IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 9, 2020, accepted November 17, 2020, date of publication November 20, 2020,
date of current version December 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3039570

Heuristic-Based Address Clustering in Bitcoin

YUHANG ZHANG[1], JUN WANG[1], AND JIE LUO[2], (Member, IEEE)
[1]School of Economics and Management, Beihang University, Beijing 100191, China
[2]State Key Laboratory of Software Development Environment, School of Computer Science and Engineering, Beihang University, Beijing 100191, China

Corresponding authors: Jun Wang (king.wang@buaa.edu.cn) and Jie Luo (luojie@nlsde.buaa.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 71871005 and Grant 71531001.

ABSTRACT With the emergence of decentralized cryptocurrencies such as Bitcoin, it has become very difficult for law enforcement to detect suspicious activities, identify users and obtain transaction records for criminals who utilize the pseudoanonymity provided by the cryptocurrency system. Address clustering aims to break such pseudoanonymity by linking addresses that are controlled by the same user based on the information available from the blockchain, such as transaction graphs. There are already two widely used heuristics for Bitcoin address clustering. One is based on the multiple input addresses of transactions. The other is based on one-time change addresses. By reconsidering the one-time change address-based heuristic from the perspective of address reuse, we propose a new heuristic that detects one-time change addresses by eliminating addresses that are reused later as non-change addresses. As a result, this heuristic works for transactions whose one-time change addresses cannot be identified by the previous two heuristics. The experimental results for different scales of Bitcoin transaction data show that the proposed heuristic has a 0.33% mean contribution to the ratio of address reduction in addition to the contribution of the multiple input addresses and one-time change address heuristics.

INDEX TERMS Heuristic, address clustering, blockchain, Bitcoin.

I. INTRODUCTION

Financial crimes are serious problems that all governments need to face, which are exemplified by fraud and money laundering.[1] They can cause nonnegligible harm to national political, economic and social stability and pose a great threat to the financial property security of individuals and countries. Criminals continuously seek new methods of money transfer to avoid government supervision. The electronic currency system Bitcoin [18], a distributed and decentralized cryptocurrency based on blockchain, attracts many of them because of its pseudoanonymous nature; i.e., users can transfer money using pseudonyms represented by public key addresses without explicit ties to real-world identities. As a result, financial crimes involving cryptocurrencies such as Bitcoin have grown explosively. Effectively linking the addresses controlled by a user and potentially obtaining clues that can eventually lead to the real-world identity of the user based on the information in the blockchain has become an important issue.

Bitcoin encourages practices such as using new addresses for each transaction to protect the privacy of users, which increases the difficulty of linking addresses controlled by the same user. However, it is still possible to link addresses controlled by the same user based on publicly available transaction information or mistakes made by users, such as address reuse. In previous work [1], [16], [20], researchers proposed several heuristics for address clustering based on transactions with multiple input addresses that have to be controlled by one user and one-time change addresses that have to be owned by the same user as the input addresses. It was also shown that it is possible to establish a link between one of a user's addresses and information from additional sources that reveals the user's identity. These heuristics have also been used to implement blockchain analysis software and services such as GraphSense [12], BitIodine [22], BitcoinWhosWho[2] and BTCTester.[3]

In this article, we aim to design a new heuristic for detecting one-time change addresses from a different perspective: address reuse. The one-time change address-based heuristic requires that all the output addresses except one appear in

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

[1]https://www.fbi.gov/stats-services/publications/financial-crimes-report-2010-2011

[2]http://bitcoinwhoswho.com

[3]http://btctester.com

previous transactions to determine the change address, which is one kind of address reuse. There is another kind of address reuse, in which the output addresses of one transaction are reused in a later transaction, which can also be used for identifying one-time change addresses. Based on this idea, we propose the address reuse-based change address detection heuristic. The experimental results for different scales of Bitcoin transactions show that the proposed heuristic has a positive contribution to the performance of the address clustering algorithm and can maintain a stable improvement in performance as the scale of transactions becomes large.

This article is organized in seven sections. The related work is discussed in section II. In section III, we introduce the basis of Bitcoin blockchain and define the concepts that are used in this article. In section IV, we present the heuristic-based address clustering algorithm for Bitcoin blockchain. Three heuristics are defined, including two widely used heuristics and a new address reuse-based heuristic. In section V, we conduct a series of experiments to evaluate the performance of the address clustering algorithm on different data scales and the contributions of different heuristics to the performance. We discuss the results and the limitations of the proposed heuristics in section VI. Finally, we conclude this article and present several issues for future work in section VII.

II. RELATED WORK

As a key process for deanonymizing the real-world identity of Bitcoin users, address clustering has always been a main research topic because of its ability to link together multiple addresses that are potentially controlled by the same user. There are two types of address clustering approaches.

One is heuristic-based address clustering, which attempts to utilize the structural information of certain transactions to reason about the relations among addresses. Reid *et al.* present a heuristic clustering algorithm to merge public key addresses that belong to the same users in Bitcoin transactions to form a user transaction network [1], [20]. Meiklejohn *et al.* give a stricter definition of multiple input addresses and one-time change address-based heuristics for address clustering and examine the gap between actual and potential anonymity in Bitcoin [16]. Spagnuolo *et al.* design and implement a modular framework named BitIodine, which parses the blockchain, clusters addresses that are likely to belong to the same user or group of users, classifies such users and labels them [22]. Fleder *et al.* propose a system for analyzing a transaction network of digital currency, which connects transactions with real users through Bitcoin public key address information obtained from websites and digital currency forums to achieve deanonymization of Bitcoin [7]. Zheng *et al.* implement a heuristic-based system for clustering incidence relations among Bitcoin addresses and design an improved Louvain algorithm for finding user transaction communities to track malicious Bitcoin transactions and reveal the privacy vulnerabilities associated with blockchain technology [25], [26]. Harrigan and Fretter investigate the primary reasons behind why simple heuristics based on the microstructure of transactions have proved very effective in practice—address reuse, avoidable merging, superclusters with high centrality, and the incremental growth of address clusters—and quantify their impact during Bitcoin's first seven years of existence [11]. For alternative blockchains other than Bitcoin, Victor proposes three heuristics specifically for Ethereum—deposit address reuse, airdrop multiparticipation, and self-authorization—and exchange deposit address reuse is shown to be the most effective heuristic [24].

The second type of address clustering approach utilizes information other than the structure of transactions in various ways. Some studies [6], [9], [21] attempt to integrate external information such as web pages, forum information, and IP addresses into transaction data by using data flow analysis and machine learning to recognize individual user behavior to obtain real user identities in the transaction network. Moser proposes a way to analyze the anonymity of Bitcoin transactions, especially with different mixing approaches and services [17]. Biryukov *et al.* propose a method to deanonymize Bitcoin users, which allows user pseudonyms to be linked to the IP addresses where the transactions are generated. This method also works for scenarios in which users are behind the NATs or firewalls of their ISPs [2]. Neudecker *et al.* analyze whether clusters created using known heuristics can be correlated to the IP addresses associated with transactions based on network observations [19]. Gaihre *et al.* analyze user behavior via Bitcoin transaction graphs to measure whether users are concerned about anonymity and find that the majority of users show weak concerns about anonymity, which makes address clustering much easier [8]. Harlev *et al.* present a novel approach to reducing the anonymity of the Bitcoin blockchain: using supervised machine learning to predict the types of yet-unidentified entities [10].

There are also many applications of address clustering approaches for deanonimizing the real-world identities of transactions and users. Biryukov *et al.* propose a novel technique for transaction clustering based on network traffic analysis and message propagation mechanics in Bitcoin [3], [4]. The timings of transaction messages leak information about their origins, which can be exploited by a well-connected adversarial node. Evaluations in the Bitcoin testnet show, with a high level of accuracy, deanonymizing transactions issued from a desktop wallet (Bitcoin Core) and from a mobile (Mycelium) wallet. Kang *et al.* propose an address clustering-based approach to find reliable mappings between a Bitcoin address and its owner's IP address to deanonymize the Bitcoin [14]. It is shown that mapping address clusters to IP addresses has better performance than mapping Bitcoin addresses to IP addresses separately, which also demonstrates the potential application of address clustering techniques. Tironsakkul *et al.* try to use address clustering in combination with address taint analysis to track the mixed Bitcoins from the deposited Bitcoins [23]. Jawaheri *et al.* also use address clustering heuristics to help deanonymize users of Tor hidden services who rely on Bitcoin as a payment method by

exploiting public information leaked from online social networks, the blockchain, and onion websites [13].

III. BITCOIN BLOCKCHAIN

A Bitcoin *blockchain* is a distributed database of transactions shared by all participants of a system based on the Bitcoin protocol. It stores every transaction ever executed in the system in the form of *blocks*, which are organized in chronological order by including a hash of the previous block in each block. As a result, a chain of blocks from the genesis block (the first block) to the current block is created, i.e., a blockchain. Because the cryptographic algorithm ensures that the alteration of block content also changes its hash, the modification of a block requires the regeneration of every block after the modified block in order to update its hash, which can guarantee that each block is computationally impractical to modify once it has been in the chain for a while. These properties make Bitcoin transactions irreversible.

The information stored in a Bitcoin blockchain is a series of transactions, which are transfers of Bitcoin values that are collected into blocks. Therefore, each block contains multiple transactions. A *transaction* typically consists of one or more inputs and outputs, which transfer all input Bitcoin values to outputs. Each input refers to the output of a previous transaction with authentication information proving that the user can spend the corresponding Bitcoins, except that the generation (first) transaction does not have an input. Each output contains an address and the value of a transfer to this address. To prevent double spending, each output in a transaction is only allowed to be referenced once by another transaction. This transaction output is then treated as spent. The tree from the generation transactions to the current transaction contains the whole flow of Bitcoin transfer for verifying the validity of this transaction. With the transaction information, one can determine how much value belonged to each address at any point in history. For each transaction, the sum of all input values equals the sum of all output values plus the transaction fee, which is an incentive for the miners who collect the transactions into blocks in Bitcoin blockchain.

As described above, the addresses in the outputs of transactions play a central role and are used to specify both the inputs and outputs of transactions. There are three main types of addresses used in Bitcoin: P2PKH, P2SH, and Bech32. Here, we only consider the P2PKH type of addresses for simplicity, i.e., public key addresses. A *public key address* is primarily a hash of a *public key* based on the ECDSA signature scheme, which represents a possible destination for a Bitcoin payment. Each user can create a practically unlimited number of distinct public/private key pairs and can use different public key addresses for different transactions to maintain anonymity by avoiding linking these addresses to their real-world identities. From the perspective of Bitcoin ownership, a user owns the Bitcoin value transferred to an address (hash of a public key) if he or she can prove that he or she possesses the corresponding private key.

Because we are mainly concerned about the transfer of Bitcoin from addresses to addresses that may have different owners in this article, all valid transactions in Bitcoin blockchain can be represented by a simple transaction graph defined as follows:

Definition 1 (Transaction Graph): A transaction graph is a pair (T, E), where T is the set of all transactions and E is the set of edges connecting transactions. For transaction t, $t' \in T$, $(t', t) \in E$ if an input of transaction t refers to an output of transaction t'.

Because each output of a transaction can only be referred to once by another transaction, the transaction graph is a directed acyclic graph, which represents the flow of Bitcoin transfer.

For each transaction, we can represent it as a 4-tuple as follows:

Definition 2 (Transaction): Each *output* of a transaction (TXO) is defined as a pair (A, v), where A is an address and v is the value of the Bitcoin transfer to A. Each input of a transaction is of the same form as the output because it refers to an output of a previous transaction.

A *transaction* is a 4-tuple $t = (\mathcal{I}, \mathcal{O}, c, s)$, where both $\mathcal{I}$ and $\mathcal{O}$ are sets of TXOs that specify the inputs and outputs of transaction t, c is the transaction fee, which satisfies

$$c = \sum_{(A_i^I, v_i^I) \in \mathcal{I}} v_i^I - \sum_{(A_j^O, v_j^O) \in \mathcal{O}} v_j^O,$$

and s is the timestamp at which transaction t is final.

IV. BITCOIN ADDRESS CLUSTERING

Each public key address in the Bitcoin blockchain can be viewed as a nickname of the user. Because each user has many nicknames, the goal of address clustering is to group addresses into clusters such that each cluster contains the addresses under the control of one user.

The address clustering procedure computes a series of disjoint subsets $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ of the set of all addresses $\mathcal{A}$ such that $\bigcup_{i=1}^{n} C_i = \mathcal{A}$, where $C_1, \ldots, C_n$ denotes the resulting clusters. Then, these disjoint subsets are converted into a map between addresses and clusters; i.e., addresses that are controlled by one user are mapped to the same user ID, which represents a specific cluster. Formally speaking, we compute a map $f : \mathcal{A} \to \mathcal{U}$ such that $A \mapsto u_C$ if $A \in C$, where $\mathcal{U}$ is obtained by assigning a unique ID to each cluster in $\mathcal{C}$ and u_C is the ID of cluster C.

The clustering procedure processes all transactions in chronological order. Each transaction t clusters the addresses $\mathcal{A}_t = \{A \mid (A, v) \in \mathcal{I} \cup \mathcal{O}\}$ in this transaction based on the predefined heuristics and information from processed history transactions and generates disjoint subsets for the addresses that belong to one user. If multiple heuristics are used in the address clustering, then these heuristics are applied sequentially to find new clusters of addresses. The resulting disjoint subsets are then merged with the previous clusters in the following way:

(1) If the subset has no intersection with any previous cluster, a new user ID is assigned to the addresses in the subset;
(2) If there are previous clusters that have intersections with this subset, all these clusters are merged with the subset to obtain a larger cluster that belongs to one user, and one of these clusters' user IDs is assigned to the addresses in the new cluster.

A. HEURISTIC 1 (H1): MULTIPLE INPUT ADDRESSES

As described previously, transactions in Bitcoin allow multiple inputs if there is not enough Bitcoin in one address for transferring. To make a valid transaction, the sender needs to provide the signatures signed using the private keys corresponding to the public key addresses in all inputs to prove that he/she owns the Bitcoin in these addresses. Hence, the initiator of the transaction must have control over all these private keys. It is safe to assume that all addresses in the inputs belong to a single user. This heuristic has been used many times in previous work, such as [20] and [16].

Heuristic 1: For any transaction $t = (\mathcal{I}, \mathcal{O}, c, s)$, if the number of inputs in t is more than one ($|\mathcal{I}| > 1$), then all addresses in $\mathcal{I}$ belong to the same user, i.e., for any $(A, v) \in \mathcal{I}$, $A \mapsto u$, where u is the ID for the user-controlled addresses in $\mathcal{I}$. Each new address in the outputs $\mathcal{O}$ belongs to a new cluster.

For instance, the input addresses $A_1, A_2, A_3, \ldots$ of the transaction shown in Figure 1 are mapped to the same user with ID u.

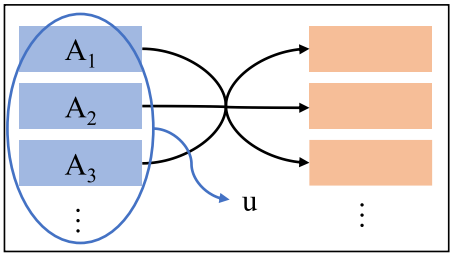

FIGURE 1. Heuristic 1: Clustering based on multiple input addresses.

Although the application of this heuristic does not depend on other transactions, the address clustering procedure enables the fusion of input addresses from different transactions when these sets of input addresses have intersections; i.e., if there is a common address in two different clusters, these two clusters must be controlled by the same user because a common address can only belong to one user. The reason behind the above clustering process is that an address can be used in more than one transaction, i.e., that address reuse occurs. This indicates that address reuse in Bitcoin can leak information about the controller of such addresses.

This heuristic is generally safe, but it can still produce false positives, i.e., sort addresses that are not controlled by the same user into the same cluster. For instance, Bitcoin allows multiple Bitcoin payments from multiple spenders to be combined into a single transaction in a trustless way. CoinJoin [15] is a service that is designed to enhance the privacy of Bitcoin by making the assumption in this heuristic invalid. However, we do not treat the clustering of the input addresses of transactions created by services such as Mt. Gox as false positives because such services have access to the actual owner's private keys, i.e., they have control of these input addresses.

B. HEURISTIC 2 (H2): ONE-TIME CHANGE ADDRESS

In Bitcoin transactions, each output of a transaction can only be referred to once as an input by another transaction. Hence, the Bitcoin in each output must be entirely spent when it is referred to as an input of a transaction. For example, if Tom owns an unspent transaction output (UTXO) with 3 Bitcoins (BTC) and wants to pay Sam 2 BTC, then the transaction created by Tom must contain two outputs: one output contains Sam's address for receiving Tom's payment, and the other output consists of a change address for Tom to use to receive the 1 BTC change. That is, if a transaction contains change, then the change address belongs to the same user that controls the input addresses. Therefore, the change address should be mapped to the same user ID as the inputs. However, the precondition for performing address clustering based on change addresses is to identify the change address a in transaction, which is very difficult for general transactions. There are two types of change addresses in Bitcoin. The first type of change address is specified by the user, and it is commonly one of the input addresses, denoted as a self-change address. The second type of change addresses is generated by the Bitcoin client automatically when change is required, which is typically only used once when the received change is spent again. As a result, this type of change address is called a one-time change address.

Meiklejohn *et al.* [16] propose the following heuristic to identify a one-time change address:

Heuristic 2: An output address O is the change address of a transaction t if the following four conditions are satisfied:
(1) This is the first appearance of address O;
(2) The transaction t is not coin generation;
(3) There is no address among the outputs that also appears in the inputs (self-change address);
(4) The output addresses other than O do not satisfy condition (1).

The change address in the outputs belongs to the same user as the input addresses; i.e., if for an address A_1 in the inputs, $A_1 \mapsto u$, then $O \mapsto u$ as well.

To make the identification more accurate and efficient, Ermilov *et al.* [5] further restrict the transactions to nongeneration transactions that have exactly two outputs, as shown in Figure 2.

The idea behind the design of the above heuristic is to identify one-time change addresses more accurately. There is no absolutely reliable way to detect one-time change addresses, so any heuristic for identifying one-time change addresses may lead to false positive and false negative results. In the design of Heuristic 2, although various restrictions are

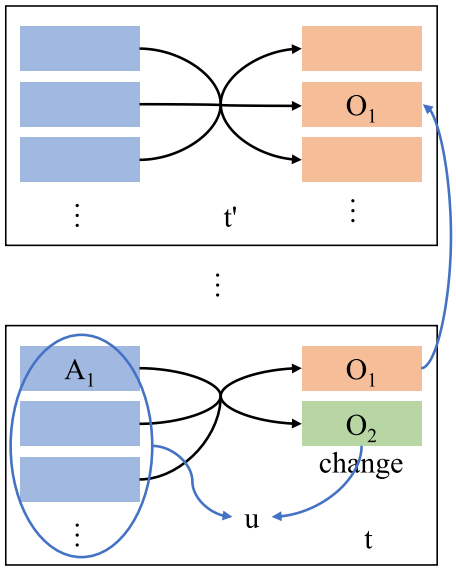

FIGURE 2. Heuristic 2: Clustering based on one-time change addresses.

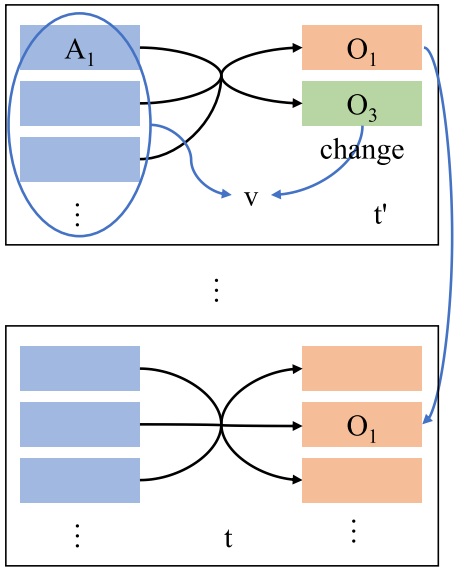

FIGURE 3. Heuristic 3: Clustering based on change address detection by address reuse.

applied to reduce the possibility of generating false positives and false negatives, there are still cases in which it may make mistakes. For instance, if the output addresses of a transaction do not appear in previous transactions, then it is impossible to determine the change address based on this heuristic, and it can lead to false negatives. If the Bitcoin in the inputs is spent without any change, then this heuristic could identify one of the recipient's addresses as a change address by mistake.

C. PROPOSED HEURISTIC

In previous work, the rationality behind the one-time change address-based heuristic was explained from the perspective of the one-time change address. However, we can observe that another key reason behind this heuristic is that address O_1 is reused as an output address in Figure 2. Thus, Heuristic 2 relies on these address reuses to differentiate change addresses from other types of addresses.

Let us consider the example demonstrated in Figure 2 again. By Heuristic 2, address O_1 in t cannot be a one-time change address, and address O_2 belongs to the user u that controls the input addresses (the input addresses are controlled by the same user based on Heuristic 1). If t' is a transaction whose change address cannot be identified with Heuristic 2, then according to the above facts derived from Heuristic 2 and the assumption that a one-time change address is only used once as an input of a transaction, the address O_1 in t' cannot also be a one-time change address. Considering an extreme case in which transaction t' consists of two output addresses O_1 and O_3, with the information that O_1 is not a change address, it can be inferred that the other output address O_3 may be a one-time change address that belongs to the same user v who controls the input addresses of t'.

According to the above discussion, we propose the following new heuristic based on address reuse for one-time change address detection.

Heuristic 3 (Address Reuse-Based Change Address Detection (H3)): An output address O' is the change address of a transaction t' if the following four conditions are satisfied:

(1) This is the first appearance of address O';
(2) The transaction t' is not coin generation;
(3) There is no address among the outputs that also appears in the inputs (self-change address);
(4) The output addresses other than O' are reused as output addresses in some later transactions.

The change address in the outputs then belongs to the same user as the input addresses; i.e., if for an address A_2 in the inputs, $A_2 \mapsto v$, then $O' \mapsto v$ as well.

As shown in Figure 3, the transaction t that reuses address O_1 as output does not require a transaction with exactly two outputs. It can be an arbitrary transaction. This heuristic has a similar problem as Heuristic 2, and it may also lead to false positive and false negative results. For transactions whose output addresses are not reused, it is not possible to determine the change address based on this heuristic, resulting in false negatives. If a change address is reused as an output address or no change occurs in the transaction, then this heuristic may treat the recipient's address as the change address by mistake and lead to false positives.

Note that the way that Heuristics 2 and 3 break the pseudoanonymity of Bitcoin is different from that of Heuristic 1. First, Heuristics 2 and 3 rely on address reuse to take effect and require access to information about other transactions. Second, unlike Heuristic 1, which compromises the privacy of the controller of the reused address, Heuristics 2 and 3 may break privacy for the user who is trading with the controller of the reused address.

For the computational complexity of the proposed heuristic, it is easy to see that Heuristic 3 requires a check for output address reuse similar to that of Heuristic 2. However,

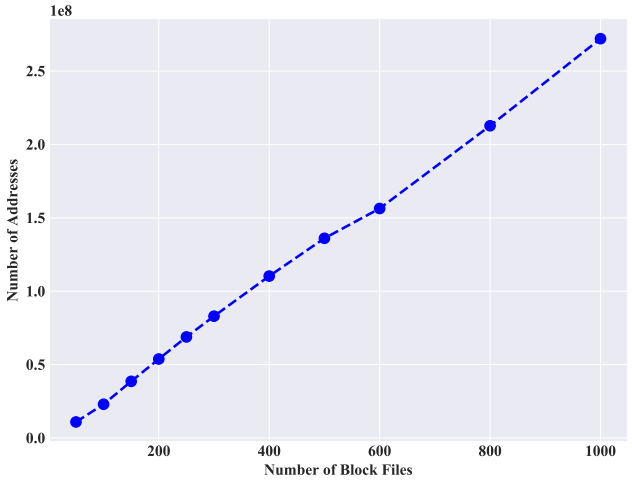

FIGURE 4. Correlation between the number of block files and the number of addresses in them.

Heuristic 3 need to keep tracking of the corresponding input addresses in order to make the detected change address maps to the same user as the input addresses, which requires an additional constant times spaces. That is, the time and space complexity of Heuristic 3 is the same as the Heuristic 2. Let us denote the number of input transactions as n and the maximal number of addresses per transaction is M. Then, the computational complexity of the three heuristics is $O(n)$ because the three algorithms iterate through the list of transactions only once. The space complexities of the three heuristics are $O(M \cdot n) = O(n)$, $O(2M \cdot n) = O(n)$, and $O(3M \cdot n) = O(n)$ respectively, i.e. the space complexity is also $O(n)$ for all three heuristics.

V. EXPERIMENTS AND EVALUATIONS

A. EXPERIMENTAL SETUP

To evaluate the effectiveness of the proposed heuristic-based address clustering algorithms, we implement the algorithm in Rust on top of the BitIodine project,[4] which uses Heuristic 1 for Bitcoin address clustering. We implement Heuristics 2 and 3 to evaluate the effectiveness of different heuristics for clustering. All experiments are performed on a workstation with a Windows 10 64-bit operating system with an Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20 GHz and 96 GB memory.

We use the blockchain data of Bitcoin downloaded with official Bitcoin clients from the creation of the first block on Jan 3, 2009, to Jul 27, 2017. All these transaction data are stored in 1,000 block files that are 128 M in size; i.e., the total size of the blockchain is about 124 G. As the number of block files increases, the number of addresses in these block files increases linearly as well, as shown in Figure 4. Thus, we can use the number of block files instead of the number of addresses in what follows for convenience.

The effectiveness of the address clustering algorithm is evaluated by the ratio of address reduction, defined as follows:

$$r = \frac{|\mathcal{A}| - |\mathcal{C}|}{|\mathcal{A}|}, \tag{1}$$

where $\mathcal{A}$ is the set of all addresses in the transactions, and $\mathcal{C}$ is the set of all clusters after address clustering.

B. EFFECTIVENESS OF DIFFERENT HEURISTICS

To evaluate the effectiveness of the above three heuristics, it is not sufficient to evaluate them only on a fixed small dataset. Because these heuristics can identify more addresses that belong to the same user as the number of transactions increases, it is important to show how the performance of these heuristics changes over different scales of transaction datasets as well. Moreover, address clustering based on different heuristics can overlap. Thus, it is also necessary to evaluate the pure contribution of heuristics on top of other heuristics. A heuristic is redundant if it can be shadowed by other heuristics, i.e., its pure contribution is zero.

Based on the above discussion, we conduct the experiments in the following way: First, the data used for the experiments are the transactions contained in the first k block files of the Bitcoin blockchain, where k ranges over {10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200, 250, 300, 400, 500, 600, 800, 1000}. Second, we evaluate the performance of the address clustering algorithm based on three experimental settings: heuristic 1 (H1), heuristics 1 and 2 (H1 + H2), and heuristics 1, 2 and 3 (H1 + H2 + H3). Because the number of transactions in the Bitcoin blockchain that can be clustered with Heuristic 1 is significantly larger than the numbers of transactions that can be clustered with the other two heuristics, Heuristic 1 is considered the basis for evaluating the pure contribution of the other two heuristics. Since Heuristic 3 is the new heuristic proposed in this article, we also want to evaluate whether it is capable of clustering addresses even after performing the existing Heuristics 1 and 2. In this way, we can determine the pure contribution of Heuristic 2 over Heuristic 1 and that of Heuristic 3 over Heuristics 1 + 2.

The experimental results for address clustering with the three different settings are listed in Table 1. We can see that Heuristic 1 is very effective in clustering addresses that belong to the same user, especially when the transaction data are large enough. It can reduce the number of clusters to less than half of the total addresses in the transactions. In other words, the average number of addresses per cluster is more than 2 when the number of input block files is larger than 100. The performance of address clustering on the three different settings improves as the number of block files increases and reaches a plateau when the number of block files is larger than 250 and less than 500.

A more detailed trend in the performance change is illustrated in Figure 5. The ratio of address reduction for all three settings increases rapidly when the number of block files is less than 50 and continues to increase slowly when the number of block files is in the range of [50, 250]. When

[4]https://github.com/mikispag/bitiodine

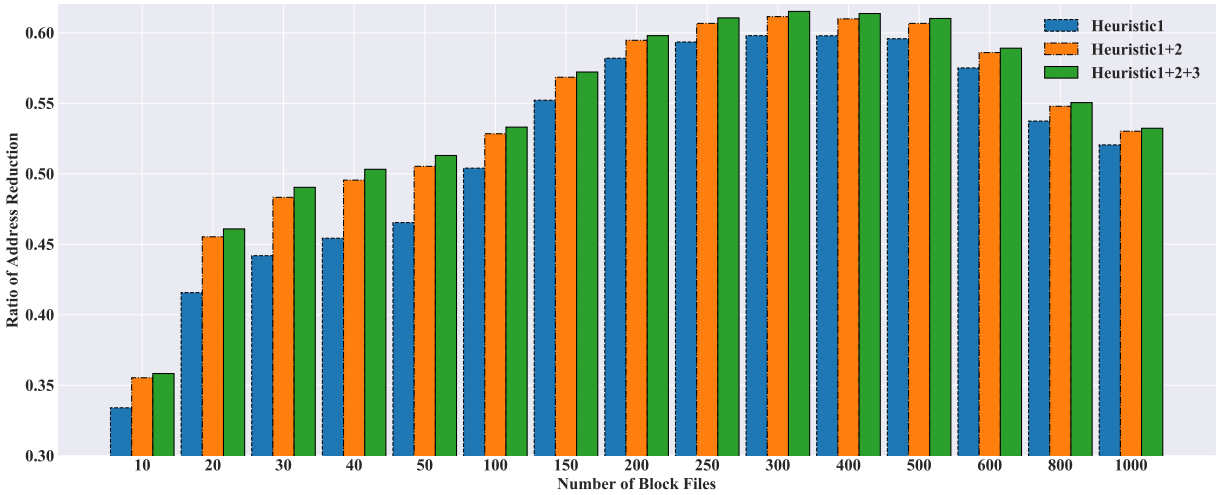

FIGURE 5. Trend of the address reduction performance change over different scales of inputs on different heuristic settings.

TABLE 1. Results of address clustering with different heuristics.

No. of Block Files	No. of Addresses	No. of Clusters			Percent of Reduction		
		H1	H1+H2	H1+H2+H3	H1	H1+H2	H1+H2+H3
50	10,921,819	5,839,291	5,403,175	5,318,602	46.54	50.53	51.30
100	22,993,182	11,403,763	10,843,784	10,735,241	50.40	52.84	53.31
150	38,602,377	17,285,930	16,655,927	16,514,076	55.22	56.85	57.22
200	53,826,469	22,499,744	21,815,857	21,636,091	58.20	59.47	59.80
250	68,849,154	27,989,099	27,077,026	26,808,301	59.35	60.67	61.06
300	83,002,060	33,363,410	32,247,415	31,936,409	59.80	61.15	61.52
400	110,321,148	44,358,131	43,039,244	42,621,950	59.79	60.99	61.37
500	136,054,241	54,993,217	53,510,718	53,039,802	59.58	60.67	61.02
600	156,351,512	66,430,596	64,737,228	64,241,493	57.51	58.60	58.91
800	212,718,722	98,395,891	96,159,791	95,613,257	53.74	54.79	55.05
1000	272,048,692	130,444,839	127,819,267	127,236,187	52.05	53.02	53.23

the number of block files is larger than 250, the ratio of address reduction does not change much and seems to fluctuate around a fixed number. The ratio of address reduction begins to decrease when the number of block files is more than 500. We will discuss the potential reason in the following subsection.

For the pure contribution of Heuristics 2 and 3, the improvement of Heuristics 1 + 2 over Heuristic 1 and Heuristics 1 + 2 + 3 over Heuristics 1 + 2 is shown in Figure 6. It can be seen that the improvement in the performance of address clustering due to Heuristics 2 and 3 increases rapidly as the number of input block files increases and then decreases until a balance is reached. The initial increase in improvement coincides with the rapid increase in the address reduction ratio for Heuristic 1, which indicates that the increase is caused by the discovery of address reuse as more transactions are processed. The decrease in the improvement indicates that as the number of processed transactions increases, the ability to identify the addresses controlled by one user with only Heuristic 1 is increasingly enhanced, or the number of address reuse is decrease. The pure contribution to the ratio of address reduction of Heuristic 2 in addition to Heuristic 1 is 1.37%

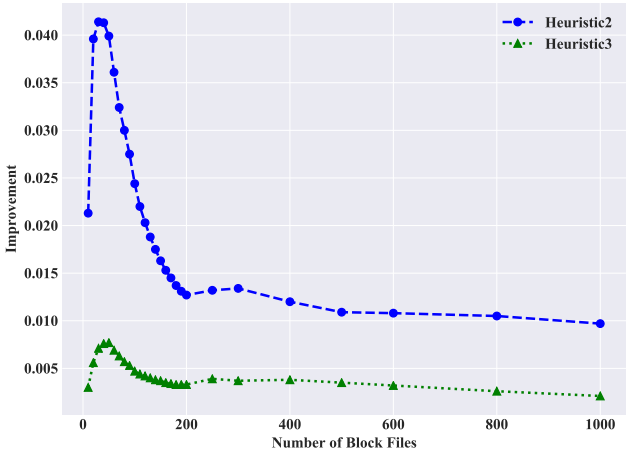

FIGURE 6. The pure contribution of Heuristics 2 and 3 to the ratio of address reduction.

on average, while Heuristic 3 contributes another 0.33% on average in addition to Heuristic 1 and 2. As a result, the number of addresses that can only be clustered with Heuristics 2 and 3 is reduced. Overall, Heuristics 2 and 3 make a stable contribution to address clustering even when the number of

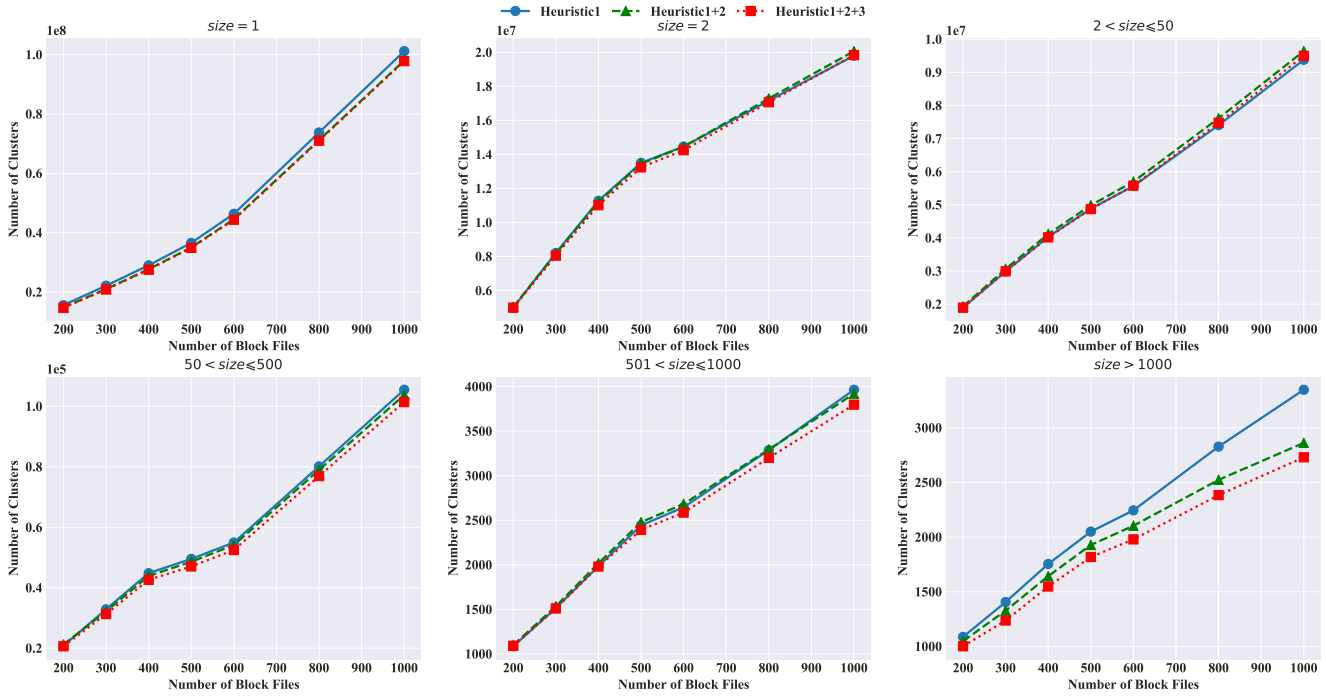

FIGURE 7. The trend of changes in the number of clusters with different sizes.

input transactions is large, but their contribution is relatively small compared to that of Heuristic 1.

C. RESULTS ANALYSIS

From Figure 5 and 6, we can see that the ratio of address reduction increase rapidly shortly after the creation of bitcoin and then reach a more stable plateau. This phenomenon may be caused by the fusion of input address sets triggered by address reuse. That is, if an address is being reused in the transactions, it is possible that there exist two transactions that contain this address in their input address sets, which implies that these two address sets are controlled by the same user. As a result, the number of clusters is reduced by one for each address reuse. At the beginning of the transaction history, the reduction in clusters caused by address reuse in new transactions is faster than the increase in new clusters, which causes a rapid increase in the reduction ratio. However, when the number of transactions is large enough, the speed of cluster reduction reaches a balance with the speed of the increase in new addresses. The change in the address reduction ratio slows and finally fluctuates around a fixed number when a balance is reached.

After a period of stable ratio of address reduction, the performance of address clustering among three settings decrease all together when the number of block files are more than 500. Especially, the decrease of Heuristic 1's performance is significant. We can see from Figure 7 that the number of clusters with size 1, i.e. the transactions with only one input, begins to increase faster at the point of 500 block files, which is around the date December, 2015. One possible explanation for that is that the market price of bitcoin begin to rise at the end of 2015 such that in more transactions, the amount of bitcoin in a single input is enough to pay or users intentionally reduce the number of inputs per transaction to save the transfer fees.

The Heuristic 2 and 3 works similarly by joining the detected one-time change address with the inputs, so the size of clusters usually increase by 1 (plus one) except it causes two clusters to merge. From Figure 7, we can observe the different behaviours of Heuristic 2 and 3. Firstly, in the transactions that trigger Heuristic 2 (or Heuristic 3), there should be almost the same number of transactions that have a single input and have two inputs, because the number of clusters with size 2 is almost the same with three different settings. The number of clusters with size larger than 2 and less than 50 increase after applying Heuristic 2 and 3, which should be the direct result of the "plus one" effect of these two heuristics. Third, both heuristics (especially Heuristic 3) trigger the merge of different clusters, which results in the decrease of number of clusters with size larger than 50.

In order to evaluate that whether there is a significant difference between experimental results of the method with the proposed heuristic and the baselines, we conduct a Analysis of Variance (ANOVA) test on the ratio of address reduction results of two different groups. Because the ratio of address reduction has another independent variable (factor), i.e., the number of input block files, a two-way ANOVA test is used, in which the alpha level is set to 0.05. The test results are listed in Table 2.

Because the p-value for both factors are far less than 0.05 and the F value for both factors are far larger than the F critical value, this indicates that both factors have significant effect on the ratio of address reduction. So there is significant difference between the method that uses the proposed heuristic and the baseline method.

TABLE 2. Results of two-way ANOVA test. Factor A is the number of input block files and Factor B is whether the proposed heuristic is used.

Sources	SS	df	MS	F	P-value	F crit
Factor A	0.1647	26	0.0063	5223.35	2.40E-42	1.93
Factor B	0.0003	1	0.0003	219.22	3.49E-14	4.23
Residual	3.15E-5	26	1.21E-6			
Total	0.1650	53				

VI. DISCUSSION

In this work, we propose a novel heuristic for address clustering in Bitcoin blockchain. Our results show that even after performing address clustering with two existing heuristics, our heuristic can still effectively identify addresses that are controlled by the same user with a pure contribution to the ratio of address reduction of 0.33% on average. It is obvious that the multiple input addresses heuristic applies to the greatest number of transactions and is very reliable before the introduction of services such as CoinJoin, PayJoin, CoinSwap and CoinJoinXT. As a result, it is the most powerful heuristic for address clustering, and the ratio of address reduction contributed by it is approximately 45.12% on average. There is no other heuristic that has such a large ratio of address reduction. For instance, the widely adopted one-time change address heuristic can only contribute another 1.37% address reduction on average in addition to the multiple input addresses heuristic. We can also see that as more independent heuristics are applied, there is less room for additional address clustering for the heuristics applied later. From this perspective, the 0.33% mean pure contribution of our heuristic in addition to the multiple input addresses and one-time change address heuristics is large enough to make our heuristic effective. This may motivate stakeholders to improve Bitcoin's underlying technology to increase privacy and foster research on cryptocurrency anonymity.

The main limitation of this work is that due to a lack of large-scale ground-truth labels indicating which addresses actually belong to the same user, it is very difficult to assess the quality of the proposed clustering heuristic, i.e., whether it will result in false positives or false negatives. The same issue is prevalent in existing clustering heuristics in Bitcoin as well. In future extensions of this work, we aim to construct a large-scale dataset based on the known information of address clusters to make the evaluation of address clustering methods much easier.

VII. CONCLUSION

In this article, we aimed to analyze the hidden links between different addresses in a Bitcoin blockchain through the transactions in order to obtain information for revealing the real identity of the users that control the addresses. In addition to the widespread input address-based and one-time change address-based heuristics for address clustering, we proposed a new address reuse-based heuristic. The experimental results of the address clustering algorithm showed that the ratio of address reduction increases rapidly at the beginning and then slows until reaching a plateau. An ablation study of the three heuristics showed that the input address-based Heuristic 1 contributes the most to all data scales, followed by Heuristic 2 and Heuristic 3. The heuristic proposed by us, Heuristics 3, can provide positive pure contributions to the performance of address clustering for different scales of input transactions and has a mean pure contribution of 0.33%.

For future work, it would be interesting to conduct a detailed analysis of the impact of address reuse in decreasing the pseudoanonymity provided by the Bitcoin blockchain. Regarding the address clustering heuristic, there is still room to make improvements by incorporating new heuristic tactics for other types of transactions to identify addresses that belong to the same user. The address clustering algorithm can also be used in address classification, transaction community identification, and abnormal transaction node detection to improve the performance of these algorithms.

REFERENCES

[1] E. Androulaki, O. G. Karame, M. Roeschlin, T. Scherer, and S. Capkun, "Evaluating user privacy in bitcoin," in *Financial Cryptography and Data Security*, A.-R. Sadeghi, Ed. Berlin, Germany: Springer, 2013, pp. 34–51.

[2] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of clients in bitcoin P2P network," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*. New York, NY, USA: Association for Computing Machinery, 2014, pp. 15–29.

[3] A. Biryukov and S. Tikhomirov, "Deanonymization and linkability of cryptocurrency transactions based on network analysis," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Jun. 2019, pp. 172–184.

[4] A. Biryukov and S. Tikhomirov, "Transaction clustering using network traffic analysis for bitcoin and derived blockchains," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2019, pp. 204–209.

[5] D. Ermilov, M. Panov, and Y. Yanovich, "Automatic bitcoin address clustering," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2017, pp. 461–466.

[6] A. Eshghi and M. Kargari, "Introducing a new method for the fusion of fraud evidence in banking transactions with regards to uncertainty," *Expert Syst. Appl.*, vol. 121, pp. 382–392, May 2019.

[7] M. Fleder, M. S. Kester, and S. Pillai, "Bitcoin transaction graph analysis," Feb. 2015, *arXiv:1502.01657*. [Online]. Available: https://arxiv.org/abs/1502.01657

[8] A. Gaihre, Y. Luo, and H. Liu, "Do bitcoin users really care about anonymity? An analysis of the bitcoin transaction graph," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1198–1207.

[9] S. Gao and D. Xu, "Conceptual modeling and development of an intelligent agent-assisted decision support system for anti-money laundering," *Expert Syst. Appl.*, vol. 36, no. 2, pp. 1493–1504, Mar. 2009.

[10] M. A. Harlev, H. S. Yin, K. C. Langenheldt, R. Mukkamala, and R. Vatrapu, "Breaking bad: De-anonymising entity types on the bitcoin blockchain using supervised machine learning," in *Proc. 51st Hawaii Int. Conf. Syst. Sci.*, 2018, pp. 1–10.

[11] M. Harrigan and C. Fretter, "The unreasonable effectiveness of address clustering," in *Proc. Int. IEEE Conf. Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People, Smart World Congr. (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, Jul. 2016, pp. 368–373.

[12] B. Haslhofer, R. Karl, and E. Filtz, "O bitcoin where art thou? Insight into large-scale transaction graphs," in *Proc. SEMANTiCS Posters Demos*, 2016, pp. 1–4.

[13] H. A. Jawaheri, M. A. Sabah, Y. Boshmaf, and A. Erbad, "Deanonymizing tor hidden service users through bitcoin transactions analysis," *Comput. Secur.*, vol. 89, Feb. 2020, Art. no. 101684.

[14] C. Kang, C. Lee, K. Ko, J. Woo, and J. W.-K. Hong, "De-anonymization of the bitcoin network usingaddress clustering," in *Proc. BlockSys*, 2020, pp. 489–501.

[15] G. Maxwell. *Coinjoin: Bitcoin Privacy for the Real World*. Accessed: Jun. 16, 2020. [Online]. Available: https://bitcointalk.org/?topic=279249
[16] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, "A fistful of bitcoins: Characterizing payments among men with no names," in *Proc. Conf. Internet Meas. Conf. (IMC)*, 2013, pp. 127–140.
[17] M. Möser, R. Böhme, and D. Breuker, "An inquiry into money laundering tools in the bitcoin ecosystem," in *Proc. eCrime Researchers Summit (eCRS)*, San Franciso, CA, USA, 2013, pp. 34–36.
[18] S. Nakamoto. (2009). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: May 14, 2020. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[19] T. Neudecker and H. Hartenstein, "Could network information facilitate address clustering in bitcoin?" in *Financial Cryptography and Data Security*, M. Brenner, K. Rohloff, J. Bonneau, A. Miller, P. Y. A. Ryan, V. Teague, A. Bracciali, M. Sala, F. Pintore, and M. Jakobsson, Eds. Cham, Switzerland: Springer, 2017, pp. 155–169.
[20] F. Reid and M. Harrigan, "An analysis of anonymity in the bitcoin system," in *Proc. IEEE 3rd Int. Conf. Privacy Secur.*, Amsterdam, The Netherlands, Oct. 2012, pp. 16–20.
[21] D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2013, pp. 6–24.
[22] M. Spagnuolo, F. Maggi, and S. Zanero, "Bitiodine: Extracting intelligence from the bitcoin network," in *Financial Cryptography and Data Security*, N. Christin and R. Safavi-Naini, Eds. Berlin, Germany: Springer, 2014, pp. 457–468.
[23] T. Tironsakkul, M. Maarek, A. Eross, and M. Just, "Tracking mixed bitcoins," in *Proc. 4th Int. Workshop Cryptocurrencies Blockchain Technol.*, Sep. 2020, pp. 1–17.
[24] F. Victor, "Address clustering heuristics for Ethereum," in *Financial Cryptography and Data Security*, J. Bonneau and N. Heninger, Eds. Cham, Switzerland: Springer, 2020, pp. 617–633.
[25] B. Zheng, L. Zhu, and M. Shen, "Malicious bitcoin transaction tracing using incidence relation clustering," in *Proc. Int. Conf. Mobile Netw. Manage.*, 2017, pp. 313–323.
[26] B. Zheng, L. Zhu, M. Shen, X. Du, and M. Guizani, "Identifying the vulnerabilities of bitcoin anonymous mechanism based on address clustering," *Sci. China Inf. Sci.*, vol. 63, no. 3, 2020, Art. no. 132101.

YUHANG ZHANG received the B.S. degree in electronic information science and technology from the Civil Aviation University of China, Tianjin, China, in 2005, and the M.S. degree in international trade from the Beijing Institute of Technology, Beijing, China, in 2008. She is currently pursuing the Ph.D. degree in management science and engineering with Beihang University, Beijing.

Her research interests include knowledge management, multisource heterogeneous data fusion and mining, online behavior analysis, and item recommendation.

JUN WANG received the Ph.D. degree in management sciences from Northeastern University, China, in 2003. He is currently a Professor with the Department of Information Systems, Beihang University, Beijing, China.

He has authored or coauthored more than 70 articles published in journals. His current research interests include information and knowledge management, social media analytics, big data analytics, business intelligence, and decision analysis.

JIE LUO (Member, IEEE) received the B.S. degree in mathematics from Peking University, Beijing, China, in 2003, and the Ph.D. degree in computer science from Beihang University, Beijing, in 2012.

From 2012 to 2019, he was an Assistant Professor with the School of Computer Science and Engineering, Beihang University, where he has been an Associate Professor since 2019. His research interests include mathematical logic, knowledge acquisition and reasoning, crowd intelligence theory, and formal methods for bug localization.