Tracking bitcoin users activity using community detection on a network of weak signals

Rémy Cazabet, Rim Baccour, Matthieu Latapy

Tracking bitcoin users activity using community detection on a network of weak signals

Cazabet Remy, Baccour Rym, Latapy Matthieu

Abstract Bitcoin is a cryptocurrency attracting a lot of interest both from the general public and researchers. There is an ongoing debate on the question of users' anonymity: while the Bitcoin protocol has been designed to ensure that the activity of individual users could not be tracked, some methods have been proposed to partially bypass this limitation. In this article, we show how the Bitcoin transaction network can be studied using complex networks analysis techniques, and in particular how community detection can be efficiently used to re-identify multiple addresses belonging to a same user.

1 Introduction

Bitcoin is a cryptocurrency, a digital currency using a technology called *blockchain* to maintain a public distributed ledger. Bitcoin is decentralized, i.e. there is no central authority. Instead, any peer can consult the distributed ledger at any time to check the amount of b itcoin controlled by each *address*, which is in fact a public key.

The bitcoin protocol therefore has to solve a difficult problem: on the one hand, one should not be able to track individual users' activity, while, on the other hand, all transactions must be public. This problem is partially solved by the possibility for users to use as many addresses as they want. As a consequence, a same user can use a different address each time he receives a payment, and observers cannot, in theory, associate these different addresses to the single user that control them.

Cazabet R.

Univ Lyon, UCBL, CNRS, LIRIS UMR 5205, F-69621, Lyon, France, e-mail: remy.cazabet@gmail.com

Baccour R., Latapy M., Cazabet R.

Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 75005 Paris France

In practice, however, previous works have shown that it is possible, using some simple heuristics, to re-identify several addresses belonging to the same user.

In this article, we show how community detection, a well-known network analysis technique, can be used to improve on these simple heuristics to better re-identify users.

In section 2, we introduce key elements of the Bitcoin transaction system, necessary to understand how re-identification is made possible. In section 3, we present existing heuristics for user re-identification, while section 4 introduces our method based on community detection. In section 5, we propose a framework to empirically assess the quality of re-identification heuristics, and compare the proposed method to existing ones.

2 Bitcoin transaction definition

To explain how re-identification techniques are made possible, it is first necessary to explain some key elements on how the Bitcoin cryptocurrency works. In this article, we will not describe all details of the *blockchain* and Bitcoin protocol, but only relevant information for the task of user re-identification. For more details on the protocol, one can refer to [13, 5, 8].

Bitcoin users can exchange amounts through *transactions*, recorded in a public ledger, the *blockchain*. These transactions are characterized by:

- *m* outputs, each associated with:
  - A bitcoin amount
  - A public key (address)

- *n* inputs, each corresponding to the output of a previous transaction (thus indirectly to its bitcoin amount and public key)
- A time stamp, which is an approximate value (due to the distributed nature of the blockchain, the timestamp is not exact, and in some cases, a transaction can take as input the output of a transaction with a later timestamp.

The sum of the inputs of a transaction is equal to the sum of the outputs plus an optional transaction fee.

Each address, which is in fact a *public key*, is associated with a *private key*. Both of them have been generated by the same user, and, while the public key is publicly shared, the pivate key is kept secret. To spend the bitcoins received at an address, its owner must "unlock" it by using its private key, as defined by the encryption protocol.

At this point, it is necessary to point out some important consequences of this mechanism:

- The balance of a user u at time t is the sum of all transaction outputs that can be unlocked by a private key owned by u, and that have not yet been unlocked (i.e. spent) at time t.
- If two transactions T_0 and T_1 have the same address d among their outputs, with respective associated amounts a_1 and a_2, these amounts are not merged or associated in any manner, i.e. there is no entity corresponding to the address d that controls, or contains, $a_2 + a_1$ bitcoins. Instead, these two outputs are uniquely identified respectively as output m_0 of transaction T_0 and output m_1 of transaction T_1, and each of these outputs can independently be used as input of a new transaction. Their only common trait is that they can be unlocked using the same private key, therefore by the same user.
- There is no partial spending: if a user who controls an output o_1 corresponding to x bitcoins use it as input of a transaction to send y bitcoins to another user, with $y < x$, then the user must send back $x - y$ bitcoins to an address that he/she controls.
- Users need to create multiple-input transactions each time they need to pay an amount superior to the value of each individual output they control.

3 Existing re-identification heuristics

Several works have tackled the problem of users re-identification. In this section, we will describe heuristics based on transactions present in the public ledger. Some other methods exist, using IP-addresses collected in real time while transactions occur [2] or global activity patterns for very active users [12]. One can refer to [10] for a recent survey on such methods.

Methods based on the public ledger are the most common ones. They have the advantage of being based on data easily accessible to anyone.

3.1 Heuristic based on transaction inputs only

A first heuristic for re-identification has been introduced in [14]: because all private keys corresponding to addresses must be used conjointly to sign a transaction, it can be safely assumed that all addresses in the input of a transaction belong to the same user. The introduction of the *multisig* protocol in 2013 does not dramatically change this observation: it allows an output to be controlled by several users (all users must use their private key to sign), but does not allow to use conjointly outputs that were initially detained privately by different users. Furthermore, these conjointly detained addresses are identified differently from normal ones, and are scarcely used (as of 2017).

This heuristic can be defined as:

Heuristic H 1 *All addresses used as input of the same transaction belong to the same controlling entity, called a User.*

A simple network algorithm can be used to discover these users efficiently:

1. Create a network with nodes representing addresses
2. For each set of n addresses used conjointly as input of a transaction, add $n - 1$ edges to create a path between them
3. Discover connected components of this network.

Connected components of this network correspond to sets of addresses belonging to a same user.

Despite the simplicity of this heuristic, it gives good results in practice, because users often have to use conjointly different addresses in order to perform transactions. Some users also tend to reuse the same addresses multiple times, either because they are not aware of the tracking possibility it allows, or because it is more convenient for them. For instance, the organization Wikileaks has an official address to which anyone can donate, therefore this address is used every time someone makes a donation to this organization. It is therefore possible to track how much money this organization receives through donations to this public address.

There is debate on the efficacy of this heuristic. Interested readers can refer to [6] for a discussion on why this method gives good results in most cases.

3.2 Heuristics based on inputs and outputs of transactions

H1 can only associate addresses because they simultaneously appear in the *input* of a transaction. Another property of bitcoin is that, because users cannot do *partial spending*, they have to send back to themselves the difference between the sum of inputs and the amount to pay. To improve the user discovery process, one needs to associate input addresses with one –or several– output addresses that belong to the same user, called *change address*, or *shadow address*. Several authors have proposed heuristics to discover which output address might be a change address.

Existing heuristics are searching for so-called *one-time change address*. Indeed, it is a common practice to receive the change on a newly generated address, that is never to be reused.

Androulaki et al. [1] proposed the following heuristic to discover *one-time change addresses*:

Heuristic H 2 *If there are exactly two output-addresses $a1$ and $a2$, that one of them ($a1$) appears for the first time and that the other ($a2$) has appeared before, then $a1$ is considered to be the change address.*

This heuristic has also been used in [15].

Meiklejohn et al. [11] propose a variation of this heuristic, not limited to transactions with exactly 2 outputs:

Heuristic H 3 *An address is considered a one-time change address if it satisfies the following properties:*

- *The transaction is not a coin generation*
- *The address is not among the input addresses (address reuse)*
- *It is the only output address appearing for the first time*

In their article, by using known addresses of several users as a reference, the authors are able to identify incorrectly grouped addresses, and propose a list of ad hoc restrictions to consider an address as a one-time change address, that involves checking if the address is not reused later, or reused only by some special services such as gambling games, that send their payout to the change address of the payment they received. The authors point out that these restrictions are only based on empirical observations, and cannot be generalized for future transactions.

A simple adaptation of the network algorithm described for **H1** can be used for **H2** and **H3**: a network is created as for **H1**, and an edge is added between each identified one-time change address and one of the corresponding input addresses. Connected components of this graph correspond to set of addresses controlled by the same user.

We can observe that there is a hierarchical relation between **H1** and **H2, H3**: each user of **H2, H3** corresponds exactly to one or several users in **H1**.

A more in-depth description of these approaches and their applications can be found in [10].

4 Proposed Heuristic for user re-identification

In this section, we propose to use community detection, a popular complex network analysis tool, to discover addresses belonging to a same user. The method is built on top of **H1** method, using its result as input.

The main idea is to create an *identity hint network*, in which edges between nodes represent hints that the corresponding set of addresses might belong to a same user. Because community detection searches for dense subgraphs, sets of nodes with multiple hints of belonging to a same user will be grouped together by the algorithm. Thus this solution exploits weak signals to re-identify users.

We propose to create the *identity hint network* using the following process:

Heuristic H 4 -

1. *A first level of aggregation is created by applying **H1**. Sets of addresses belonging to a same user are used as nodes of the hint network*
2. *For each transaction in the dataset, considering users found by **H1** instead of individual addresses, an edge is added (if not already present) between the (necessarily unique) sender and each recipient if:*

   - *there are less than 10 users in the ouput of the transaction (recipients)*

- *all recipients are different from the sender, i.e there is no already known change address*

On this network, a community detection algorithm is applied. Communities correspond to unique users.

In our experiments, we used a popular hierarchical community detection algorithm, the Louvain algorithm [3], that yields several solutions at different levels of aggregation.

For a given level, we consider that a community corresponds to a user, and that this user controls all addresses associated with nodes in the cluster. As with **H2** and **H3**, users found by **H4** can be associated with one or more users of **H1**, i.e. there is a hierarchical relation between them.

5 Evaluation of user discovery heuristics

To the best of our knowledge, the quality of user re-identification heuristics has never been assessed empirically. In this section, we propose a method to do so, using tools from *community detection* and *clustering*.

Let's assume that we have at our disposal a ground truth defined as a set of n labeled addresses $a_{GT} = \{a_1, a_2, \ldots, a_n\}$, with a^L the label of address a, corresponding to the user that controls it.

By using a heuristic on a transaction dataset, we discover sets of addresses identified as belonging to a same user. We assign a same label to all addresses of the same set, thus obtaining a set of m labeled addresses, a_H. By filtering out all addresses of a_H not present in a_{GT}, we obtain a set of addresses with two sets of labelling, the ground truth and a computed one. These labelling can be considered as clusters or network communities, and their similarity can be computed using, for instance, the NMI, chance adjusted NMI (aNMI) or Precision, Recall and F_1-score, which all give a different view on the tested clustering.

5.1 NMI and aNMI

The Normalized Mutual Information is frequently used to assess the correspondence between clustering or community partitions. It comes from information theory, and corresponds to the mutual information between the two compared partitions normalized between 0 –no similarity– and 1 –identical partitions. Several variants of the normalization can be used; in this article, we use the following definition:

$$NMI = \frac{I(U,V)}{\sqrt{H(U)H(V)}}$$

With $I(U,V)$ the mutual information between partitions U and V, and $H(U)$ the entropy (or quantity of information) of partition U. For more details on NMI, please refer to [16, 9].

Despite being often used in community detection evaluation, NMI suffers from known limitations, in particular that random partitions have expected values higher than 0, that can even be relatively high in some cases. A commonly used variant is the *adjusted for chance* NMI, called aNMI, which conserves the normalization but correct for the effect of chance. Please refer to [16] for more information on aNMI.

5.2 Precision, recall and F_1 score

Precision and Recall are typically used to assess the quality of classification tasks. They can also be used to evaluate the quality of clustering: Precision corresponds to how often elements are correctly grouped together, while Recall evaluate how often elements belonging to a same cluster are labeled as such in the tested partition.

More formally, for each pair of addresses, we check if they are **TP**,**FP** or **FN**:

- **TP** (True Positive): two addresses with the same label are in the same cluster
- **FP** (False Positive): two addresses with different labels are in a same cluster
- **FN** (False Negative): two addresses with the same label are in different clusters

Then Precision and Recall are defined as usual:

$$Precision = \frac{TP}{TP+FP}, Recall = \frac{TP}{TP+FN}$$

The F_1 score is the harmonic mean of Precision and Recall, and summarize the quality of a clustering in a single value.

5.3 Test dataset

In order to compare heuristics empirically, we need a labeled dataset of users and their known addresses. In the article [11], the authors have collected such a dataset by engaging in interactions with many bitcoin entities. In total, they engaged in 344 transactions with a variety of services, classified in 6 categories: Mining, Wallets, Exchanges, Vendors, Gambling and Miscellaneous. Please refer to the original article for more information on how the data was collected. After removing some particular cases, such as addresses of uncertain owners, or addresses controlled by the authors of the article, we were left with 776 addresses belonging to 90 different users. The user with the most known addresses, wallet service *instawallet*, controls 151 of them, while we know only one address for 33 users.

This dataset, due to its manual collection, is of relatively limited size, but still allows us to test our method and observe broad differences between heuristics. In

future works, it will be necessary to collect larger labeled datasets for robuster results.

For the bitcoin transaction dataset, we limited ourselves to the one used by the authors of the ground truth dataset [11]. It is composed of 16,086,073 transactions and 12,056,684 distinct addresses.

5.4 Results

We computed NMI, aNMI, Precision, Recall and F_1-score for **H1**, **H3** and all hierarchical levels of our proposed method **H4**. Results can be found in Tab. 1

Heur.	Precision	Recall	F_1	NMI	aNMI
H1	**0.98**	*0.77*	**0.86**	**0.89**	0.65
H3	*0.09*	0.83	*0.16*	*0.47*	*0.15*
H4-l1	0.75	0.79	0.77	0.86	0.66
H4-l2	0.50	0.87	0.63	0.81	**0.67**
H4-l3	0.27	0.90	0.42	0.70	0.47
H4-l4	0.25	**0.91**	0.39	0.67	0.43

Table 1 Empirical comparison of User re-identification algorithms. **H4-l1 to H4-l4** correspond to the increasing hierarchical levels of the community detection based method (**H4**). Highest values in bold, lowest ones in italic.

As expected, Precision score for the **H1** Heuristic is very high, which means that when two addresses are grouped together, they nearly always belong to the same user. By manually investigating the error, we found that addresses mistakingly grouped together belonged to two companies, one being in fact a branch of the other. More surprisingly, the Recall score is already relatively high for this heuristic.

By adopting **H3**, which creates larger clusters, the Recall score rises. However, this gain is made at the cost of an important decrease in Precision, thus resulting in lower overall scores (F_1, NMI and aNMI)

We can observe that, as expected, our method reach higher Recall than the **H1** method, at the cost of a lower Precision. However, compared to **H3**, the loss in precision is lower. In particular, partitions of the highest level (**H4-l3**,**H4-l4**) obtain higher scores than **H3** on all tested metrics (Pareto dominance). **H4-l1** and **H4-l2**, solutions of lower hierarchical levels, obtain highest aNMI score than the reference **H1** metric.

5.5 Visual representation of re-identification

To better understand the differences between different heuristics, we propose to use a visual representation. We propose to use alluvial diagrams to compare re-

identification algorithms results. In this visualization, each represented clustering is represented on a vertical axis; each address is represented by a horizontal, going from the cluster it belongs to in one clustering to the cluster it belongs to in the other clustering. A layout algorithm minimizes line crossings; horizontal lines with two common endpoints are grouped into a larger line, whose width corresponds to the sum of widths of addresses it represents.

To limit the visual complexity, we restrain the representation to one type of labeled users, which correspond to *wallet services*. Results for other types of users leads to the same kinds of observations.

Fig. 1 represents the correspondence between clusters in the ground truth and H1. We can observe that users controlling large sets of addresses tend to be well discovered (large horizontal lines for i*nstawallet, easywallet, paytunia, flexcoin*). We can also observe that there is no cluster in **H1** with addresses belonging to different users in the ground truth.

Fig. 2 represents the correspondence between clusters of addresses between the ground truth, **H1** and **H3**. We can observe in **H3** the presence of a large cluster (0) gathering addresses from different users in the ground truth. We can observe one occurrence (cluster 24620 in **H3**) of a cluster in **H3** rightfully grouping together distinct clusters from **H1**.

Fig. 3 represents the matching between clusters of addresses in the ground truth, **H1** and **H4-l2**. We can observe that some entities in the ground truth (*flexcoin*, *instawallet*, parts of *coinbase* and *easywallet*), are erronously merged in **H4-l2**, but less than with **H3**, as shown in Fig. 2. On the contrary, we observe several cases where different clusters in **H1** are correctly merged by **H4-l2** into clusters very close to those of the ground truth (for *paytunia*, *easywallet* and *coinbase*, in particular)

We can therefore conclude that the proposed method offer a better solution than the **H3** method, and even provide a relevant alternative to the widespread **H1** solution.

5.6 Advantages of the proposed method

On top of yielding a relevant user discovery solution, the community detection approach has several advantages:

- Because it allows to extract several hierarchical levels, one can choose, depending on the application, to increase Recall at the cost of Precision, or said differently, to discover more addresses for known users, at the risk of mistakingly merging different users.
- Because the solution is not based on a set of rules, but on *weak signals*, it is more difficult for users to develop countermeasures to escape re-identification of their addresses
- The proposed solution is a generic framework that can be adapted in countless ways to be improved: one could create a different *hint network*, using weighted edges based on rules decided by experts, or based on machine learning. One
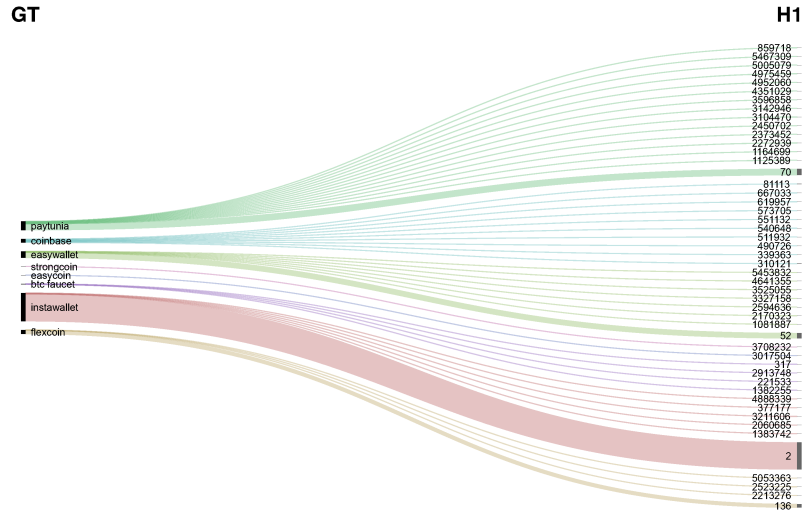
Fig. 1 Alluvial diagram representing the matching between clusters of addresses in the ground truth and **H1**. Each horizontal line corresponds to an address. A thick line corresponds to the aggregation of a set of addresses with common left and right belongings. We can observe that some entities (e.g *instawallet*) are already quite well identified by this heuristic, for the given test set. Nevertheless, for most entities, their addresses are divided into several clusters in H1.
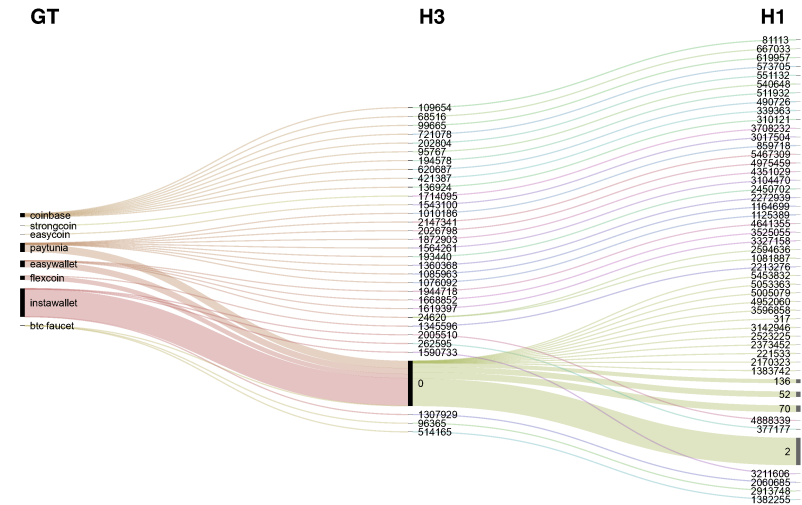

Fig. 2 Alluvial diagram representing the matching between clusters of addresses in the ground truth, H3 and H1. We can observe the apparition of a large cluster (named 0) that merge together many addresses corresponding in fact to different entities (Paytunia, easywallet, flexcoin, instawallet, etc.).
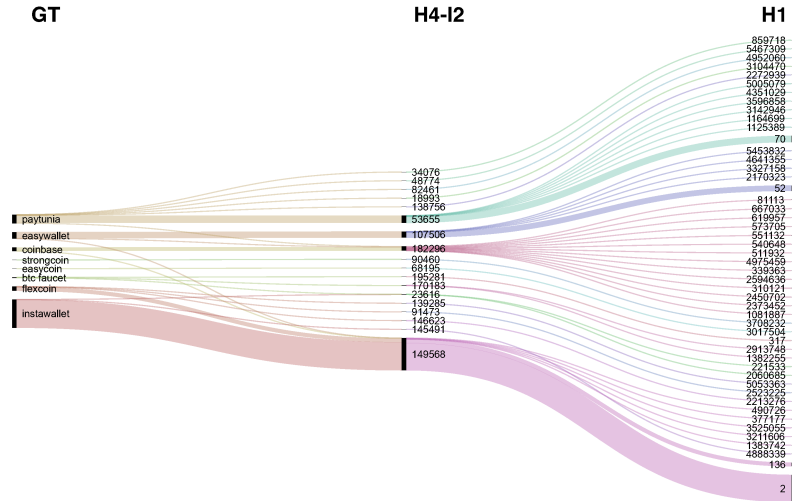
Fig. 3 Alluvial diagram representing the matching between clusters of addresses in the ground truth, H3 and H1. We can observe the apparition of a large cluster (named 0) that merge together many addresses corresponding in fact to different entities (Paytunia, easywallet, flexcoin, instawallet, etc.).

could also use other community detection algorithms. Finally, one could also take into account the dynamic of transactions, through temporal networks [7] and dynamic community detection [4].

6 Conclusion

In this article, we have shown how community detection could be used to improve re-identification of users in bitcoins.

We think that this article has two main contributions: on the one hand, it provides a framework for empirical testing and comparison of re-identification algorithms. The dataset used as ground truth is limited in size and representativeness, and further work is needed to produce more ambitious ground truth datasets in order to validate algorithms on a large scale.

On the other hand, we have shown how using complex networks analysis tools such as community detection could improve existing techniques, based on set of rules. In particular, contrary to set of rules techniques, they cannot be easily by-passed by users when publicly released.

References

1. Androulaki, E., Karame, G.O., Roeschlin, M., Scherer, T., Capkun, S.: Evaluating user privacy in bitcoin. In: International Conference on Financial Cryptography and Data Security, pp. 34–51. Springer (2013)
2. Biryukov, A., Khovratovich, D., Pustogarov, I.: Deanonymisation of clients in bitcoin p2p network. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 15–29. ACM (2014)
3. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008**(10), P10,008 (2008)
4. Cazabet, R., Amblard, F.: Dynamic community detection. In: Encyclopedia of Social Network Analysis and Mining, pp. 404–414. Springer (2014)
5. Garay, J., Kiayias, A., Leonardos, N.: The bitcoin backbone protocol: Analysis and applications. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 281–310. Springer (2015)
6. Harrigan, M., Fretter, C.: The unreasonable effectiveness of address clustering. In: Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences, pp. 368–373. IEEE (2016)
7. Holme, P., Saramäki, J.: Temporal networks. Physics reports **519**(3), 97–125 (2012)
8. Kosba, A., Miller, A., Shi, E., Wen, Z., Papamanthou, C.: Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In: Security and Privacy (SP), 2016 IEEE Symposium on, pp. 839–858. IEEE (2016)
9. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics **11**(3), 033,015 (2009)
10. Lischke, M., Fabian, B.: Analyzing the bitcoin network: The first four years. Future Internet **8**(1), 7 (2016)
11. Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G.M., Savage, S.: A fistful of bitcoins: characterizing payments among men with no names. In: Proceedings of the 2013 conference on Internet measurement conference, pp. 127–140. ACM (2013)
12. Monaco, J.V.: Identifying bitcoin users by transaction behavior. In: SPIE Defense+ Security, pp. 945,704–945,704. International Society for Optics and Photonics (2015)
13. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)
14. Reid, F., Harrigan, M.: An analysis of anonymity in the bitcoin system. In: Security and privacy in social networks, pp. 197–223. Springer (2013)
15. Spagnuolo, M., Maggi, F., Zanero, S.: Bitiodine: Extracting intelligence from the bitcoin network. In: International Conference on Financial Cryptography and Data Security, pp. 457–468. Springer (2014)
16. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. Journal of Machine Learning Research **11**(Oct), 2837–2854 (2010)