.

[15] J. Xu, S. Wang, B. Bhargava, and F. Yang, "A blockchain-enabled trustless crowd-intelligence ecosystem on mobile edge computing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3538–3547, Jun. 2019.

[16] Y. Xu, J. Ren, G. Wang, C. Zhang, J. Yang, and Y. Zhang, "A blockchain-based non-repudiation network computing service scheme for industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3632–3641, Jun. 2019.

[17] A. W. Colombo, S. Karnouskos, O. Kaynak, Y. Shi, and S. Yin, "Industrial cyberphysical systems: A backbone of the fourth industrial revolution," *IEEE Ind. Electron. Mag.*, vol. 11, no. 1, pp. 6–16, Mar. 2017.

[18] C. H. Liu, Q. Lin, and S. Wen, "Blockchain-enabled data collection and sharing for industrial IoT with deep reinforcement learning," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3516–3526, Jun. 2019.

[19] M. U. Hassan, M. H. Rehmani, and J. Chen, "Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions," *Future Gener. Comput. Syst.*, vol. 97, pp. 512–529, 2019.

[20] Y. Zhang, R. Deng, D. Zheng, J. Li, P. Wu, and J. Cao, "Efficient and robust certificateless signature for data crowdsensing in cloud-assisted industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5099–5108, Sep. 2019.

[21] G. Han, L. Liu, S. Chan, R. Yu, and Y. Yang, "HySense: A hybrid mobile crowdsensing framework for sensing opportunities compensation under dynamic coverage constraint," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 93–99, Mar. 2017.

[22] C. Fiandrino, F. Anjomshoa, B. Kantarci, D. Kliazovich, P. Bouvry, and J. N. Matthews, "Sociability-driven framework for data acquisition in mobile crowdsensing over fog computing platforms for smart cities," *IEEE Trans. Sustain. Comput.*, vol. 2, no. 4, pp. 345–358, Oct.-Dec. 2017.

[23] B. Guo *et al.*, "TaskMe: Toward a dynamic and quality-enhanced incentive mechanism for mobile crowd sensing," *Int. J. Human-Comput. Stud.*, vol. 102, pp. 14–26, 2017.

[24] 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 Things J.*, vol. 6, no. 3, pp. 4711–4718, Jun. 2019.

[25] V. Primault, A. Boutet, S. B. Mokhtar, and L. Brunie, "The long road to computational location privacy: A survey," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2772–2793, Jul.-Sep. 2019.

[26] J. Hua, W. Tong, F. Xu, and S. Zhong, "A geo-indistinguishable location perturbation mechanism for location-based services supporting frequent queries," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1155–1168, May 2018.

[27] C. Yin, J. Xi, R. Sun, and J. Wang, "Location privacy protection based on differential privacy strategy for big data in industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3628–3636, Aug. 2018.

[28] M. Yang, T. Zhu, Y. Xiang, and W. Zhou, "Density-based location preservation for mobile crowdsensing with differential privacy," *IEEE Access*, vol. 6, pp. 14779–14789, 2018.

[29] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," pp. 1–9, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[30] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE Int. Congr. Big Data*, 2017, pp. 557–564.

[31] P. Slavík, "Improved performance of the greedy algorithm for partial cover," *Inf. Process. Lett.*, vol. 64, no. 5, pp. 251–254, 1997.

[32] Y. Wang, X. Li, X. Li, and Y. Wang, "A survey of queries over uncertain data," *Knowl. Inf. Syst.*, vol. 37, no. 3, pp. 485–530, 2013.

[33] T. Brinkhoff, "A framework for generating network-based moving objects," *GeoInformatica*, vol. 6, no. 2, pp. 153–180, 2002.

[34] Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang, "TaskMe: Multi-task allocation in mobile crowd sensing," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2016, pp. 403–414.

[35] B. Guo, Y. Liu, W. Wu, Z. Yu, and Q. Han, "ActiveCrowd: A framework for optimized multitask allocation in mobile crowdsensing systems," *IEEE Trans. Human-Mach. Syst.*, vol. 47, no. 3, pp. 392–403, Jun. 2017.

Shihong Zou (M'19) received the B.E. degree in computer engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 1999, and the Ph.D. degree in communication and information systems from the Beijing University of Posts and Telecommunications, Beijing, China, in 2004.

He is currently an Associate Professor with the School of CyberSpace Security, Beijing University of Posts and Telecommunications. He has authored or coauthored more than 40 papers and applied more than 20 patents. His research interests include mobile security, Internet of Things security, blockchain, and wireless networking.

Jinwen Xi received the M.E. degree location privacy-preserving, blockchain from the Nanjing University of Information Science and Technology, Nanjing, China, in 2018. He is currently working toward the Ph.D. degree in information security with the Beijing University of Posts and Telecommunications, Beijing, China.

He focuses on the security and privacy issues for industrial Internet of Things. His current research interests include blockchain-based big data privacy preserving in the Internet of Things as well as security multiparty computation.

Honggang Wang (M'06–SM'09) received the Ph.D. degree in computer engineering from the University of Nebraska–Lincoln, Lincoln, NE, USA, in 2009.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Massachusetts Dartmouth, Dartmouth, MA, USA, where he is also a faculty member of the Biomedical Engineering and Biotechnology Ph.D. program. He is an affiliated faculty member with the Advanced Telecommunications Engineering Laboratory, University of Nebraska–Lincoln. He was a Member of Technical Staff with Bell Labs Lucent Technologies China from 2001 to 2004. His research interests include wireless health, body area networks, cyber security, mobile multimedia and cloud, wireless networks and cyber-physical systems, and big data in m-Health.

Dr. Wang is an Associate Editor-in-Chief for the IEEE INTERNET OF THINGS JOURNAL.

Guoai Xu received the Ph.D. degree in signal and information processing from the Beijing University of Posts and Telecommunications, Beijing, China, in 2002.

He was awarded the title of Professor by the National Engineering Laboratory of Mobile Network Security and the School of Cyberspace Security, Beijing University of Posts and Telecommunications, in 2011. He is currently an Associate Director with the National Engineering Laboratory of Security Technology for Mobile Internet, School of Cyberspace Security, Beijing University of Posts and Telecommunications. His research interests include software security and data analysis.