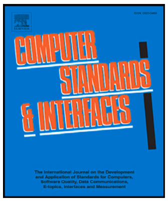
Computer Standards & Interfaces 83 (2023) 103669

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi



# An analysis model for detecting misbehaviors in anonymous cryptocurrency

Shiyong Huang [a], Xin Yang [b], Langyue He [c], Xiaohan Hao [a], Wei Ren [a,d,e,*]

[a] School of Computer Science, China University of Geosciences, Wuhan, China
[b] Wuhan Institute of Marine Electric Propulsion, Wuhan, China
[c] Institute of Information Engineering, Chinese Academy of Science, Beijing, China
[d] Yunnan Key Laboratory of Blockchain Application Technology, Kunming, China
[e] Hubei Key Laboratory of Intelligent Geo-Information Processing, Wuhan, China

## ARTICLE INFO

*Keywords:*
Cryptocurrency
Click farm
Anonymous
Blockchain

## ABSTRACT

In online shopping, consumers often rely on information such as sales, reviews or ratings to inform their decision making. Such preferences or user behaviors can be subjected to manipulation. For example, a merchant can artificially inflate product sales by paying a click farm. Specifically, the click farm will recruit a number of non-genuine buyers to purchase the products. After the purchases have been made, the buyers will either refund the product minus the commission or no product exchange actually takes place and these buyers are paid a commission for their role in the activity. Increasingly due to the popularity of cryptocurrency, such as bitcoin, such payment mechanisms are used in such activities. Hence, in this paper, we seek to detect click farm transactions using cryptocurrency. Specifically, we propose three models to capture click farm operations, and based on the models we design three algorithms to detect anonymous click farm transactions. Extensive analysis demonstrates that our model achieves a high accuracy rate in detecting anonymous click farm transactions, without incurring expensive computational costs.

## 1. Introduction

Click farm is a cheating manner by artificially inflating or manipulating the number of sales transactions, with the aim of making products or services appear more popular than they actually are. In such activities, dishonest merchants will reach out to click farm operators, who will then employ a large number of non-genuine online buyers to manipulate/exaggerate the sale data and forge user reviews. These buyers, also referred to as “click farm workers”, can use compromised or fake accounts, or their own accounts to carry out the activities. In other words, both merchants and click farm operators collude to cheat customers.

There exist three entities in a click farm ecosystem, namely: some merchant(s), delegations for click farm operators, and the actual click farm workers. Delegations employ a large number of click farm workers, who are paid typically based on commissions. Merchants tend to look for such delegations to increase the sales of their products and/or obtain a higher rating on review websites, as discussed earlier. First, the merchants need to apply for a click farm service with the delegation. Once the delegation accepts the merchant’s application, workers will be tasked to purchase the target product and provide a positive review. When the merchants’ products are in the top sales category (e.g., as agreed in their contract), they will pay the agreed financial compensation to the delegation, and the relevant commission will be paid to the workers.

With the increasingly popularity of cryptocurrencies, they have been (ab)used to pay the click farm operators and workers. According to https://coinmarketcap.com/, as of April 28, 2020, there are 5,397 cryptocurrencies in 21,438 markets, with a market capitalization of $223,505,803,308. Of these cryptocurrencies, Bitcoin is the most popular with a market capitalization of $142,516,713,078. In a bitcoin network, no trusted third-party is required and transactions will be verified through a consensus mechanism. Verified transactions will then be stored in blocks, and the transaction ledger is transparent, immutable, and traceable.

To better understand the challenges involved in the investigation of anonymous cryptocurrency-facilitated click farm activities, we examine the characteristics of such transactions between different Bitcoin accounts. We then propose a basic model, an enhanced model and an advanced model to describe the click farm ecosystem. We also design three algorithms to detect click farm activities based on the advanced model. Specifically, the contributions of this paper are listed as follows:

- We designed a series of models incrementally for detecting click farm in terms of graph theory, which facilitate for illustrating the complex relations of click farm.

[*] Corresponding author at: School of Computer Science, China University of Geosciences, Wuhan, China.
*E-mail address:* weirencs@cug.edu.cn (W. Ren).

https://doi.org/10.1016/j.csi.2022.103669
Received 29 March 2022; Received in revised form 29 May 2022; Accepted 30 June 2022
Available online 6 July 2022

- Based on the advanced model of click farm, the ring structure for information exchange between merchant, delegation and workers, the time threshold and the probability threshold are proposed to design three algorithms to detect click farm.

The rest of the paper is organized as follows. Section 2 reviews the related literature. Section 3 describes basic model, enhanced and advanced model. Sections Section 4 illustrates the proposal and implementation of our scheme. The proposed approach is then evaluated and the findings presented in Section 5. Finally, we conclude this paper in Section 6.

## 2. Related work

### 2.1. Cryptocurrency

A number of studies have focused on the characteristics, factors that facilitate/enable adoption, potential application domains, security and privacy challenges, and other topics relating to cryptocurrencies. For example, in a typical anonymous cryptocurrency system, credibility and security are two key issues. There have also been interest in identifying vulnerabilities and exploiting such vulnerabilities in the underpinning cryptocurrency infrastructure (e.g., wallets) to acquire the cryptocurrencies stored in the system or evidence to facilitate digital/forensic investigations [1–3].

Hence, a number of security measures have been presented in the literature. For example, Rezaeighaleh, et al. [4] proposed a new scheme for creating sub-wallets to resist man-in-the-middle attacks, and achieve both backup and restoring functionalities. In a separate work, He et al. [5] proposed an effective social-network-based cryptocurrency wallet management, which realizes time sharing authorization through the hierarchical key isolation encryption scheme. There have also been attempts to design privacy-preserving solutions, as noted in literature survey of Herskind et al. [6]. As we discussed earlier, efforts have also been devoted to understand factors that influence adoption of cryptocurrency, such as in the studies of Schaupp et al. [7]. Based on the identified factors, the authors proposed a model to determine when cryptocurrencies are accepted [7]. Besides, Makarov et al. [8] decomposed the signature volume on each switch into a common and special component, which can explain 80% of bitcoin returns, in order to reduce arbitrage opportunities. Liu et al. [9] proposed a method to detect the silent mining behavior of the browser. By modifying the kernel code of Chrome, a browser-based silent miner detection prototype system BMDetector was designed and implemented. Their method can drive known malicious mining samples, extract heap snapshots and stack code functions of dynamically running browsers, and perform automatic detection based on recurrent neural networks. In addition, the experimental results show that the recognition rate of original mining samples is 98%, and the recognition rate of encrypted and confused samples is 92%, which prove efficiency and flexibility of their method. Li et al. [10] proposed a decentralized group signature-based method and a verifiable encryption-based method that can detect suspicious transactions, thereby solving the problem of insufficient supervision of existing blockchain-based cryptocurrencies. Zhang et al. [11] proposed linkable group signatures for signing cryptocurrency transactions to prevent some cases of user misbehaving. Li et al. [12] solved the problem of user traceability in Monero by introducing a new cryptocurrency called Traceable Monero. To address the fairness issues of the self-counting system, such as adaptation and miscarriage caused by malicious voters, Li et al. [13] proposed a blockchain-based framework for self-counting systems and demonstrated that their scheme satisfies fairness, non-controversy, and maximum vote confidentiality. To solve the problem that the privacy of electronic learning records cannot be guaranteed in massive open online courses, Li et al. [14] proposed a blockchain-based secure storage and sharing scheme for electronic learning records, which supports efficient conditional anonymity, safety sharing, and secure storage.

There are also many types of attacks in the cryptocurrency and blockchian field [15]. Encryption hijacking is a secret attack in the user's browser, that is, hackers implant mining scripts into vulnerable pages to make huge profits. In order to resist encryption hijacking, Petrov et al. [16] provided a novel detection method, coinpolice, which can reverse the control of the encryption hijacker and artificially change the CPU power of the browser to observe whether there is control. The experimental results show that coinpolice can detect 97.87% of hidden miners. In addition, they identified 6700 sites that benefited from traffic by coinpolice. Kharraz et al. [17] proposed an automatic password hijacking detection system named Outguard, which implements 97.9% TPR and 1.1% FPR, and can reasonably tolerate adversarial evasion. In addition, they found 6302 password hijacking sites by deploying Outguard on Alexa Top 1M websites. By taking advantage of the inherent characteristics of a set of password hijacking scripts, Hong et al. [18] a behavior-based detector named CMTracker, which can automatically track cryptocurrency mining scripts and their related domains. They also found 2770 unique cryptojacking samples from 853,936 popular web pages. Yulianto et al. [19] used the Taint analysis method to build the password protection mitigation measures inside the browser as an extension of the Google Chrome browser and used man-in-the-middle (MITM) attack as an attack modeling mitigation test in case of abuse, which users will be notified to check the features of the script running on the website background when password hijacking attacks occur. The results of their study indicate that taint analysis is a promising method to mitigate cryptojacking attacks. They conducted a comprehensive analysis of Alexa's top 1 million websites to reveal the prevalence and profitability of cryptojacking attacks. In order to identify 20 active cryptomining campaigns, Konoth et al. [20] studied 28 Coinhive-like services and studied possible countermeasures against cryptojacking attacks. They proposed a new detection technology MineSweeper based on the inherent characteristics of cryptomining code. Besides, their method can be integrated into the browser to warn users not to use silent password mining when visiting websites without their consent. In order to resist XSS-assisted attacks and web gadget-exploiting counterfeit mining, Wang et al. [21] proposed a browser-based password mining method based on semantic signature matching to detect and interrupt unauthorized access. The evaluation results show that their method is more robust than existing static code analysis defenses, which are vulnerable to code obfuscation attacks. They also provided a browser-independent deployment strategy based on the implementation of inline reference monitoring, which was suitable for general end-user systems without dedicated hardware or operating systems. Eskandari et al. [22] examined the recent trend of mining cryptocurrency in browsers through Coinhive and similar code base mining Monero and conducted some measurements to determine its penetration and profitability, outlined an ethical framework for considering whether it should be classified as an attack or business opportunity, and provided some suggestions on the detection, mitigation and prevention of browser-based mining by non-consensual users.

### 2.2. Click farms

There are a number of ways that dishonest merchants can attempt to influence the sales of their products or services, for example by artificially inflating their sales and reviews as discussed earlier. Hence, there have been attempts to design detection and mitigation strategies. For example, Xie et al. [23] constructed a multidimensional time series anomaly detection algorithm based on curve fitting. The authors reported that their evaluation findings suggested the algorithm to be effective in detecting single-user comment spam.

Detecting the true identity of social network users is an important step in mitigating such fraudulent activities [24–26]. For example, Singh et al. [27] used machine learning techniques such as Bayesian networks, logistic regression, and random forests on Twitter to classify

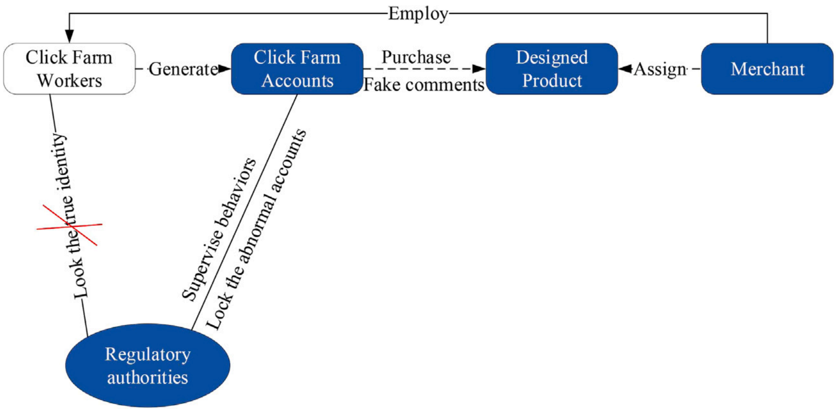


**Fig. 1.** An example on why conventional click farm detection approaches cannot be applied to blockchains.

users based on trust, users, and content-based features. Then, the authors proposed a system to facilitate spam detection, by attempting to identify these users. Taking a different approach, Boshmaf et al. [28] focused on detecting fake accounts in online social networks, since they argued that such accounts are often created and used to perform fraudulent activities. Cao et al. [29] also designed a detection system. The system clusters user accounts according to the similarity of their actions, using user activity data in online social networks. They evaluated their detection system using execution logs from Facebook in August 2013, but failed to detect malicious accounts in real-time. Li et al. [30] conducted a three-stage method to detect click farming. They clustered the communities based on the newly defined collusion network and applied Louvain community detection method to community detection. Finally, they performed a binary classification of the detected communities. In order to identify click farming on Taobao, a China's largest online shopping platform, Jiang et al. [31] adopted a positive unmarked learning method to find reliable negative examples from the unmarked set, and output the click farm recognition probability levels of all stores, from the product and The online store has created several functions. They also use weighted logit model to study the role of the extracted features in segmented click agriculture. Their research can help reduce the risk of online consumers being deceived and improve the platform's ability to monitor click farming.

Another mitigation strategy is to trace the financial trails / transactions, such as those on the distributed ledger [32–34]. For example, Li et al. [35] proposed using a phased approach to detect click farming, and they found that click farming ecosystem has relatively tight relations between users and more highly-ranked stores have a greater portion of fake reviews.

We observe that the investigation of anonymous cryptocurrency-facilitated click farm activities is an understudied area, and hence this is so we presented our click farm model. Based on the advantages of blockchain [36–39], we also proposed 3 blockchain-based detections for click farm after summarizing the behavior of workers.

## 3. Problem statement

Blockchain-based ledgers are open, transparent, traceable, chronological, and tamper-proof [40]. A transaction is a data structure comprising information about the sender, receiver, and Bitcoin transfer. Using Unspent Transaction Output (UTXO), the transaction constitutes a chain structure.

Fig. 1 illustrates why conventional detection of click farm activities cannot be applied to blockchains. For example, typically abnormal accounts exhibit different behavioral characteristics, which can be used to facilitate detection and mitigation approaches. For example, once these accounts are flagged as suspicious or abnormal, they can be locked and the associated user's permissions limited. In addition, tracing of these accounts can also lead to the identification of click farms. However, on a blockchain-based system, users can generate a large number of anonymous accounts, in order to hide their identity and circumvent detection approaches. This will limit the effectiveness of conventional click farm detection approaches. Hence, this necessitates the design of a new detection mechanism.

**Table 1**
Summary of notations.

| Notation | Description |
|---|---|
| $V$ | Entity set |
| $E$ | Edges set |
| $V_{Mc}$ | Merchants set |
| $V_{Cf}$ | Click farm workers set |
| $V_{Dl}$ | Delegations set |
| $E_{Cf}$ | Click farming edges set |
| $E_{Pcft}$ | Posting click farm tasks edges set |
| $E_{Dcft}$ | Distributing click farm tasks edges set |
| $E_{Rrt}$ | Returning click farm results edges set |
| $E_{Rrd}$ | Returning reward edges set |
| $CFG$ | Click farm model |

Specifically, we will explain in this paper how we can extract information such as transaction time, transaction amount, transaction sender, receiver collection, and other product related information (e.g., sales and rank), to facilitate click farm activity detection.

### 3.1. Proposed models to model click farm operations

In order to explain click farm operations intuitively, we propose three click farm models (see Sections 3.2 to 3.4) which are then leveraged by our algorithms (see Section 4) to detect click farm activities.

A summary of notations is presented in Table 1.

### 3.2. Basic model

In this basic model, merchants wishing to increase the sales of their products and/or obtain more positive reviews will generate a large number of accounts themselves. These accounts are then used to facilitate these fraudulent activities (e.g., make fake purchases and/or post fake reviews) — see also Fig. 2.

In the model, let $CFG ::= \langle V, E \rangle$, where $V$ is a set of vertexes and $E$ is a set of edges; $V = V_{Mc}$, where $V_{Mc}$ is a set of merchants; $E = E_{Cf}$, where $E_{Cf}$ is a set of click farming edges; $E_{Cf} ::= \langle V_{from}, V_{to} \rangle, V_{from} \in V_{Mc}, V_{to} \in V_{Mc}$; and $\forall e \in E_{Cf}$, $e = \langle from, to \rangle$, where $from \in V_{Mc}, to \in V_{Mc}$.

Such a basic model is relatively inefficient, and may not be sustainable particularly for small-sized merchants (e.g., due to the time costs involved in creating the fake accounts and using the accounts to make fake purchases and constantly post reviews).

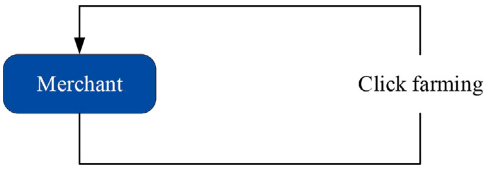


**Fig. 2.** Basic click farm model.

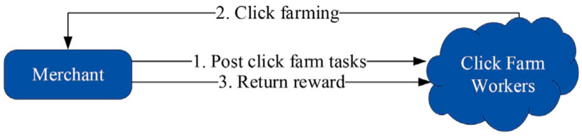


**Fig. 3.** Enhanced click farm model.

### 3.3. Enhanced model

In this enhanced model, the dishonest merchant outsources the fraudulent activities to a group of users. In other words, unlike the single-entity basic model, this is a dual-entity model comprising merchants and click farm workers (see also Fig. 3).

The notations in this model are as follows:

- $CFG ::= \langle V, E\rangle$, where $V$ is a set of vertexes and $E$ is a set of edges.
- $V = V_{Mc} \cup V_{Cf}$, where $V_{Mc}$ is a set of merchants and $V_{Cf}$ is a set of Click farm workers.
- $E = E_{Cf} \cup E_{Pcft} \cup E_{Rrd}$, where $E_{Cf}$ is a set of click farming edges, $E_{Pcft}$ is a set of posting click farm tasks edges, and $E_{Rrd}$ is a set of returning reward edges.
- $E_{Cf} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Cf}, V_{to} \in V_{Mc}$.
- $E_{Pcft} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Mc}, V_{to} \in V_{Pf}$.
- $E_{Rrd} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Mc}, V_{to} \in V_{Cf}$.
- $\forall e \in E_{Cf}$, $e = \langle from, to\rangle$ where $from \in V_{Cf}$, $to \in V_{Mc}$.
- $\forall e \in E_{Pcft}$, $e = \langle from, to\rangle$ where $from \in V_{Mc}$, $to \in V_{Pf}$.
- $\forall e \in E_{Rrd}$, $e = \langle from, to\rangle$ where $from \in V_{Mc}, to \in V_{Pf}$, or $from \in V_{Pf}$, $to \in V_{Cf}$.

The interactions between $V_{Mc}$ and $V_{Cf}$ in the model are described below:

1. The merchant in $V_{Mc}$ hires a group of click farm workers from $V_{Cf}$ and negotiates with them the terms of engagement (e.g., tasks and payments). Then, the merchant posts the agreed upon tasks through $E_{Pcft}$.
2. Workers work on the agreed upon tasks.
3. Upon completion of the agreed upon tasks, the merchant will pay out the rewards through $E_{Rrd}$.

Compared with the single-entity model, this dual-entity model is more efficient and the merchant's workload is substantially reduced.

### 3.4. Advanced model

To further increase efficiency and reduce the need to engage with many workers, the dishonest merchant can outsource the tasks to a click farm operator, as illustrated in Fig. 4. This three-entity model comprises merchants, delegations and workers.

The notations in this model are as follows:

- $CFG ::= \langle V, E\rangle$, where $V$ is a set of vertexes and $E$ is a set of edges.
- $V = V_{Mc} \cup V_{Cf} \cup V_{Dl}$, where $V_{Mc}$ is a set of merchants, $V_{Cf}$ is a set of Click farm workers and $V_{Dl}$ is a set of delegations.

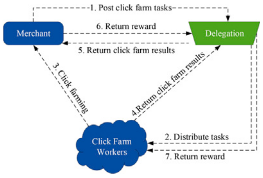


**Fig. 4.** Advanced click farm model.

- $E = E_{Cf} \cup E_{Pcft} \cup E_{Dcft} \cup E_{Rrt} \cup E_{Rrd}$, where $E_{Cf}$ is a set of click farming edges, $E_{Pcft}$ is a set of posting click farm tasks edges, $E_{Dcft}$ is a set of distributing click farm tasks edges, $E_{Rrt}$ is a set of returning click farm results set. $E_{Rrd}$ is a set of returning reward set.
- $E_{Cf} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Cf}, V_{to} \in V_{Mc}$.
- $E_{Pcft} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Mc}, V_{to} \in V_{Dl}$.
- $E_{Dcft} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Dl}, V_{to} \in V_{Cf}$.
- $E_{Rrt} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Cf}, V_{to} \in V_{Dl}$ or $V_{from} \in V_{Dl}, V_{to} \in V_{Mc}$.
- $E_{Rrd} ::= \langle V_{from}, V_{to}\rangle, V_{from} \in V_{Mc}, V_{to} \in V_{Dl}$ or $V_{from} \in V_{Dl}, V_{to} \in V_{Cf}$.
- $\forall e \in E_{Cf}$, $e = \langle from, to\rangle$ where $from \in V_{Cf}$, $to \in V_{Mc}$.
- $\forall e \in E_{Pcft}$, $e = \langle from, to\rangle$ where $from \in V_{Mc}$, $to \in V_{Dl}$.
- $\forall e \in E_{Dcft}$, $e = \langle from, to\rangle$ where $from \in V_{Dl}$, $to \in V_{Cf}$.
- $\forall e \in E_{Rrt}$, $e = \langle from, to\rangle$ where $from \in V_{Cf}$, $to \in V_{Dl}$, or $from \in V_{Dl}, to \in V_{Mc}$.
- $\forall e \in E_{Rrd}$, $e = \langle from, to\rangle$ where $from \in V_{Mc}$, $to \in V_{Dl}$, or $from \in V_{Dl}$, $to \in V_{Cf}$.

The interactions between $V_{Mc}$, $V_{Cf}$ and $V_{Dl}$ are as follows:

1. The merchant contacts the delegation from $V_{Dl}$ and issues click farm tasks through $E_{Pcft}$
2. The delegation then distributes tasks to workers through $E_{Dcft}$, $V_{Cf}$, where a large group of workers are hired to complete the tasks.
3. The workers work on their agreed upon tasks.
4. The workers complete the tasks and report back to the delegation.
5. The delegation reports the collective task completion to the merchant.
6. The merchant pays out the reward to the delegation.
7. The delegation pays out the reward to the workers.

## 4. Proposed scheme

### 4.1. Proposed click farm detection algorithms

In this section, we will present our proposed algorithms to detect click farms for all three models described in the preceding section. Since the advanced model is the most complex, we will discuss our algorithms in the context of the advanced model.

In a typical click farm operation, the merchant, click farm workers and delegations will form a circular structure due to information exchange between both merchants and delegation, between delegation and workers, and between workers and merchants. In other words, we can extract useful information from the ring structure between the three entities and use it in the algorithm. Typically, timeliness is crucial in artificially inflating product ranking and popularity. Hence, the time to task completion by the workers is short. Here, a time threshold (see Algorithm 2) can be set to identify click farm accounts from normal user accounts. In addition, the sales volume of the merchant's product is generally not at the top of the rankings, prior to the fraudulent activities. When the product's ranking or popularity increases, this is most likely due to the click farm workers' actions. Thus, a probability threshold (see Algorithm 3) can be set in the algorithm. Specifically, our detection algorithm (see Algorithm 1) leverages both Algorithms 2 and 3.

### 4.2. Ring structure based click farm detection algorithm

Algorithm 1 deals with the transactional information between merchants and delegation, delegation and workers, and workers and merchants, during click farm operations.

```
Data: $Merchant[.], Worker[.], Customer[.][.], i = 0, j = 0, Transaction[.]$
Result: Worker set $C$
for each $m \in Merchant$ do
    for each $R1 \in Transaction$ do
        for each $R2 \in Transaction$ do
            if $R1 == R2$ then
                $continue$;
            end
            if $R1.to == m \bigcap R2.to == R1.from \bigcap R1.money < R2.money \bigcap R1.time < R2.time$ then
                $Customer[i][j] = R1.from$;
                $j = j + 1$;
            end
        end
    end
    $i = i + 1$;
end
for each $C \in Customer$ do
    if $C \neq null$ then
        $Worker = C$;
        $break$;
    end
end
if $Worker \neq null$ then
    for each $C1 \in Customer$ do
        if $C1 \neq null$ then
            $Worker = Worker \bigcap C1$;
        end
    end
end
return $Worker$;
```

**Algorithm 1:** Ring Structure Based Click Farm Detection Algorithm.

The algorithm is also described below:

1. From the blockchain transactions, we obtain transactional relationship $r = \langle from, to, money, time \rangle$, where $from$, $to$ represent the payment account and the receiving account respectively, $money$ is the payment amount, and $time$ is the payment time. We also let $R$ denotes the transaction set, and assume that the number of transactions in the set of $R$ is $n$, that is, $|R| = n$.
2. For the given $mi \in V_{Mc}$, we select the transaction set $R1$ from $R$ such that $R1.to = mi$, and the transaction set $R2$ from $R$ such that $R2.to = R1.from$. We also remark that conditions $R2.time > R1.time$ and $R2.money > R1.money$ must be satisfied. Then, for every transaction in $R1$, we have $R1.from$ in $Customer[i]$.
3. As shown in step 2, we iteratively go over $k$ $mi \in V_{Mc}$ to obtain $k$ sets of $Customer[i]$.
4. We calculate the intersection $Worker$ of $k$ sets $Customer$, $Worker = Customer[1] \bigcap Customer[2] ... \bigcap Customer[k]$. Thus, the set $Worker$ is a set of suspected workers accounts.

### 4.3. Time threshold based detection algorithm

The proposed time threshold based detection algorithm is described below, and in Algorithm 2:

1. In blockchain, we attempt to find the time period $[T1, T2]$ when the sales volume and ratings of the store increase rapidly. This time period can also be used to select the corresponding transactions and from these transactions we can obtain the respective transaction relationship: $r = \langle from, to, money, time \rangle$, where $from$, $to$ represent the payment account and the receiving account respectively, $money$ is the payment amount, and $time$ is the payment time. Again, let $R$ denote the transaction set. We assume that the number of transactions in the set of payment relations is $n$, that is, $|R| = n$.
2. For the given $mi \in V_{Mc}$, we select the transaction set $R1$ from $R$ that $R1.to = mi$, which satisfies conditions $R1.time > T1$ and $R1.time < T2$. Then, for every transaction in $R1$, we make $R1.from$ in $Customer[.][.]$.
3. As shown in steps 1 and 2, we traverse $k$ $mi \in V_{Mc}$ to get $k$ sets of $Customer[i][]$.
4. We then calculate the intersection $Worker$ of $k$ sets $Customer[i]$, $Worker = Customer[1] \bigcap Customer[2] ... \bigcap Customer[k]$. Thus, the set $Worker$ is a set of suspicious workers accounts.

```
Data: $Merchant[.], Worker[.], T1, T2, Customer[.][.], i = 0, j = 0, Transaction[.]$
Result: Worker set $C$
for each $m \in Merchant$ do
    for each $R1 \in Transaction$ do
        if $T1 < R1.time < T2$ then
            $Customer[i][j] = R1.from$;
            $j = j + 1$;
        end
    end
    $i = i + 1$;
end
for each $C \in Customer$ do
    if $C \neq null$ then
        $Worker = C$;
        $break$;
    end
end
if $Worker \neq null$ then
    for each $C1 \in Customer$ do
        if $C1 \neq null$ then
            $Worker = Worker \bigcap C1$;
        end
    end
end
return $Worker$;
```

**Algorithm 2:** Time Threshold Based Detection Algorithm.

### 4.4. Probability threshold based detection algorithm

In Algorithm 3, a probability threshold is added to improve the accuracy of Algorithm 1. The number of click farm workers will account for a significant proportion of an increase in the product's sales, rankings or popularity. Here, we set a probability threshold, when the proportion of suspicious accounts to the total number of customers is greater than this threshold.

The algorithm is described below:

1. Let the payment relationship be denoted as $r = \langle from, to, money, time\rangle$, where $from$, $to$ represent the payment account and the receiving account respectively, $money$ is the payment amount, and $time$ is the payment time. Let $R$ be the transaction set, and the number of transactions in the set of payment relations to be $n$ (i.e., $|R| = n$).
2. For the given $mi \in V_{Mc}$, we select the transaction set $R1$ from $R$ such that $R1.to = mi$ and make $R1.from$ in $All\_customer[i]$. We also select all transactions $R2$ from $R$ such that $R1.to = mi$ and $R2.to = R1.from$, which satisfy conditions $R2.time > R1.time$ and $R2.money > R1.money$. Then, for every transaction in $R1$, we make $R1.from$ in $Customer[i]$.
3. As is shown in step 2, we iterate over $k$ $mi \in V_{Mc}$ to get $k$ sets of $Customer[i]$.
4. We calculate the intersection $Worker$ of $k$ sets $Customer$, $Worker = Customer[1] \bigcap Customer[2] ... \bigcap Customer[k]$.
5. We set a probability threshold $Pr$, for each $All\_customer[i]$, if $Worker/All\_customer[i] >= Pr$, then the set $Worker$ is a set of suspected workers accounts.

```
Data: $Merchant[.], Worker[.], All\_customer = [.][.], Customer[.][.], i = 0, j = 0, k = 0, Transaction[.], Pr$
Result: Worker set $C$
for each $m \in Merchant$ do
    for each $R1 \in Transaction$ do
        if $R1.to == m$ then
            $All\_customer[i][k] = R1.from$;
            $k = k + 1$;
        end
        for each $R2 \in Transaction$ do
            if $R1 == R2$ then
                continue;
            end
            if $R1.to == m \bigcap R2.to == R1.from \bigcap R1.money < R2.money \bigcap R1.time < R2.time$ then
                $Customer[i][j] = R1.from$;
                $j = j + 1$;
            end
        end
    end
    $i = i + 1$;
end
for each $C \in Customer$ do
    if $C \neq null$ then
        $Worker = C$; break;
    end
end
if $Worker \neq null$ then
    for each $C1 \in Customer$ do
        if $C1 \neq null$ then
            $Worker = Worker \bigcap C1$;
        end
    end
    for each $C2 \in All\_customer$ do
        if $C2 \neq null \bigcap Worker/C2 < Pr$ then
            $Worker$=null; break;
        end
    end
end
return $Worker$;
```

**Algorithm 3:** Probability Threshold Based Detection Algorithm.

## 5. Performance analysis

### 5.1. Performance analysis

#### 5.1.1. Algorithm 1

The time complexity analysis of Algorithm 1 is divided into three cases, namely: best case, worst case, and random.

In the best case scenario, no eligible transaction is found in the blockchain ledger. Therefore, the number of loops in the inner loop of the algorithm is 0, and the number of assignments for data element in each loop is 1. In other words, the number of executions of the assignment statement in the outer loop is $1 * |Merchant|$, in the two single-layer loop, and the number of executions of the assignment statement in the entire process is 0. Thus, the best case complexity is $O(|Merchant|)$.

**Proposition 1.** *The least time cost of Algorithm 1 is $T(m, n) = T(m)$, where $m = |Merchant|$, $n = |Transaction|$.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m, n) = m * (1 + 0 * n * n) + 0 = T(m)$$

The worst case occurs in the three-layer loop, where each transaction satisfies the second if-statement. Therefore, the innermost loop has 1 comparison each time, and the number of executions of assignment statement in each inner loop is 2. Thus, the total number of three-layer loop comparisons is $|Transaction| * |Transaction| * |Merchant|$, and the number of assignment statement executions is $|Transaction| * |Transaction| * |Merchant| * 2 + |merchant| * 1$. In two one-layer loops, the numbers of executions of assignment statement in the former loop and the latter loop are 1 and $|Merchant|$, respectively. Thus, the worst-case time complexity of the algorithm is $O(|Transaction|^2 * |Merchant|)$.

**Proposition 2.** *The worst-case time complexity of the Algorithm 1 is $T(m, n) = T(mn^2)$, where $m = |Merchant|$, $n = |Transaction|$.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m, n) = m * (n * n * 2 + 1) + 1 + 1 * m = T(mn^2)$$

In random cases, because the transaction meeting the conditions are random in the two-layer loop, the expected numbers of comparisons and assignment statement executions are $|Merchant| * |Transaction| * |Transaction|/2$ and $|Merchant| * |Transaction| * |Transaction|$, respectively. In the two one-layer loop, the numbers of executions of assignment statement in the former loop and the latter loop are $|Merchant|/2$ and $|Merchant|/2$, respectively. Therefore, the time complexity in the random case is $O(|Merchants| * |Transaction|^2)$.

**Proposition 3.** *The time complexity in the random case of Algorithm 1 is $T(m, n) = T(mn^2)$, where $m = |Merchant|$, $n = |Transaction|$.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m, n) = m * (n * n/2 * 2 + 1) + 2 * m/2 = T(mn^2)$$

#### 5.1.2. Algorithm 2

The time complexity analysis of Algorithm 2 is also divided into best case, worst case, and random.

Since in the best case no eligible transaction is found in the blockchain, the entire algorithm consists of a two-layer loop and a single-layer loop. In a double-layer loop, the number of executions of the assignment statement is 1 in each loop, so the number of loops for the entire double-layer loop is $1 * |Merchant|$. In the two single-layer loops, the number of executions of assignment statement in the former loop and the latter loop is 0 and 0. Thus, the best case complexity is $O(|Merchant|)$.

**Table 2**
Performance of Algorithms 1 to 3: A comparative summary.

| Algorithm | Minimal T(n) | Maximum T(n) | Average T(n) |
|---|---|---|---|
| Ring structure based detection algorithm | $O(m)$ | $O(mn^2)$ | $O(mn^2)$ |
| Time threshold based detection algorithm | $O(m)$ | $O(mn)$ | $O(mn)$ |
| Probability threshold based detection algorithm | $O(m)$ | $O(mn^2)$ | $O(mn^2)$ |

**Proposition 4.** *The best case complexity of Algorithm 2 is* $T(m,n) = T(m)$*, where* $m = |Merchant|$*,* $n = |Transaction|$*.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m,n) = m * (1 + 0 * n) + 0 = T(m)$$

The worst case is that in the two-layer loop, each transaction satisfies the if-statement, so that the innermost loop has 1 comparisons each time, the number of executions of assignment statement in each innermost loop is 2. In the outer loop, the number of executions of the assignment statement is 1. Thus, the number of two-layer loop comparisons is $|Transaction| * |Merchant|$, and the number of assignment statement executions is $|Transaction| * |Merchant| * 2 + |merchant| * 1$. In the two single-layer loop, the number of executions of assignment statement in the former loop and the latter loop is 1 and $|Merchant|$. Thus, the worst-case time complexity of the algorithm is $O(|Transaction| * |Merchant|)$.

**Proposition 5.** *The worst case complexity of Algorithm 2 is* $T(m,n) = T(mn)$*, where* $m = |Merchant|$*,* $n = |Transaction|$*.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m,n) = m * (1 + 1 * n) + 1 + m = T(mn)$$

In random cases, because the transaction meets the conditions are random in the two-layer loop, the expected numbers of comparisons and assignment statement executions are $|Merchant| * |Transaction|/2$ and $|Merchant| * |Transaction|$, respectively. In the two single-layer loop, the numbers of executions of assignment statement in the former loop and the latter loop are $|Merchant/2|$ and $|Merchant|/2$, respectively. Therefore, the time complexity in the random case is $O(|Merchants| * |Transaction|)$.

**Proposition 6.** *The random case complexity of Algorithm 2 is* $T(m,n) = T(mn)$*, where* $m = |Merchant|$*,* $n = |Transaction|$*.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m,n) = m * (1 + n/2 * 2) + 2 * m/2 = T(mn)$$

### *5.1.3. Algorithm 3*

The time complexity of Algorithm 3 is divided into best, worst, and random cases. Algorithm 3 differs from Algorithm 1 in that a single layer loop is added at the end, and the remaining loops are unchanged. Here, we only discuss the impact of the new single-layer loop on time complexity.

The best case is that no eligible transaction is found in the blockchain ledger. Therefore, in the bottom loop, the number of executions of the assignment statement is 0. Thus, the best case complexity is $O(|Merchant|)$.

**Proposition 7.** *The least time cost of Algorithm 3 is* $T(m,n) = T(m)$*, where* $m = |Merchant|$*,* $n = |Transaction|$*.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m,n) = m * (1 + 0 * n * n) + 3 * 0 = T(m)$$

The worst case is that in the tree-layer loop, each transaction satisfies the if-statement. Therefore, in the bottom loop, the number of executions of the assignment statement is $|Merchant|$. Thus, the worst-case time complexity is $O(|Transaction|^2 * |Merchant|)$.

**Table 3**
Feasibility analysis.

| | Experiment 1 | Experiment 2 | Experiment 3 |
|---|---|---|---|
| TP | 16 | 19 | 16 |
| FP | 0 | 0 | 0 |
| FN | 4 | 1 | 4 |
| TN | 69 | 69 | 69 |
| Accuracy | 0.955 | 0.989 | 0.955 |

**Proposition 8.** *The worst-case time complexity of Algorithm 3 is* $T(m,n) = T(mn^2)$*, where* $m = |Merchant|$*,* $n = |Transaction|$*.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m,n) = m * (n * n * 2 + 1) + 1 + 2 * m = T(mn^2)$$

In the random case, due to the fact that the transaction meets the conditions randomly, in the bottom loop, the number of executions of the assignment statement is $|Merchant|/2$. Thus, the random-case time complexity is $O(|Transaction|^2 * |Merchant|)$.

**Proposition 9.** *The time complexity in the random case of Algorithm 3 is* $T(m,n) = T(mn^2)$*, where* $m = |Merchant|$*,* $n = |Transaction|$*.*

$$m = |Merchant|, n = |Transaction|,$$

$$T(m,n) = m * (n * n/2 * 2 + 1) + 3 * m/2 = T(mn^2)$$

### *5.1.4. Summary*

A comparative summary of the three algorithms' performance is presented in Table 2.

### *5.1.5. Feasibility analysis*

According to the above 3 algorithms, we have carried on the simulation experiment, tested 220 transactions randomly and successfully found the click farming operations. Specifically, we set 89 users, including click farming workers, normal users and merchants. In addition, we conclude TP (True Positive), FP (False Positive), FN (False Negative), TN (True Negative) and Accuracy = (TP + TN)/(TP + TN + FP +FN), which are presented in Table 3.

## *5.2. Soundness and completeness*

Li et al. [35] explained that workers working on the same campaign, particularly from the same click farm, tend to focus on the same product and using similar comments. Thus, they used the similarities between the reviews to build a social graph, and applied the Louvain community detection method to the social graph. In other words, the more similar the comments are between the users, the more similar the users are likely to be. Their proposed algorithm introduces a time period $T$ and a similarity threshold S. For a particular merchant in time period T, the similarity between users is compared using the reviews. When the similarity of users exceeds the similarity threshold S, a social link is built between these users.

However, merchants involved in click farm activities often sell different types of products (e.g., household cleaning products, and electronic appliances). The Louvain community monitoring approach only focuses on workers with similar reviews on a particular product or product category, which is not reflective of real-world or more complex click farm operations. Therefore, we choose to not focus on

**Table 4**
Scheme comparison.

| | Our scheme | Neng Li [41] | Qingpeng Cai [42] |
|---|---|---|---|
| Methods | Graph theory<br>Timethreshold<br>Probability threshold<br>Cryptocurrency<br>Ring structure | Louvain community detection<br>Clustering communities<br>Binary classification | Tailored model<br>Incentive mechanism |
| Differential models for entry number | ✓ | × | ✓ |
| Detect over anonymity | ✓ | × | × |
| Cryptocurrency support | ✓ | × | × |

review similarities, and instead focused on the analysis of the ring structure formed by the click farm workers, click farm delegations and merchants. We demonstrated that through the interaction between these three entities, our 3 algorithms allow us to flag workers suspected of been involved in click farm activities from other normal customers. In addition, we also compare our scheme with the other two schemes from four perspectives: methods, differential models for entry number, detect over anonymity and cryptocurrency support. After comparison, we can conclude that our scheme meet all above conditions, while other schemes can only meet parts of them. Therefore, our scheme has good applicability and feasibility. The specific results are shown in the Table 4.

## 6. Conclusion and future work

In this paper, we focused on click farm operations and presented three models to capture real-world interactions in such ecosystem. Then, we designed three algorithms to facilitate the detection of click farm activities. Specifically, we proposed a ring structure based detection algorithm on the basis of the circular structure formed by the flow of information between merchants, delegations (click farm operators) and workers. This algorithm also relies on two other algorithms we designed in this paper, namely: a time threshold based detection algorithm (that uses a time threshold to filter transactions occurring within a particular time threshold) and a probability threshold based detection algorithm (to identify suspicious transactions exceeding a certain probability threshold).

Future research includes implementing and evaluating our proposed approach in collaboration with a real-world e-commerce platform.

## CRediT authorship contribution statement

**Shiyong Huang:** Conceptualization, Methodology, Software, Investigation, Formal analysis, Writing – original draft. **Xin Yang:** Data curation, Writing – original draft. **Langyue He:** Visualization, Investigation. **Xiaohan Hao:** Resources, Supervision, Writing – review & editing. **Wei Ren:** Conceptualization, Funding acquisition, Resources, Supervision, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability statement

The [code] data used to support the findings of this study have been deposited in the [IEEE DATAPORT] repository ([10.21227/d509-qf61]). The [code] data used to support the findings of this study have been deposited in the [IEEE DATAPORT] repository ([10.21227/cr9q-1a58]).

## Acknowledgments

The research is financially supported by the Foundation of Yunnan Key Laboratory of Blockchain Application Technology (No. 202105A G070005, No. YNB202103), the Provincial Key Research and Development Program of Hubei (No. 2020BAB105), National Natural Science Foundation of China (No. 61972366), and the Foundation of Hubei Key Laboratory of Intelligent Geo-Information Processing (No. KLIGIP-2021B06).

## References

[1] S. Zollner, K.R. Choo, N. Le-Khac, An automated live forensic and post-mortem analysis tool for bitcoin on windows systems, IEEE Access 7 (2019) 158250–158263.

[2] T. Volety, S. Saini, T. McGhin, C.Z. Liu, K.R. Choo, Cracking bitcoin wallets: I want what you have in the wallets, Future Gener. Comput. Syst. 91 (2019) 136–143.

[3] W. Ren, R. Liu, M. Lei, K.K.R. Choo, SeGoAC: A tree-based model for self-defined, proxy-enabled and group-oriented access control in mobile cloud computing, Comput. Stand. Interfaces 54 (2017) 29–35, http://dx.doi.org/10.1016/j.csi.2016.09.001.

[4] H. Rezaeighaleh, C.C. Zou, Deterministic sub-wallet for cryptocurrencies, in: 2019 IEEE International Conference on Blockchain, Blockchain, 2019, pp. 419–424.

[5] S. He, Q. Wu, X. Luo, Z. Liang, D. Li, H. Feng, H. Zheng, Y. Li, A social network based cryptocurrency wallet management scheme, IEEE Access 6 (99) (2018) 7654–7663.

[6] L. Herskind, P. Katsikouli, N. Dragoni, Privacy and cryptocurrencies - A systematic literature review, IEEE Access 8 (2020) 54044–54059.

[7] L.C. Schaupp, M. Festa, Cryptocurrency adoption and the road to regulation, in: Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age, ACM, New York, NY, USA, 2018, pp. 1–9.

[8] I. Makarov, A. Schoar, Trading and arbitrage in cryptocurrency markets, J. Financ. Econ. 135 (2) (2020) 293–319, http://dx.doi.org/10.1016/j.jfineco.2019.07.001.

[9] J. Liu, Z. Zhao, X. Cui, Z. Wang, Q. Liu, A novel approach for detecting browser-based silent miner, in: 2018 IEEE Third International Conference on Data Science in Cyberspace, DSC, 2018, pp. 490–497.

[10] Y. Li, W. Susilo, G. Yang, Y. Yu, X. Du, D. Liu, N. Guizani, Toward privacy and regulation in blockchain-based cryptocurrencies, IEEE Netw. 33 (5) (2019) 111–117.

[11] L. Zhang, H. Li, Y. Li, Y. Yu, M.H. Au, B. Wang, An efficient linkable group signature for payer tracing in anonymous cryptocurrencies, Future Gener. Comput. Syst. 101 (2019) 29–38.

[12] Y. Li, G. Yang, W. Susilo, Y. Yu, M.H. Au, D. Liu, Traceable monero: Anonymous cryptocurrency with enhanced accountability, IEEE Trans. Dependable Secure Comput. 18 (2) (2021) 679–691.

[13] Y. Li, W. Susilo, G. Yang, Y. Yu, D. Liu, X. Du, M. Guizani, A blockchain-based self-tallying voting protocol in decentralized IoT, IEEE Trans. Dependable Secure Comput. 19 (1) (2022) 119–130, http://dx.doi.org/10.1109/TDSC.2020.2979856.

[14] D. Li, D. Han, Z. Zheng, T.H. Weng, H. Li, H. Liu, A. Castiglione, K.C. Li, MOOCsChain: A blockchain-based secure storage and sharing scheme for MOOCs learning, Comput. Stand. Interfaces 81 (2022) 103597, http://dx.doi.org/10.1016/j.csi.2021.103597, https://www.sciencedirect.com/science/article/pii/S0920548921000921.

[15] D. Liu, Y. Zhang, D. Jia, Q. Zhang, X. Zhao, H. Rong, Toward secure distributed data storage with error locating in blockchain enabled edge computing, Comput. Stand. Interfaces 79 (2022) 103560.

[16] I. Petrov, L. Invernizzi, E. Bursztein, CoinPolice: Detecting hidden cryptojacking attacks with neural networks, 2020, arXiv preprint arXiv:2006.10861.

[17] A. Kharraz, Z. Ma, P. Murley, C. Lever, J. Mason, A. Miller, N. Borisov, M. Antonakakis, M. Bailey, Outguard: Detecting in-browser covert cryptocurrency mining in the wild, in: The World Wide Web Conference, 2019.

[18] G. Hong, Z. Yang, S. Yang, L. Zhang, Y. Nan, Z. Zhang, M. Yang, Y. Zhang, Z. Qian, H.X. Duan, How you get shot in the back: A systematical study about cryptojacking in the real world, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018.

[19] A. Yulianto, P. Sukarno, A.A. Wardana, M.A. Makky, Mitigation of cryptojacking attacks using taint analysis, in: 2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering, ICITISEE, 2019, pp. 234–238.

[20] R.K. Konoth, E. Vineti, V. Moonsamy, M. Lindorfer, C. Krügel, H. Bos, G. Vigna, MineSweeper: An in-depth look into drive-by cryptocurrency mining and its defense, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018.

[21] W. Wang, B. Ferrell, X. Xu, K.W. Hamlen, S. Hao, SEISMIC: Secure in-lined script monitors for interrupting cryptojacks, in: ESORICS, 2018.

[22] S. Eskandari, A. Leoutsarakos, T. Mursch, J. Clark, A first look at browser-based cryptojacking, in: 2018 IEEE European Symposium on Security and Privacy Workshops, EuroS and PW, 2018, pp. 58–66.

[23] S. Xie, G. Wang, S. Lin, P.S. Yu, Review spam detection via temporal pattern discovery, in: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012, pp. 823–831.

[24] Z. Yang, C. Wilson, X. Wang, T. Gao, B.Y. Zhao, Y. Dai, Uncovering social network sybils in the wild, ACM Trans. Knowl. Discov. Data 8 (1) (2014) 2:1–2:29.

[25] Y. Boshmaf, K. Beznosov, M. Ripeanu, Graph-based sybil detection in social and information systems, in: 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2013, in: ASONAM '13, ACM, New York, NY, USA, 2013, pp. 466–473.

[26] N.Z. Gong, M. Frank, P. Mittal, SybilBelief: A semi-supervised learning approach for structure-based sybil detection, IEEE Trans. Inf. Forensics Secur. 9 (6) (2017) 976–987.

[27] M. Singh, D. Bansal, S. Sofat, Who is who on twitter–spammer, fake or compromised account? a tool to reveal true identity in real-time, Cybern. Syst. 49 (1) (2018) 1–25.

[28] Y. Boshmaf, D. Logothetis, G. Siganos, J. Leria, J. Lorenzo, M. Ripeanu, K. Beznosov, H. Halawa, Integro: Leveraging victim prediction for robust fake account detection in large scale OSNs, Comput. Secur. 61 (2016) 142–168.

[29] Q. Cao, X. Yang, J. Yu, C. Palow, Uncovering large groups of active malicious accounts in online social networks, in: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, in: CCS '14, ACM, New York, NY, USA, 2014, pp. 477–488.

[30] N. Li, S. Du, H. Zheng, M. Xue, H. Zhu, Fake reviews tell no tales? dissecting click farming in content-generated social networks, China Commun. 15 (4) (2018) 98–109.

[31] C. Jiang, J. Zhu, Q. Xu, et al., Dissecting click farming on the Taobao platform in China via PU learning and weighted logistic regression, Electron. Commer. Res. (2020) 1–20.

[32] M. Dai, S. Zhang, H. Wang, S. Jin, A low storage room requirement framework for distributed ledger in blockchain, IEEE Access 6 (2018) 22970–22975.

[33] S.V. Lokam, S. Ruj, K. Sakurai (Eds.), Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts, BCC@AsiaCCS 2018, Incheon, Republic of Korea, June 4, 2018, ACM, 2018.

[34] K. Zhang, H. Jacobsen, Towards dependable, scalable, and pervasive distributed ledgers with blockchains, in: 38th IEEE International Conference on Distributed Computing Systems, ICDCS 2018, Vienna, Austria, July 2–6, 2018, 2018, pp. 1337–1346.

[35] H. Zheng, N. Li, M. Xue, S. Du, H. Zhu, Fake reviews tell no tales? Dissecting click farming in content-generated social networks, in: 2017 IEEE/CIC International Conference on Communications in China, ICCC, 2017, pp. 1–6, http://dx.doi.org/10.1109/ICCChina.2017.8330469.

[36] R. Chen, Y. Li, Y. Yu, H. Li, X. Chen, W. Susilo, Blockchain-based dynamic provable data possession for smart cities, IEEE Internet Things J. 7 (5) (2020) 4143–4154, http://dx.doi.org/10.1109/JIOT.2019.2963789.

[37] G. Tian, Y. Hu, J. Wei, Z. Liu, X. Huang, X. Chen, W. Susilo, Blockchain-based secure deduplication and shared auditing in decentralized storage, IEEE Trans. Dependable Secure Comput. (2021) 1, http://dx.doi.org/10.1109/TDSC.2021.3114160.

[38] X. Ma, C. Wang, X. Chen, Trusted data sharing with flexible access control based on blockchain, Comput. Stand. Interfaces 78 (2021) 103543.

[39] B. Bera, A. Vangala, A.K. Das, P. Lorenz, M.K. Khan, Private blockchain-envisioned drones-assisted authentication scheme in IoT-enabled agricultural environment, Comput. Stand. Interfaces 80 (2022) 103567.

[40] X. Zheng, Y. Zhao, H. Li, R. Chen, D. Zheng, Blockchain-based verifiable privacy-preserving data classification protocol for medical data, Comput. Stand. Interfaces 82 (2022) 103605.

[41] N. Li, S. Du, H. Zheng, M. Xue, H. Zhu, Fake reviews tell no tales? dissecting click farming in content-generated social networks, China Commun. 15 (4) (2018) 98–109.

[42] Q. Cai, A. Filos-Ratsikas, C. Liu, P. Tang, Mechanism design for personalized recommender systems, in: Proceedings of the 10th ACM Conference on Recommender Systems, 2016, pp. 159–166.