IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 20, 2020, accepted December 2, 2020, date of publication December 7, 2020,
date of current version December 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043040

TSWCrowd: A Decentralized Task-Select-Worker Framework on Blockchain for Spatial Crowdsourcing

LIPING GAO[1,2], TIAN CHENG[1], AND LI GAO[1]
[1]School of Optical-Electrical Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China
[2]Shanghai Key Laboratory of Data Science, Fudan University, Shanghai 200093, China

Corresponding author: Liping Gao (lipinggao@usst.edu.cn)

This work was supported by the National Science Foundation of China under Grant 61572325 and Grant 60970012.

ABSTRACT Spatial crowdsourcing is an effective and novel method. In crowdsourcing systems, a centralized platform is traditionally used to allocate tasks and select workers. Centralized platforms always face following challenges: 1) How to ensure the rationality of tasks allocating; 2) How to ensure the payments of workers in the system when dishonest requesters exist; 3) How to ensure the maximum number of tasks are assigned. 4) How to ensure the integrity and reliability of the centralized platform. To solve these problems, this article proposed a distributed blockchain-based crowdsourcing framework – TSWCrowd (Task Select Worker Crowd). In this framework, tasks are sorted according to specific rules, thus tasks with higher priority are assigned to workers earlier. Workers who are available for a task will be selected and return a result. Then the deployed smart contracts will pay the basic payment automatically. At the same time, relevant contracts also calculate and pay the quality payment according to the proposed quality reward formulation. The proposed TSWCrowd framework on-chain involves a public dataset and uses solidity to compile the smart contracts. The framework was deployed on a local private blockchain. The decentralization property of the blockchain ensures the reliable assignment of tasks. Task-select-worker (TSW) algorithm sorts tasks to ensure reliability. In this paper, the proposed framework was compared with the ABCrowd auction mechanism on-chain and the VCG mechanism off-chain. The results show that the average distance is shorter and the payment is higher, thus reaches the reasonability, reliability and availability.

INDEX TERMS Blockchain, crowdsourcing, smart contracts, task management.

I. INTRODUCTION

As a new way of crowd sensing, crowdsourcing has gradually become an important method to handle the complex problems and complete massive tasks by using crowd power and machine learning because of its high efficiency [1]. For example, large-scale translation, information gathering, and even more sophisticated computer programs, such as image annotation. In recent years, platforms such as MTurk [2], CrowdFlower [3], Taskcn [4], TaskRabbit [5] and TopCoder [6] are developing greatly rapidly. In traditional crowdsourcing task assignment, requesters post the task to the crowdsourcing platform, then the workers select the appropriate task according to their interests and execute.

The associate editor coordinating the review of this manuscript and approving it for publication was Moayad Aloqaily.

Spatial crowdsourcing is a new crowdsourcing mode and accelerates the development of platforms such as Uber [27]. Although traditional crowdsourcing platforms can handle complex computational problems, how to ensure the rationality of task assignment is difficult. In this process, workers spend almost the same time to select tasks as completing them, or even more [7]. A good task allocation scheme can help workers to quickly match their own suitable tasks, thus greatly reducing the selecting time. When going to task allocation, the platform will select tasks for workers according to their interest sets in order to provide them with appropriate tasks more accurately. If tasks performed by a worker are accepted by requesters, the worker's reputation was also increased accordingly. It can be seen that task assignment is one of the most important process in crowdsourcing [8].

In traditional crowdsourcing environment, many optimal schemes for task matching are proposed [9] which improved the performance in crowdsourcing based on data analysis. However, during the process of data analysis, the platform recommend tasks based on the information provided by users [10], [11], which means that the crowdsourcing platform is not only a service, it needs to implement additional functions such as user registration, task publishing, task filtering, result collection and salary distribution.

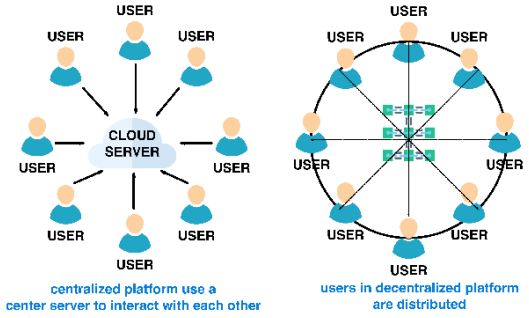

FIGURE 1. Comparison between centralized platform and decentralized platform.

Figure 1 shows the working patterns of the centralized and decentralized platforms in crowdsourcing. In the traditional crowdsourcing platform, it obtains a large amount number of users' private information, such as identity, reputation, interest, etc. We usually assume that centralized platforms are honest and trustworthy, but this may not be the case. The shortcomings of the centralized platforms are obvious, such as task execution is not verification and high intermediary fees. More seriously, if something goes wrong with the platform which leads to server paralysis, the whole crowdsourcing process will break down. Finally, as users' private information is a means of additional monetization, if the crowdsourcing platform is a curious, even malicious, it will use the available information to analyze and sell to other organizations for extra profits.

In order to construct a completely distributed and trusted task allocation scheme in crowdsourcing, we should not pay too much attention to construct a completely honest and uncurious platform because it is impossible. We're going to focus on decentralization. If centralized storage is not used, that is, task allocation and payment distribution in crowdsourcing are not totally executed by a third-party centralized platform, then it is feasible to guarantee the reliability and privacy of task allocation.

Blockchain is a totally distributed ledger that serves as the underlying technology for Bitcoin and Ethereum and is currently popular in both industrial production and academic. Among them, Ethereum, as Blockchain 2.0, realizes the automatic execution of smart contract on the basis of Bitcoin, and has the characteristics of centralization, transparency and anti-interference. And many crowdsourcing platforms based on Blockchain has been proposed to guarantee the decentralization. In the Blockchain, every user has a full copy of the transaction, but unfortunately, when a large number of tasks are issued by requesters in the system, it is still necessary to put forward a reasonable task allocation scheme to allocate tasks.

To ensure the rationality of the task allocation scheme. Some scholars have put forward the scheme of combining Blockchain and auction mode to realize resource allocation [28], which has also been applied to crowdsourcing [29], but in the end, unfair task allocation and reward allocation may be implemented in an untraceable way. This is mainly due to the limited computing scale of Blockchain, and a large number of computing processes are still executed off-chain. [30] puts forward a completely distributed crowdsourcing framework, which executes auctions on Blockchain, and replaces centralized auction platform with intelligent contracts of Blockchain Ethereum. However, after the auction stage, the model only considers the bidding price and the distance between workers and tasks, ignoring the important factors such as workers' reputation and completion time, so the rationality of workers' selection needs to be improved. At the same time, if there are a large number of malicious requesters in the system, the workers will not be paid if they cancel the publishing task after submitting the answers.

In this paper, we design a crowdsourcing task management scheme based on Ethereum: TSWCrowd. The main contributions of this paper are as follows:

- We construct a new crowdsourcing model based on blockchain technology, which uses smart contract running on Ethereum to realize the interaction between requesters and crowdsourcing workers, ensuring the complete decentralization of the crowdsourcing system.
- We deploy TSWCrowd-Task Select Worker Crowd strategy in smart contracts and calculate the payment of the workers uniformly to realize reliable task allocation. At the same time, we divided the payment into two parts: basic payment and quality payment, ensuring workers get payment as long as the complete the task.
- We make comparable experiments to compare with ABCrowd and VCG. Experimental results show that TSWCrowd achieves a reliable task matching with lower cost and gains higher payment for workers.

The rest of this paper are organized as follows. The second part shows the related work and the third part raises the question and gives the correlation algorithm of the system task assignment. The fourth part carries on the frame construction, the fifth part carries on the experiment and makes the result analysis, and the sixth part summarizes the whole article.

II. RELATED WORK

In this section, the TSWCrowd-Task Select Worker strategy is described. There are three related topics in the research: task allocation in crowdsourcing, Ethereum Blockchain and malicious users.

A. BLOCKCHAIN ETHEREUM

Blockchain [18], as a new application model of distributed storage and complete decentralization, was used as an

implementation platform for virtual currency and smart contracts in the past few years. It owns characteristics of transparency, traceability, collective maintenance, openness and transparency has attracted great attention from academia and industry. Its essence is a distributed, non-tamperable ledger. Ethereum, on the other hand, uses smart contracts to enable transactions to be carried out independently and automatically on the blockchain. When conducting transactions, compared with a centralized platform, the blockchain realizes identity privacy protection because users on it use their own public and private keys.

In order to ensure the distributed storage of crowdsourcing system and meet the decentralized nature, [7] puts forward the concept of decentralization in crowdsourcing based on blockchain environment, and some researchers crowdsource in blockchain environment. [34] A trust service mechanism is proposed to transfer assets within the blockchain for payment, which proves that the crowdsourcing system service mechanism without a central institution is more effective and applicable.

However, most blockchain-based systems will suffer from bifurcation and centralization problems. [33] proposes a novel block symbiosis mechanism, which can greatly reduce the computational overhead and realize completely distributed and decentralized management.

To combine blockchain and crowdsourcing, [7] firstly proposed a crowdsourcing framework based on blockchain to use reduce the high service fee and delete the centralized crowdsourcing platform. But in this model, there only exist workers and requesters which is not the crowdsourcing framework.

However, most blockchain-based systems will suffer from bifurcation and centralization problems. [33] proposes a novel block symbiosis mechanism, which can greatly reduce the computational overhead and realize completely distributed and decentralized management. [30] uses the repeated bidder bidding mechanism, the auction algorithm is deployed in the blockchain environment, and the task allocation strategy is optimized. However, the privacy of users is not protected, and only the location factor is considered in the selection criteria of bidding workers, while other important attributes such as completion time are ignored.

However, blockchain technology also faces many challenges, such as 51% attacks [20]. Similar to other blockchain-based crowdsourcing platforms [20], this model also assumes that no organization can control more than 50% of the system.

B. TASK ALLOCATION IN CROWDSOURCING

Jeff Howe in Wired magazine in 2006 argued that crowdsourcing refers to the outsourcing of a task traditionally performed by employees themselves to a large group of people in a public way [12].

Workers in crowdsourcing often take a long time to complete tasks that don't seem to pay much. Therefore, since the concept of crowdsourcing was proposed, task assignment, as a key service of crowdsourcing, has attracted great attention from research institutions and industries. Over the past few years, a lot of researches have been done to improve the reliability and efficiency of task recommendation systems [13], [14]. In some solutions, centralized crowdsourcing platform allocated tasks by analyzing the information collected by workers and requesters. Task matching can recommend a task for workers who are more suitable to complete it. At the same time, a task matching mechanism can also help requesters to obtain satisfactory results without extra operations after uploading tasks. However, due to the needs of queries, which usually contain some privacy-sensitive information, privacy protection is usually ignored. Thus some researches are focused on privacy protection. [15] introduced a trusted third-party platform based on differential privacy to protect workers' location privacy, [16] designed a secure task allocation protocol and introduced a semi-integrity third-party platform that relies on asynchronous distributed task selection algorithms to help users choose their own tasks, but it still assumes that requesters and workers are trading on an honest platform. However, the prerequisites of these two methods of task allocation are the assumption that there is an honest or semi-integrity third-party platform.

Among the existing privacy protection mechanisms, there are some schemes [15], [17], which take into account the protection of workers' privacy, but ignore the privacy protection of task information. For some malicious platforms, the task is combined with the results of task matching. It is not difficult to infer workers' private information from information. To make matters worse, malicious platforms may obtain worker identity information through analysis and then sell this information to obtain additional revenue. It can be seen that due to the existence of a centralized platform, existing solutions cannot guarantee the correctness of the matching results.

At present, most of the crowdsourcing models based on blockchain only allocate tasks according to workers' interest sets [7] [31], which is not a reliable task allocation method without ensuring the reliability of workers' information, that is, there are a large number of malicious workers in the system. [32] The task allocation scheme of the model is to allocate tasks according to the time sequence requested by workers, which does not guarantee the maximization of the rationality of workers' task allocation.

Some crowdsourcing models in blockchain environment combine apply auction mode to allocate tasks, [30] using repeated bidders bidding mechanism, deploying auction algorithm in blockchain environment, optimizing task allocation strategy, but not protecting user privacy, and only considering location factor in bidding selection criteria, ignoring other important attributes such as workers' reputation and compensation. [35] introduced a distributed service to deliver complex problems for cloud users. This scheme doesn't need any intermediary platform. But it is not used in spatial crowdsourcing.

Although the above-mentioned task allocation strategies can satisfy the requirement that tasks are completely allocated, it is impossible to prove that these allocation schemes

are optimal in the rationalization direction due to the lack of comprehensive factors in the formulation of allocation strategies.

C. MALICIOUS USERS

If there exist majority of users who are malicious, the crowdsourcing system will not work very well. For example, if a malicious requester published a large number of tasks, the system will select relevant number of workers to work for those tasks. When workers finished and waiting for the payment, the requester cancel the account and quit from the system. In this case, workers who spend time and energy for those tasks are not paid and those real tasks also may not be completed due to workers are shortage.

A scheme [36] named adaptive boosting for binary classification. But this scheme uses machine learning algorithm. More tasks in the system, the more accurate the prediction is. It is not applicable to systems with a large number of malicious users at the beginning.

Some scholars emphasize the link between MCS and FC. [37] Fog computing is an extension of the concept of cloud computing. It is a distributed framework and used in crowdsourcing and eliminate the bottleneck of data storage and data transmission. But for those malicious users, as it is easy to join in the system, this condition may be worse in crowdsourcing.

A crowdsourcing platform [38] which is trust-worthy only concerned reputation of workers not the requesters and didn't prevent malicious requesters. [39] proposed a reputation-aware crowdsourcing framework based on a SNM model. This scheme increases the platform utility and consider the workers' reputation. But from this point of view, it just cares about the rights of the requesters and ignored that workers are also needed to be protected too.

Comprehensive analysis shows that no framework can not only guarantee the rationality of the task allocation scheme in the blockchain environment but at the same time considering the protection of workers' remuneration when there are malicious requesters in the system. In order to highlight the advantages of BFP-Crowd more obviously, we compare BFP-Crowd with some crowdsourcing models from four dimensions: whether to ensure decentralization, reliable guarantee of workers, optimization of task allocation scheme, and whether the framework consider malicious user. The results are shown in Table 1.

TABLE 1. Comparison table of crowdsourcing system.

	Decentralization	Reliable guarantee	Optimized allocation	Malicious user
CrowdBC[7]	✓	×	×	×
DttCrowd[8]	✓	×	✓	×
GeoCrowd[9]	×	✓	✓	×
OEAT[10]	×	✓	✓	×
multiCrowd[11]	×	✓	×	×
TAGQCrowd[13]	×	✓	×	×
OAERM[29]	✓	×	✓	×
AStERISK[30]	×	×	✓	×
ABCrowd[31]	×	×	✓	×
BPTM[32]	✓	✓	×	×
CrowdBC[33]	✓	×	×	×
FogCrowd[39]	✓	×	×	×
BPCM[34]	✓	×	×	×
VBPTCrowd[40]	×	✓	×	✓
Our scheme	✓	✓	✓	✓

III. TASK-SELECT-WORKER MECHANISM

In this section, we propose a task-select-worker strategy TSW. The traditional crowdsourcing task allocation strategy, whether it is the BPTM, ABCrowd on the chain, or the ordinary centralized crowdsourcing system, it assumes that the requester is honest, and system always assigns a certain number of workers to the crowdsourcing requester to complete the task. But once there is a dishonest requester, it is very likely that workers will complete tasks without payment. In this case, there wouldn't be workers to perform the real tasks, and if a large number of workers in the system perform invalid tasks, the availability of the crowdsourcing system will be reduced. TSW is used to allocate tasks and effectively eliminates this situation. Workers who complete the tasks are paid into two parts: basic compensation and quality compensation. Among them, the basic payment is the basic salary for the worker to complete the task. Regardless of whether the task is finally cancelled in the system, the corresponding basic salary will be allocated to the worker to protect the interests of the worker.

Before proposing the TSW mechanism, some parameters are first declared in Table 2. T is the set of tasks that need to be assigned to workers in the system, which W is the set of workers. For every task $i \in T$, worker $j \in W$, we define:

TABLE 2. Explanation of some symbols.

Mapping Variable	Key	Value
WorkerList(uint[])	TaskID	Worker[]
Rank(uint[])	TaskID	uint[]
Tasks(uint[])	address	Task[]

For a worker selected by a task to complete the task, we assume that the task is completed and he will be paid as:

$$b_{ij} = e_{ij} + p_{ij} \tag{1}$$

Among the formulation,e_{ij} is the basic payment. It is defined by n_i and task i's basic payment rate α:

$$e_{ij} = (b_i * \alpha)/n_i \tag{2}$$

As for the quality payment, it is defined by worker j to task i's answer quality. It assumed that the best answer is the closest to average.

$$q_{ij} = \frac{\frac{\sum_{a=1}^{n_i} f(a) - f(j)}{n_i - 1}}{\frac{\sum_{a=1}^{n} f(a)}{n}} \tag{3}$$

$$f(j) = \frac{n_i \times val(j)}{\sum_{a=1}^{n_i} val(a)} \tag{4}$$

$$p_{ij} = 1 - \frac{n_{ij}}{\frac{(1+n_i)\times n_i}{2}} \times (1-\alpha) \times b_i \tag{5}$$

Algorithm 1 Repeated Allocated Mechanism

a_r = The total number of distributions
n = The number of tasks
T = set of waiting allocated tasks
FT = set of fully allocated tasks
W = The available worker set
n_i = The number of workers that task i need
PW_i = The pending worker set for task i
W_i = The worker set that task i select
FT_j = The maximum tasks for worker j to accept one round

1: **for** *round* $= 1, 2, \ldots, a_r \&\& T \neq \Phi$ **do**
2: **determines** Wi, for every $i \in n$

% Allocation Mechanism

3: sort $\{t1, t2, \ldots, tn\}$ in T in increasing order
4: **for** $i = 1, \ldots,$ n do
5: select PW_i and determines the Wi
6: **if** Wi is full then
7: $FT \leftarrow FT \cup \{ti\}$
8: $T = T \backslash ti$
9: **break**
10: **else if** $PWi < Wi$ then
11: $Wi \leftarrow Wi \cup PWi$

%Payment Function

12: **for** $i = 1, \ldots, n$ **do**
13: **if** $i \in FT$ then
14: **for** $j = 1, \ldots, n_i$ **do**
15: $b_{ij} = e_{ij} + p_{ij}$;
16: **else**
17: **for** $j = 1, \ldots, nW_i$ **do**
18: $b_{ij} = (1 - \alpha) * b_i / nW_i$

The task allocation and payment calculation for TSW mechanism is shown in Algorithm 1.

The distribution strategy is described in lines 3-11. All tasks are sorted according to the publish time. The tasks are divided into fully assigned tasks and not fully assigned tasks. A fully allocated task means that the task has enough workers to perform it, while a task that is not fully allocated indicates that the task still needs workers to perform the task. For each task that is not fully allocated, the set of workers available for the task is first calculated, that is, the set of workers to be determined PW_i and then a set of workers W_i is selected to perform the task. If there are enough available workers to perform the task, the task is assigned to workers, and the current task is added to the set of fully allocated tasks. If the number of available workers is less than the number of workers required by the task currently, all the workers in PW_i are added to W_i.

Payment is calculated in the lines 12-18, for every task, if it is fully allocated. the payment will be set to the workers who complete the task.

The task will be completely allocated when the program terminates ideally. But if there are no available workers in the system, the program will enter an endless loop, so it is possible to set a maximum number of allocations ar. If the tasks are not all allocated after the ar round, the program will be terminated.

IV. PROPOSED FRAMEWORK

Since no platform is completely credible, it is obviously unrealistic for the existing crowdsourcing task allocation mechanism to focus on solving the privacy protection problem by finding a completely honest third-party platform.

In this section, using the blockchain, the proposed TSW mechanism is constructed in a decentralized and credible manner. The first part is an overview of the part of the framework deployed in Ethereum. The second part is the design of smart contracts. The third part is the process description of the framework.

A. OVERVIEW

The TSW framework realized the fully distributed crowdsourcing process by using blockchain Ethereum. TSWCrowd uses smart contracts and is completely deployed on the Ethereum, so there is no need for users to pay additional service fees to a third-party. In addition, due to the transparency and immutability of the blockchain, it provides a transparent and traceable transaction method to users.

Firstly, the requester published the geographical crowdsourcing tasks and send their rewards on Ethereum through smart contracts, and then the smart contracts are used to select workers for the tasks.

Workers publish their Ethereum addresses, longitude and latitude to the crowdsourcing platform, and then the smart contract uses Algorithm 1 to select workers for each task.

B. IMPLEMENTATION IN BLOCKCHAIN

In order to realize the proposed framework, a smart contract is designed through multiple variables, data structures, mappings and functions. The two data structures Task and Worker are shown in Table 3. The task consists of the requester's address, task ID, longitude, latitude, budget, release time, worker list and the number of workers required. Where unit stands for unsigned integer. Worker's data structure includes the worker's Ethereum address, task list, the number of tasks that can be assigned, and the ranking of a certain task.

Mappings are key-value pairs, representing the correspondence between a given keyword and value. As shown in Table 4, WorkerList represents the correspondence between tasks and the worker set, Rank represents the correspondence between tasks and workers' rankings in the task, and *Tasks* represents the correspondence between worker addresses and the assigned task IDs of workers.

Table 5 shows several functions in smart contracts. Among them, *addTask ()* is the function called by the requester to publish a task. The parameters that need to be passed are longitude and latitude of the task, the number of workers

TABLE 3. Data struction in smart contracts.

Task		
Requester(address)	TaskID(uint8)	Longitude(uint)
Latitude(uint)	Budget(uint)	PublishTime(uint)
WorkerList(uint[])	numbers(uint8)	
Worker		
Worker(address)	Tasks(uint[])	numbers(uint8)
Rank(uint[])		

TABLE 4. Mapping relationships.

Mapping Variable	Key	Value
WorkerList(uint[])	TaskID	Worker[]
Rank(uint[])	TaskID	uint[]
Tasks(uint[])	address	Task[]

TABLE 5. Main functions.

Function	Parameters	Return
addTask()	Longitude,Latitude,Budget,numbers	Boolean
allocateWorker()	None	WorkerList
workerSort()	WorkerList	WorkerList
paymentFunction()	None	Workers' Payment

required and the Ethereum address of the requester. It returns a Boolean value, where true represents successful and false means the publish process fails. *allocateWorker ()* allocates workers to tasks in the system, and the result returned is a list of workers to which tasks are allocated. Workers are sorted according to their quality after inputting the worker list. *paymentFunction ()* is to allocate payment to workers who participated in each task.

C. FRAMEWORK PROCESS

The flowchart of this framework is shown in Figure 2, which describes the timeline of interaction between requester and worker through the smart contract. Red represent processes and functions, and black words represent operations initiated by the user.

1) USER REGISTRATION

Requesters and workers need to register in the system before they publish and accept tasks. Each user in the blockchain has a unique Ethereum address to identify their identity, which can effectively prevent the same worker from using different accounts to accept tasks in the crowdsourcing system or the same requester to use it in order not to reduce reputation multiple accounts posted a large number of invalid tasks.

2) CREATING TASK

The requester needs to register an account on Ethereum and obtain an Ethereum address before publishing a task. Then call the addTask () function and pass in the longitude, latitude,

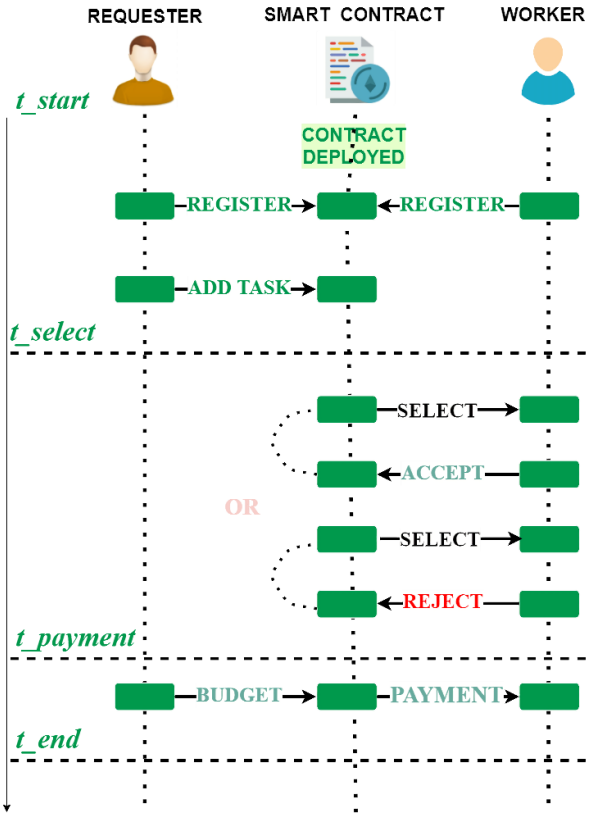

FIGURE 2. Flowchart of TSWCrowd.

and budget, and set its Ethereum address as the requester's address.

3) TASK SELECT WORKER

Algorithm 2 describe the process of task select worker.

For each task, the worker set to be determined is empty, and then analyze the workers one by one. If the worker set of the task is full, the current cycle ends and the next worker is processed. If the worker still has a quota for assignable tasks, the smart contract calculates distance d_{ij} between the worker and the task. If the distance d_{ij} is less than the maximum radius of the task i r_i, j is added to the set of workers to be determined. And if the pending task set equals to the workers that task i needed then ends the loop.

4) REPEATED ALLOCATED MECHANISM

After the first round of task allocation is over, if a task is not fully allocated, the task should be allocated continuously. The maximum number of allocations is ar, if after ar times no matter whether there are still tasks in the system that have not been fully allocated, the program will end. According to Algorithm 1, our smart contract has two functions, an allocation mechanism and a compensation mechanism. The distribution mechanism determines the assignment of tasks to the workers, and the payment mechanism determines and returns the remaining payment to the requester.

Algorithm 2 Task Select Worker

m = number of workers
n = number of tasks
T = set of available tasks
PW = set of pending workers
FT = set of fully allocated tasks
WT = set of tasks a worker can be allocated
nT = number of workers that task need

for $i = 1, \ldots, m$ **do**
  $FT = \Phi$
  $PW = \Phi$
  for j = 1 . . . , m **do**
    if WT is full **then**
      break
  $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
    if $dij < ri$ **then**
      $PW \leftarrow PW \cup \{j\}$
    if $PW = nT$ **then**
      $FT = FT \cup \{i\}$
      break

TABLE 6. Some parameters.

Tasks Parameters	
number of tasks	[50,300]
area(latitude,longtitude)	([34,36], [135,140])
budget	[0.18,...,30]
publish time	[00: 00: 00,...,24: 24: 24]

V. EXPERIMENTS

The TSWCrowd framework we designed is deployed on the chain and has the same performance as the R-SMB auction mechanism, but is better than the offline VCG auction mechanism. In addition, the goal of TSWCrowd is to optimally allocate tasks and save workers' costs.

In order to evaluate these goals, we deploy TSWCrowd on a local private chain and compare them through some experiments.

A. PROGRAM ASSIGNMENT

We applied a real dataset [21]. The relevant parameters of the task set are shown in Table 6. We deploy smart contracts on the local private chain. We use solidity [22] language for programming and use truffle framework [23] and ganache client [24] to deploy smart contracts.

B. REPEATED ALLOCATED MECHANISM-ROUNDS

Before comparing with R-SMB and VCG, we first analyzed the algorithm of TSWCrowd and found that for a crowdsourcing system, the number of rounds allocated by TSWCrowd is related to the maximum radius of each task set.

It can be concluded that when the same number of tasks are allocated in the TSW system, as the task radius increases, the less times it takes to allocate all tasks. When the task radius is the same, if there are more tasks to be allocated, the round part is higher and therefore, the more time it will takes.

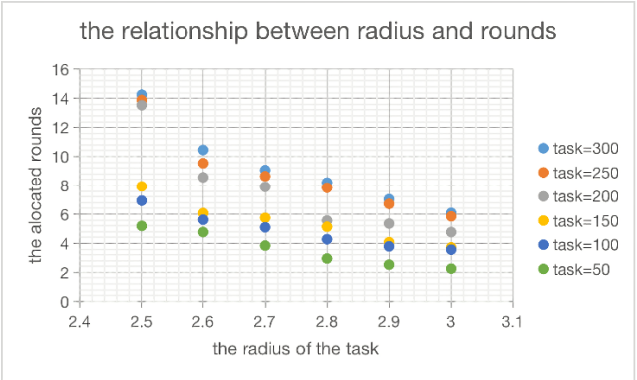

FIGURE 3. The relationship between the maximum number of rounds and the worker radius.

C. FRAMEWORK PERFORMANCE

To compare TSW with VCG and R-SMB from the following two indicators. 1) The total distance traveled by the workers, and 2) The payment for workers.

The first comparison is the total distance between TSW and VCG. Figure 5 shows the total distance of the two frameworks TSW and VCG when the number of tasks is 50, 100, 150, 200, 250.

We can clearly find that the total distance of the workers for TSW is much shorter than that of VCG. When the number of tasks increases, regardless of the model, the total travel path of the workers increases. Similarly, when the total number of tasks increases, the gap between TSW and VCG also increases.

FIGURE 5 shows the payment of workers under the TSW and R-SMB models.

In this experiment, we set the percent of dishonest tasks to 30% and the basic rate of return α to 0.5. In the R-SMB model the reward for dishonest tasks is 0, and workers who accept dishonest tasks will not get any basic payment. In TSWCrowd, the basic payment of workers is guaranteed. Even if the requesters maliciously cancels the task, the worker cannot get the quality payment, he can still get the basic payment.

FIGURE 5 shows that in the case of 50, 100, 150, 200, 250 tasks, there are 30% dishonest workers in the system, and the average payment of R-SMB and TSW. We can clearly conclude that when the cancellation of the tasks is the same, in the TSW model guaranteed by the basic payment, the total average payment for workers is higher.

D. TSW SMART CONTRACT COST

We have demonstrated the feasibility and cost acceptability by deploying the TSWCrowd smart contract on Ethereum. There are 100 workers selected randomly and 50 tasks

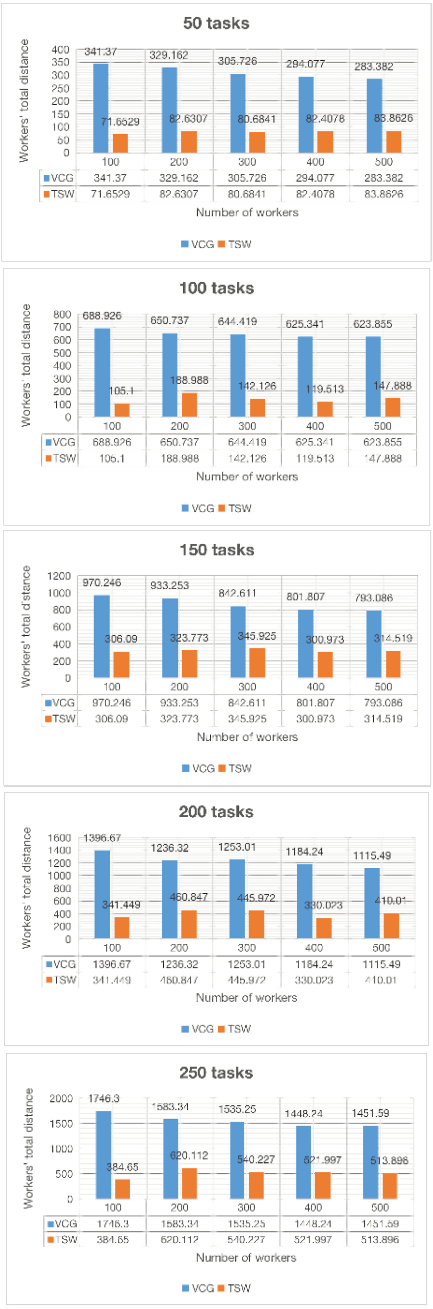

FIGURE 4. comparison of the total distance traveled by workers.

TABLE 7. Consumptions of several main functions.

Function	Gas Consumption	Ether Cost	USD Cost
Deployment	4700000	0.01411	3.2979
Add Task	172699	0.00052	0.1211
TSW(1 round)	5954538	0.01789	4.1687

were invested. The operation cost is calculated by the gas consumption, 1gas is set to 5∗10-9 Ether, the average verification time is 0.8 minute [25].

At the same time 1Ether = $233.02(2020.7.6) [26]. TABLE 7 shows the consumptions of several main functions.

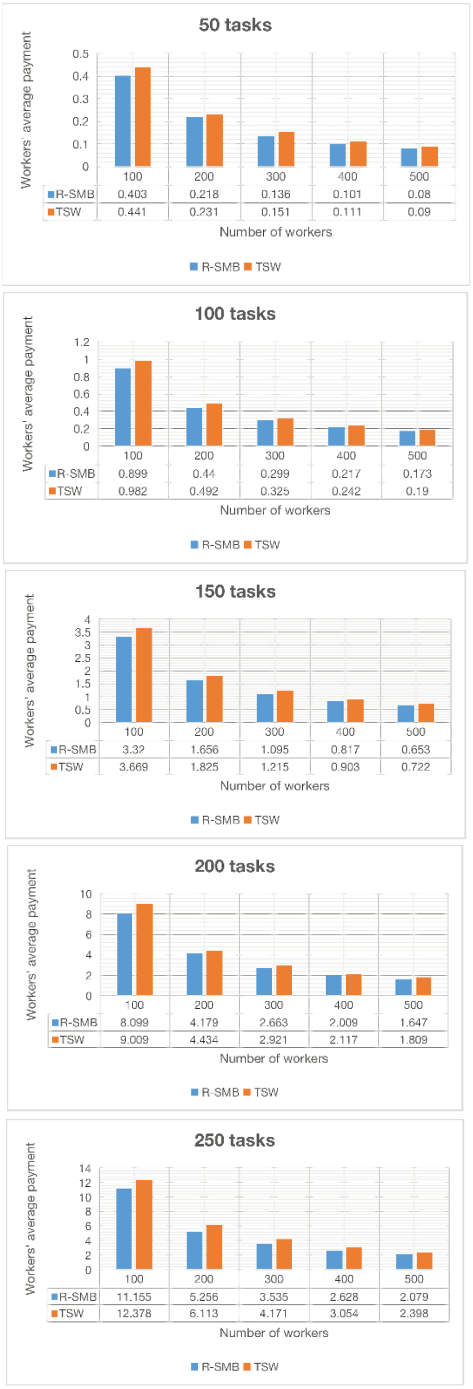

FIGURE 5. The Average Payment for workers between R-SMB and TSW.

Compared with ABCrowd, the cost of TSWCrowd is slightly higher, which is mainly due to the increase of Ether's price.

Only one operation is required when the contract is deployed, and no other expenses are required for maintenance. In *AddTask ()*, the requester spends a part of the cost to publish the task.

As for allocation, the cost for 50 tasks is $4.1687 which is about $0.0833 per task and paid by the requester and

no additional cost is required. Compared with the centralized platform Mechanical Turk (MTurk) [2] which needs to input at least 20% of the cost to the platform to manage tasks, the cost of TSWCrowd to publish tasks is still greatly reduced.

In conclusion, deploying TSWCrowd on smart contracts will significantly reduce the cost of deployment and services compared with traditional centralized crowdsourcing platforms.

VI. CONCLUSION

This paper combines crowdsourcing and blockchain to propose a fully distributed ''task selection worker'' crowdsourcing framework-TSWCrowd. Compared with the traditional centralized crowdsourcing platform, TSWCrowd mainly solves two problems, the reliable distribution of crowdsourcing tasks and the guarantee of worker benefits. And through the use of blockchain technology to achieve transparency and fairness of transactions, transfer invisible off-chain operations to on-chain operations. TSWCrowd requires requesters to pay in advance, and use the feature of blockchain information that cannot be tampered with to ensure workers who perform tasks and submit answers can be allocated basic remuneration, which improves workers' enthusiasm for participating in the system. Compared with the off-chain VCG mechanism, the total travel time of workers is greatly shortened, which proves that TSW is smaller and more reasonable. Compared with the on-chain ABCrowd model, workers' average payment is higher, which proves TSWCrowd is more considerate in ensuring the interests of workers.

In future optimization work, we will focus on task encryption and the searchable encryption part of workers. We will use a new key management system to match the encrypted task keywords of the task with the interest set submitted by the worker, so that the searchable encryption process can further protect the privacy of the worker, combining crowdsourcing with the blockchain model the practicality is greatly improved.

ACKNOWLEDGMENT

The authors would like to thank the reviewers, whose valuable critique and comments helped to improve this paper.

REFERENCES

[1] S. Guan, ''Analysis of optimal pricing model of crowdsourcing platform based on cluster and proportional sharing,'' in *Proc. 6th Int. Symp. Comput. Bus. Intell. (ISCBI)*, Basel, Switzerland, Aug. 2018, pp. 99–103, doi: 10.1109/ISCBI.2018.00029.
[2] (2020). *Amazon Mechanical Turk*. [Online]. Available: https://www.mturk.com/
[3] (2020). [Online]. Available: http://www.crowdflower.com
[4] (2020). *Taskcn*. [Online]. Available: http://www.taskcn.cc
[5] (2020). *TaskRabbit*. [Online]. Available: https://www.taskrabbit.com/
[6] (2020). *TopCoder*. [Online]. Available: https://www.topcoder.com/
[7] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. H. Deng, ''CrowdBC: A blockchain-based decentralized framework for crowdsourcing,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1251–1266, Jun. 2019.
[8] M. H. Cheung, F. Hou, J. Huang, and R. Southwell, ''Distributed time-sensitive task selection in mobile crowdsensing,'' *IEEE Trans. Mobile Comput.*, early access, Feb. 24, 2020, doi: 10.1109/TMC.2020.2975569.
[9] C. Pang, C. Qiu, and N. Wang, ''A geo-obfuscation approach to protect worker location privacy in spatial crowdsourcing systems,'' in *Proc. IEEE 16th Int. Conf. Mobile Ad Hoc Sensor Syst. Workshops (MASSW)*, Monterey, CA, USA, Nov. 2019, pp. 166–167, doi: 10.1109/MASSW.2019.00042.
[10] F. Wang, J. Xu, and S. Cui, ''Ptimal energy allocation and task offloading policy for wireless powered mobile edge computing systems,'' Tech. Rep., 2019.
[11] A. Hu and Y. Gu, ''Mobile crowdsensing tasks allocation for mult-parameter bids,'' in *Proc. IEEE 3rd Inf. Technol. Mechatronics Eng. Conf. (ITOEC)*, Oct. 2017, pp. 489–493.
[12] J. Howe, ''The rise of crowdsourcing,'' *Wired*, vol. 14, no. 6, pp. 1–4, Jun. 2006.
[13] X. Yin, Y. Chen, and B. Li, ''Task assignment with guaranteed quality for crowdsourcing platforms,'' in *Proc. IEEE/ACM 25th Int. Symp. Qual. Service (IWQoS)*, Jun. 2017, pp. 1–10.
[14] L. Ma, X. Liu, Q. Pei, and Y. Xiang, ''Privacy-preserving reputation management for edge computing enhanced mobile crowdsensing,'' *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 786–799, Sep. 2019, doi: 10.1109/TSC.2018.2825986.
[15] H. To, G. Ghinita, and C. Shahabi, ''A framework for protecting worker location privacy in spatial crowdsourcing,'' *Proc. VLDB Endowment*, vol. 7, no. 10, pp. 919–930, Jun. 2014.
[16] M. H. Cheung, R. Southwell, F. Hou, J. Huang, Distributed time-sensitive task selection in mobile crowdsensing, in *Proc. 16th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, New York, NY, USA, 2015, pp. 157–166, doi: 10.1145/2746285.2746293.
[17] Y. Shen, L. Huang, L. Li, X. Lu, S. Wang, and W. Yang, ''Towards preserving worker location privacy in spatial crowdsourcing,'' in *Proc. IEEE Global Commun. Conf.*, San Disco, CA, USA, Dec. 2015, pp. 1–6.
[18] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, J. Wang, ''Untangling blockchain: A data processing view of blockchain systems,'' *IEEE Trans. Know. Data Eng.*, vol. 30, no. 7, pp. 1366–1385, Apr. 2018, doi: 10.1109/TKDE.2017.2781227.
[19] S. Nakamoto. (2009). *Bitcoin: A Peer-to-Peer Electronic Cash System, Cryptography Mailing*, [Online]. Available: https://metzdowd.com
[20] A. Baliga. (2019). *Understanding Blockchain Consensus Models, Persistent*. [Online]. Available: https://pdfs.semanticscholar.org/da8a/37b10bc1521a4d3de925d7ebc44bb606d740.pdf
[21] M. Venanzi, A. Rogers, and N. R. Jennings. (2013). *Crowdsourcing Spatial Phenomena Using Trust-Based Heteroskedastic Gaussian Processes*. [Online]. Available: https://eprints.soton.ac.uk/354861/
[22] (2018). *Solidity*. [Online]. Available: https://solidity.readthedocs.io/en/ latest/
[23] (2020). *Truffle One Click Blockchain*. [Online]. Available: https://www.trufflesuite.com/truffle
[24] (2020). *Ganache One Click Blockchain*. [Online]. Available: https://www.trufflesuite.com/ganache
[25] J. O. Ledyard, ''Optimal combinatoric auctions with single-minded bidders,'' in *Proc. 8th ACM Conf. Electron. Commerce*, 2007, pp. 237–242, doi: 10.1145/1250910.1250945.
[26] (2020). *Ethereum Price*. [Online]. Available: https://ethereumprice.org/
[27] (2020). *UBER*. [Online]. [Online]. Available: https://www.uber.com.cn/
[28] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, ''Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach,'' in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–5.
[29] A. Sonnino and A. Tasiopoulos, ''AStERISK: Auction-based shared economy resolution system for Blockchain,'' Tech. Rep., 2019.
[30] M. Kadadha, R. Mizouni, S. Singh, H. Otrok, and A. Ouali, ''ABCrowd an auction mechanism on blockchain for spatial crowdsourcing,'' *IEEE Access*, vol. 8, pp. 12745–12757, 2020.
[31] Y. Wu, S. Tang, B. Zhao, and Z. Peng, ''BPTM: Blockchain-based privacy-preserving task matching in crowdsourcing,'' *IEEE Access*, vol. 7, pp. 45605–45617, 2019.
[32] X. Xu, Q. Liu, X. Zhang, J. Zhang, L. Qi, and W. Dou, ''A blockchain-powered crowdsourcing method with privacy preservation in mobile environment,'' *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1407–1419, Dec. 2019.

[33] W. Feng and Z. Yan, "MCS-chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain," *Future Gener. Comput. Syst.*, vol. 95, pp. 649–666, Jun. 2019.
[34] L. Tan, H. Xiao, X. Shang, Y. Wang, F. Ding, and W. Li, "A blockchain-based trusted service mechanism for crowdsourcing system," in *Proc. IEEE 91st Veh. Technol. Conf.*, May 2020, pp. 1–6.
[35] I. A. Ridhawi, M. Aloqaily, A. Boukerche, and Y. Jaraweh, "A blockchain-based decentralized composition solution for IoT services," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Dublin, Ireland, Jun. 2020, pp. 1–6, doi: 10.1109/ICC40277.2020.9149031.
[36] M. Simsek, B. Kantarci, and Y. Zhang, "Detecting fake mobile crowdsensing tasks: Ensemble methods under limited data," *IEEE Veh. Technol. Mag.*, vol. 15, no. 3, pp. 86–94, Sep. 2020, doi: 10.1109/MVT.2020.3002522.
[37] D. Belli, S. Chessa, B. Kantarci, and L. Foschini, "Toward fog-based mobile crowdsensing systems: State of the art and opportunities," *IEEE Commun. Mag.*, vol. 57, no. 12, pp. 78–83, Dec. 2019, doi: 10.1109/MCOM.001.1900003.
[38] M. Pouryazdan, B. Kantarci, T. Soyata, and H. Song, "Anchor-assisted and vote-based trustworthiness assurance in smart city crowdsensing," *IEEE Access*, vol. 4, pp. 529–541, 2016, doi: 10.1109/ACCESS.2016.2519820.
[39] B. Kantarci and H. T. Mouftah, "Trustworthy crowdsourcing via mobile social networks," in *Proc. IEEE Global Commun. Conf.*, Austin, TX, USA, Dec. 2014, pp. 2905–2910, doi: 10.1109/GLOCOM.2014.7037249.

LIPING GAO received the B.Sc. and master's degrees in computer science from Shandong Normal University, China, in 2002 and 2005, respectively, and the Ph.D. degree in computer science from Fudan University, China, in 2009. She is currently involved in her research work with the University of Shanghai for Science and Technology as an Associate Professor. Her current research interests include CSCW, heterogeneous collaboration, consistency maintenance, and collaborative engineering.

TIAN CHENG received the bachelor's degree in e-commerce engineering with law from the Beijing University of Posts and Telecommunications in 2016. She is currently pursuing the Ph.D. degree with the University of Shanghai for Science and Technology. Her main research interests include collaborative computing, crowdsourcing, and blockchain.

LI GAO is involved in research work with the Department of Library, University of Shanghai for Science and Technology, as an Associate Professor. Her main research interests include algorithms and process modeling, knowledge discovery, and knowledge mining.