Examining Community Structure in Bitcoin User Network

Yokila Arora, Crystal Wu

December 12, 2016

1 Introduction

Bitcoin is an example of a peer-to-peer virtual currency which does not have a central authority (i.e. mint) to validate transactions. Instead, transaction management is collectively carried out by users of the network. Bitcoins are generated (mined) by using a CPU to find special solutions of a hash function. Once mined, they are assigned to a public address. All Bitcoin transactions are tracked on a public ledger where blocks of new transactions are "certified" as legitimate by what amounts to a majority of CPU power [Nak08].

The publicity of Bitcoin transactions provides opportunity for researchers to gain insights into transaction activities which are usually considered private and sensitive. In this paper, we study the network structure in detail and propose an algorithm to improve it. Further, we examine the community structure which emerges from Bitcoin transactions. Specifically, we are interested in comparing results from different community detection methods and evaluating the results to identify the most effective approach for Bitcoin user network.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 describes the dataset, the construction of the user network and provides basic statistics of the user network. Section 4 proposes an algorithm for improvement of the user network, based on pattern analysis. Section 5 explains the three community detection methods which were implemented. Section 6 evaluates the results and Section 7 concludes the study.

2 Literature Review

Most of the researchers use the data obtained from blockchains to develop two graphs, namely the user graph and the transaction graph for analysis. This process is listed in the paper by Reid et al. [RH13]. The transaction network consists of transactions as nodes and the flow of bitcoins from one transaction(as input) to another transaction as directed edges. User network contains users(public keys) as nodes and transactions between them as directed edges. Using this structure, we develop the user graph as defined in Section 3.2.

Taint Analysis and Pattern Analysis are two most commonly used methods to analyze network structure [Mös13]. Moser overviews the flow of bitcoins to determine frequently occuring patterns. We use pattern analysis and based on it develop an algorithm to improve the user graph.

There are many studies around community detection. K-means clustering [Llo82] partitions nodes based on Euclidean distances, which can be derived from specified characteristics of the nodes. Clique percolation method [PV05] can be used to identify overlapping communities built upon k-cliques. Clauset, Newman, and Moore proposed a modularity based graph clustering [CM14] which can be applied to very large network. These methods have their own importance in different network structures. We study these algorithms in detail and implement these algorithms on the bitcoin user network, to evaluate which works best on it.

3 Data Collection and Parsing

3.1 Data Source

The data set obtained is from the work of Ivan Brugere at the Laboratory for Computational Population Biology, University of Illinois at Chicago [Bru]. The dataset contains bitcoin blockchains upto 7th April, 2013 and comprises of 230686 blockchains.

The data is provided in the form of 6 text files, namely 'pubkey_list.txt', 'transactionkey_list.txt', 'user_edge_input_public_keys.txt', 'user_edge_inputs.txt', 'user_edges.txt' and 'userkey_list.txt'. The text files 'pubkey_list.txt' and 'transactionkey_list.txt' contain the list of all public keys and transaction keys. The file 'userkey_list.txt' groups the user keys which belong to the same user. Each line in the file 'user_edges.txt' contains the transaction id, the sender user id, the receiver user id, the date of the transaction and the value (in BTC). Figure 1 below describes these files and the relationship between them.

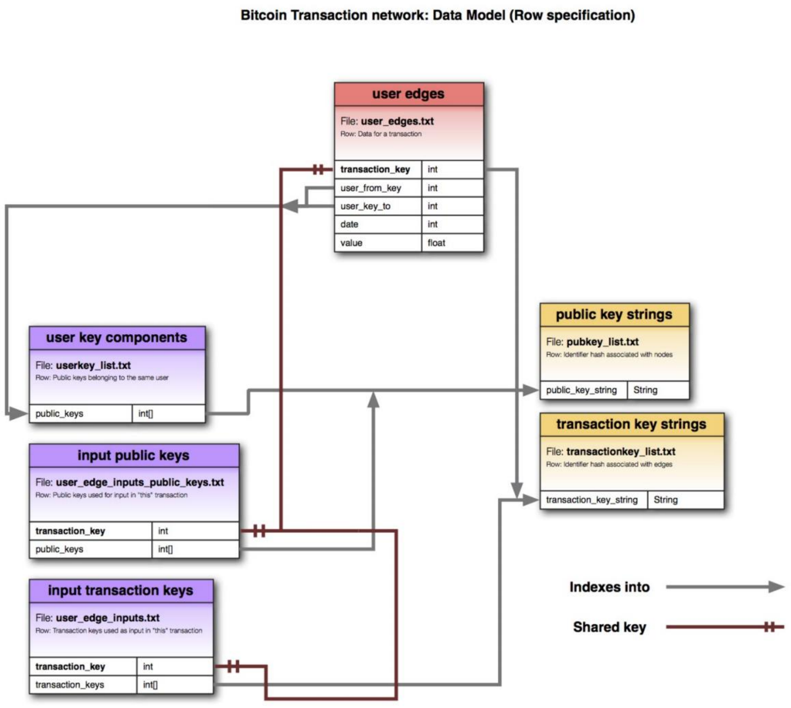

Bitcoin Transaction network: Data Model (Row specification)
user edges
File: user_edges.txt
Row: Data for a transaction
transaction_key | int
user_from_key | int
user_key_to | int
date | int
value | float
user key components
File: userkey_list.txt
Row: Public keys belonging to the same user
public_keys | int[]
input public keys
File: user_edge_inputs_public_keys.txt
Row: Public keys used for input in "this" transaction
transaction_key | int
public_keys | int[]
input transaction keys
File: user_edge_inputs.txt
Row: Transaction keys used as input in "this" transaction
transaction_key | int
transaction_keys | int[]
public key strings
File: pubkey_list.txt
Row: Identifier hash associated with nodes
public_key_string | String
transaction key strings
File: transactionkey_list.txt
Row: Identifier hash associated with edges
transaction_key_string | String
Indexes into
Shared key

Figure 1: Bitcoin network structure.

3.2 Network Construction

There are 15.89 million transactions and 11.88 unique public user keys in this dataset. As done in [RH13], we group the user keys that are used as input in a single transaction, i.e. if two (or more) public keys are used in a transaction then we assume that they belong to the same user. Figure 2 and Figure 3 explain this process of grouping. This yields us 6.3 million unique users(almost half of the original), which form the nodes of our network. The number of transactions between these users is 37.45 million, which form the edges of our network.

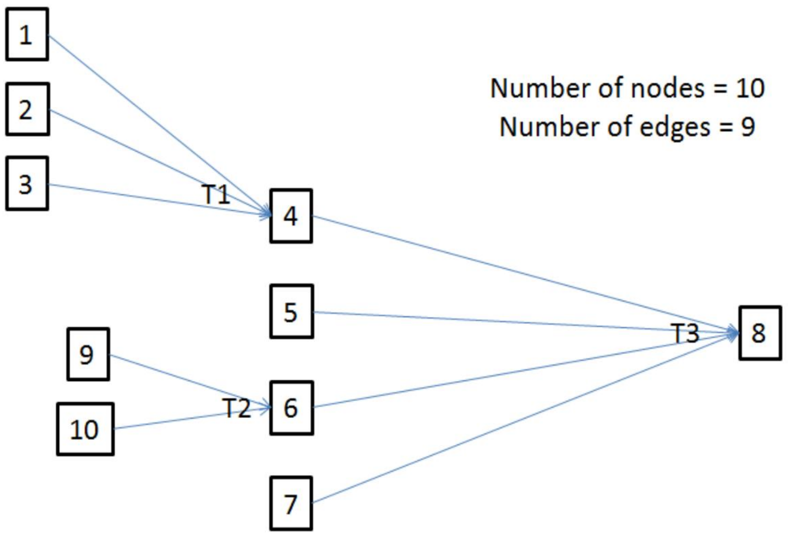


Figure 2: An example of the original network structure;
Nodes 1 to 10 represent user public keys;
T1,T2 and T3 represent the transactions.

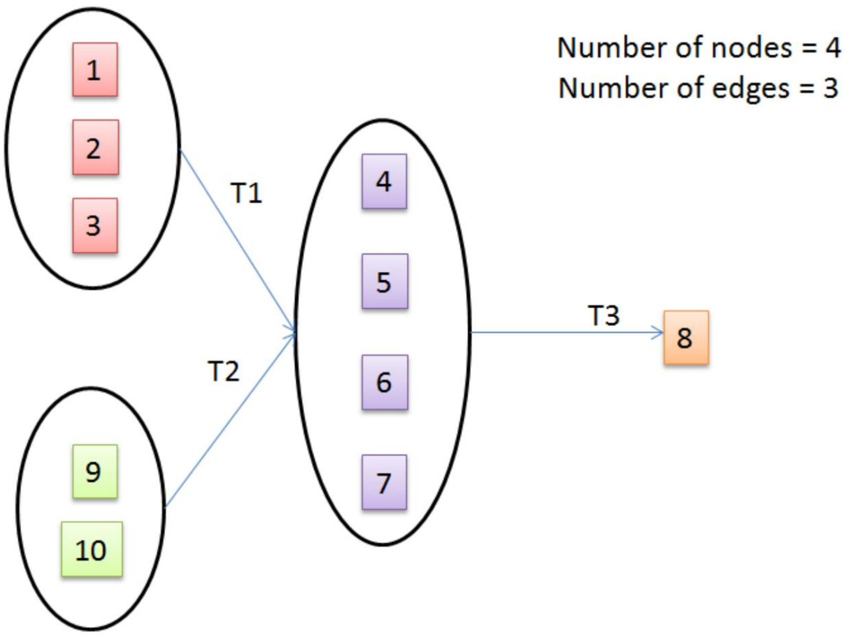


Figure 3: Network shown in Figure 2, after grouping the nodes;
the number of nodes has decreased to less than half in this case.

After grouping the public keys, we construct a directed graph where the users are represented as nodes and the transactions between them are represented as edges, going from the sender to the receiver. Each transaction has a timestamp and the number of bitcoins exchanged(BTC) associated with it. This representation allows us to learn the properties of the network and examine bitcoin mixing schemes.

The user graph is further modified in Section 4.4.1.

3.3 Basic Statistics

After constructing the user graph as mentioned in the previous section, we observe some basic properties of the network for analysis. Table 1 shows some statistics of the user graph generated.

Table1: User graph statistics

Total number of transaction keys	15898625
Total number of public keys	11885361
Number of nodes(users after grouping)	6336769
Number of edges	37450461
Number of self-edges	9307396
Number of unique bidirectional edges	5053993
Number of unique directed edges	16057711
Number of nodes with zero degree	0
Number of nodes with degree greater than 0	6336769

Further, we observe that the number of transactions in which the amount of bitcoins exchanged is less than 10, is 32114920, which is 85.75% of the transactions. Figure 4 shows the log-log plot of the out-degree distribution of the user graph. By this plot, we see that only a very few number of nodes have higher out-degree, and 90% of them have lower out-degree. Also, since this is a real-world network it should follow the power-law distribution but we can notice that the relation is not strictly linear. These properties show that there exist some anomalies in this network.

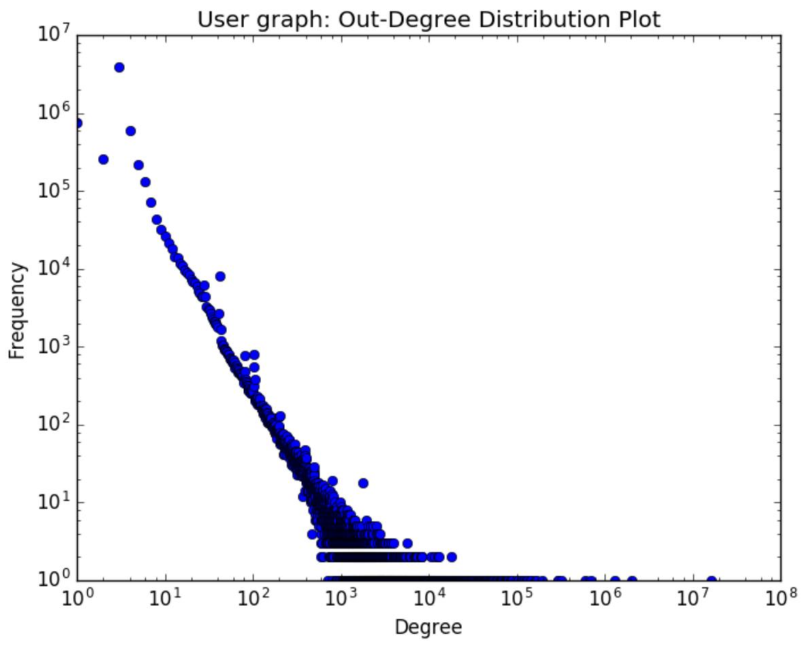


Figure 4: Log-log plot of out-degree distribution.

4 Improvement on User Network

We used pattern analysis to further consolidate the user network. We considered the Satoshi Dice related users, looked at the bitcoin flow pattern and then performed analysis based on it. Because of the size of the dataset, we only considered the transactions which took place in March 2013.

First, we observe that there are 21 transactions which are made from Node 25(SatoshiDice) to Node 67, and vice-versa. We notice that the dates of these transactions are exactly same. Timestamps for 18 of these transactions are exactly same, and for the rest 3 are very close. (In each of these 3 transactions, SatoshiDice returns the bitcoins late.) By looking at the amount of bitcoins exchanged, we see a relation between the bitcoins. In most of the cases, 76% of the bitcoins sent from SatoshiDice are returned back to it. In other cases, SatoshiDice returns 0.5% of the bitcoins sent to it. This is depicted by an example in Figure 5.

Also, we observe that Node 67 didn't make any transaction to any other node, i.e. all the out-edges from this node go to Node 25 only.

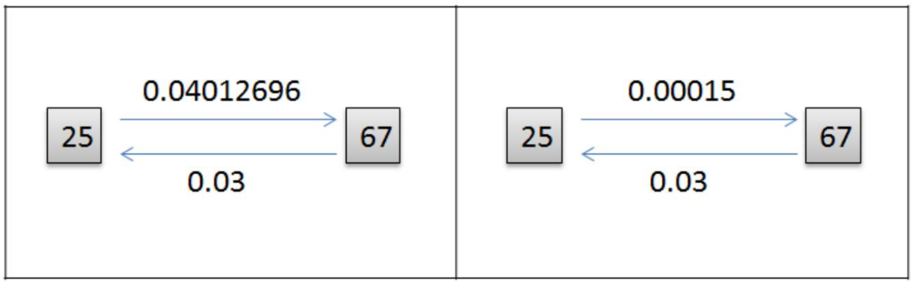


Figure 5: Example showing 76% and 0.5% bitcoin returns;
left transactions took place on 14th March, 2013 at the same time
and right transactions took place on 9th March, 2013 at the same time.

We observe that there are many nodes like Node 67, which only have out-edges to Node 25. This might be because these nodes are related to SatoshiDice, and are used to only collect bitcoins from its users.

Taking this into consideration, we modify the user graph as given in the Algorithm 1 below. In this algorithm, we group a node u with another node w if u only has out-edges to w, and no other node. It should be noted that the number of edges from u to w should be more than one, for this grouping. Figure 6 and Figure 7 depict this grouping procedure. (It may be noted that we have not considered self-loops(edges (u,w) and (w,u)) for this implementation, but they can be added easily.)

Algorithm 1 Modify user graph $G = (V, E)$

```
1: for each node $u \in V$ do
2:     Create a list $l$, which consists of all nodes $v$ s.t. $(u, v) \in E$
3:     if $l$ contains only one node, say $w$, repeated $n$ times($n > 1$) then        ▷ Merge $u$ and $w$
4:         if $u \neq w$ then
5:             for each node $x \in V$ s.t. $(x, u) \in E$ do
6:                 Add edge $(x, w)$                ▷ Make all incoming edges to $u$ point to $w$
7:             Remove node $u$
```

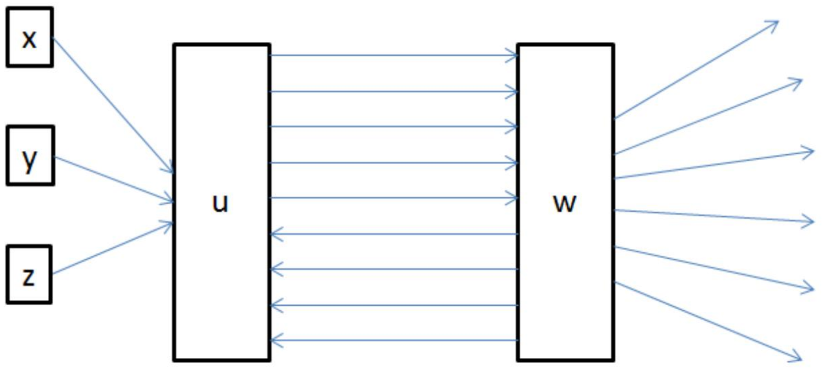


Figure 6: Sample network before grouping;
Node u has out-edges to Node w only.

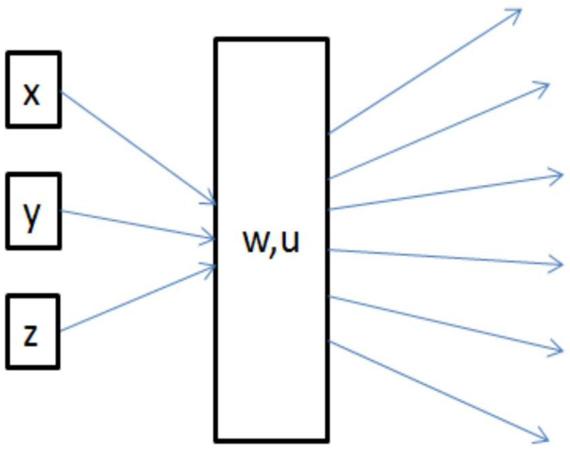


Figure 7: Network shown in Figure 6, after grouping using Algorithm 1.

5 Community Detection

We used three community detection approaches and compared the results in order to evaluate the pros and cons for each methods. In order to get results within reasonable runtime, we used the Bitcoin user network within a date in 2013 for this project (the rationale for the date selection will be explained later in the paper). Based on the graph below, Bitcoin network was becoming increasingly active in 2013.

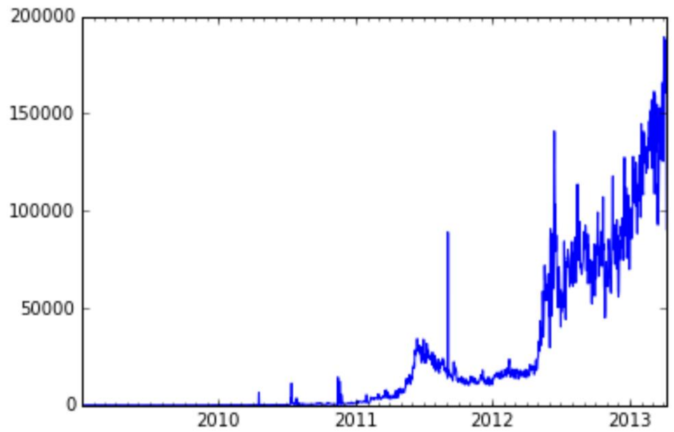


Figure 8: Transaction Count by Dates

The three approaches we used are:

- K-means clustering [Llo82]
- Clique percolation method [PV05]
- Modularity based graph clustering, proposed by Clauset, Newman, and Moore in 2004 (i.e. CNM) [CM14]

5.1 K-means clustering

K-means clustering method is to partition nodes into k clusters based on specified characteristics [Llo82]. For k-means to work, each node needs to be represented as a multi-dimensional vector in the Euclidean space for distance calculation. In order to account for the network structure of the Bitcoin user network, we selected the following features for each node:

- In-degree
- Out-degree
- Clustering coefficient

The procedure of k-means clustering aims to partition all nodes into k clusters in order to minimize the target function:

$$\sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|$$

where μ_i is the centroid of cluster S_i.

We selected k = 350 to be close to the final number of clusters produced by the CNM algorithm. However, the sizes of the top communities identified in k-means clustering were not much impacted when we adjusted k from 100 to 350.

5.2 Clique percolation method

The clique percolation method is used when detecting overlapping community structure of networks [PV05]. The clique percolation method follow:

1. Finding k-cliques of the graph, which are the complete sub-graphs of k nodes. Each k-clique is considered as a node in the clique graph;
2. Two k-cliques are adjacent in the clique graph if the two cliques share k-1 nodes;
3. Communities are identified as the connected components in the clique graph

Since each node can belong to multiple cliques, this approach allows overlapping between communities. When deciding which k to use, we tried $k = 3$ and $k = 4$. When $k = 4$, we didn't find enough 4-node complete sub-graphs which would cover significant part of the constructed Bitcoin user network. Thus, we used $k = 3$ in the implementation.

5.3 Clauset, Newman, and Moore Algorithm

The CNM algorithm is a modularity based graph clustering method, which uses a greedy approach to maximize "modularity" of the resulted community structure [CM14]. The Modularity Q is defined as:

$$Q = \frac{1}{2m} \sum [A_{ij} - \frac{k_i k_j}{2m}] \delta(c_i, c_j)$$

where A_{ij} represent the adjacency matrix, and $\delta(c_i, c_j)$ indicates whether the two nodes belong to the same community. In each step, two communities are selected to merge so that the incremental modularity gain is maximized.

The advantage of CNM algorithm is that it decreased the runtime in a network with n nodes and m edges from $O(m^2 n)$ in Girmvan-Newman algorithm to $O(mdlogn)$, where d is the depth of the dendrogram representing the hierarchical decomposition of the network into communities structured through the greedy process. This increase in speed allows community-finding to be applied to much larger networks.

6 Results

We constructed the Bitcoin user network for the date 2013.01.26, and tested the three community detection approaches listed out in Section 4.4. As described in Section 4.3, we used the known public keys of the user SatoshiDice (i.e. public keys with prefix "1dice") to evaluate the result. Thus, we selected the date with the most active SatoshiDice users. There are 10 active users related to Satoshi Dice on the date selected for the project.

6.1 Basic Network Statistics

After implementing Algorithm 1, we compare some statistics of the original graph and the new graph obtained. Both graphs are considered after removing self-edges.

	Original Graph	**New graph**
Number of nodes	26939	26606
Number of edges	86595	73155
Number of unique bidirectional edges	19437	13145
Number of unique directed edges	48364	47829
Number of nodes with degree=0	29	34
Number of nodes with degree greater than 0	26910	26572

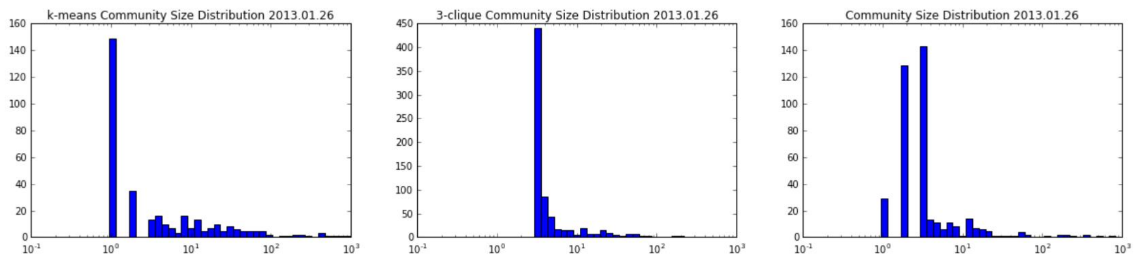
We observe that the number of nodes has decreased substantially even in one day. All the numbers except the number of nodes with zero degree has increased. This may happen when some nodes point to each other only. Further analysis can be done on the graph to analyze this better.

6.2 Distribution of Community Sizes

The three community detection algorithms generated very different results in terms of community sizes. The results are compared in the chart below:

	Number of Communities	**Size of Maximum Community**
K-Means	350	8,303
Clique Percolation	713	1,448
CNM	414	3,216

We put community sizes distribution in histograms:



K-Means, with k = 350 Clique Percolation, with k = 3 Modularity-Based CNM

Figure 9: Approach Comparison - Community size vs. Number of Communities

- **K-means clustering:** k-means clustering produces one very large cluster and many 1-node clusters, due to the power law on degree distribution. In addition, since k-means is based on selected node characteristics, the information on network structure is not fully exploited. Even though this is not the best approach to detect communities, this is the fastest approach of all the three and is efficient to classify nodes based on specified features.
- **Clique percolation method:** Clique percolation produces the smallest community sizes. The identified communities are generally more closely related due to the construction process. However, over half of the detected communities have the size of 3, which is the size of the pre-defined clique. Also, if a node does not belong to any clique, which in our case means not being part of a triangle, it won't be considered in the procedure.
- **Modularity based graph clustering CNM:** Out of the three clustering methods we tested, CNM utilized the most information of the network. In later sessions, we will show CNM produce the best community detection results for Satoshi Dice.

6.3 Community Detection of Satoshi Dice

After obtaining the results, we tagged the 10 user nodes identified as Satoshi Dice in order to evaluate which method produces better community detection result. The assumption is that nodes related to Satoshi Dice are likely to belong to the same community.

- **K-means clustering:** 10 Satoshi Dice nodes were spread out in 9 different clusters sized from 1 to 322. K-means was not able to detect the Satoshi Dice community.
- **Clique percolation method:** 5 out of the 10 Satoshi Dice node were not part of any triangles and thus not considered in the clustering procedure. Among the rest 5 nodes, 1 node (Node 25) with the highest degree was included in multiple clusters sized from 3 to 1448. Also, the same node shared community with two other Satoshi Dice nodes. Since significant portion of the Satoshi Dice nodes were not part of a 3-clique, this method was not able to detect the Satoshi Dice community either.
- **Modularity based graph clustering CNM:** 10 Satoshi Dice nodes were found in 5 different communities. Among those, 6 nodes were included in the same community of size 179. Thus CNM was able to successfully cluster most of the Satoshi Dice nodes into the same community.

7 Conclusion

We observe that using the proposed network improvement algorithm, the number of nodes has decreased. Out of the three community detection algorithms implemented, CNM works the best in detecting communities based on network structure, and generates most accurate result in clustering SatoshiDice nodes.

References

[Bru] Ivan Brugere. Dataset. http://compbio.cs.uic.edu/data/bitcoin/.

[CM14] Newman M.E.J Clauset, Aaron and Cristopher Moore. Finding community structure in very large networks. In *Phys. Rev. E70, 066111, 2004*. MBC, 2014.

[Llo82] Stuart P. Lloyd. Least squares quantization in pcm. In *IEEE Transactions on Inforamtion Theory 28(2): 129-137*. IEEE, 1982.

[Mös13] Malte Möser. Anonymity of bitcoin transactions, an analysis of mixing services. In *Münster Bitcoin Conference (MBC), 17 - 18 July '13, Münster, Germany*. MBC, 2013.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[PV05] Derenyi Imre Farkas Illes Palla, Gergely and Tamas Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. In *Nature 435, 814*, 2005.

[RH13] Fergal Reid and Martin Harrigan. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*, pages 197–223. Springer, 2013.