Community Detection and Analysis in the Bitcoin Network
CS 224W Final Report

April Yu Benedikt Bünz

December 9, 2015

1 Introduction

Bitcoin is a decentralized payment system and electronic cryptocurrency first published in 2009, which has steadily grown to a market cap of around $4 trillion USD and over 150k transactions a day. Through the use of ingenious cryptographic techniques to sign transactions and determine control of funds, Bitcoin is a peer-to-peer system where users transact directly, without need for an intermediary, and in which transactions are recorded, for posterior verification by all nodes, in a public, distributed ledger called a Blockchain. Given the public nature of the Blockchain, in which every transaction is recorded, it provides an excellent source of data from which to infer and analyze flows of funds and connections between users.

Each transaction in the Blockchain is represented as a set of input addresses, from which funds are drawn, and a set of output addresses, to which those funds are sent. Users typically have one (or more) wallets, from which they can generate addresses to hold and receive funds; to send a specific amount to a specific address, the software chooses which input addresses to draw funds from, specifies the amount being sent to the output address, and adds other addresses in the senders control as outputs in case change needs to be sent back, i.e., the amounts in input addresses exceed those the user intends to send, given that for any input address, its amount must be entirely sent to other addresses (it is not possible to withdraw only a fraction of whats stored in an address).

Given the wide array of uses for bitcoin - as a money transmission mechanism, as a store of value, as a way to avoid international remittance fees, as a way to facilitate currency trading, and even as a way to facilitate gambling and trade in illegal merchandise - the potential for insights to be gleaned from detecting and analyzing the communities composing the network is immense.

2 Related Work

The privacy of the Bitcoin transaction has been one of its most discussed properties since its inauguration in 2008 [Nakamoto, 2008]. Specifically with the rise of darknet markets, selling drugs and other illicit goods for bitcoin, the discussion about Bitcoins privacy has even reached the mainstream media (http://www.wired.com/2015/04/silk-road-1/). Interestingly the case of Silk Road also shows that with the analytical power of federal agencies the privacy of Bitcoin can in fact be broken. In 2014 the FBI took down Silk Road and arrested its founder who later got sentenced to life in prison. The FBI analyzed the Bitcoin network as well as other online fingerprints to identify the creator and owner of the site.

The privacy of Bitcoin has also been investigated by researchers. Most importantly Ron and Shamir [2013], have done an in-depth analysis of the transaction graph with the specific focus on statistical properties of the network. Fleder et al. [2015] have investigated how to map identities, scraped from public forums, to Bitcoin addresses. Androulaki et al. [2013] also evaluated the user-privacy in Bitcoin. In particular they conducted a field experiment with students who were attempting to maintain privacy. Despite this the authors were able to break anonymity for 40% of the users.

2.1 Community Detection

We employed various algorithms to determine communities within the bitcoin blockchain. Newman and Girvan propose the use of a divisive, rather than agglomerative, strategy, in which the algorithm starts with the complete network and removes edges. As the divisions proceed, the network gets split up into successively smaller communities, and the process can be stopped at any point to yield one potential division of the network into communities.

Clauset et al propose an algorithm that decreases the runtime of finding communities of a network with n vertices and m edges from $O(m^2n)$ to $O(md \cdot log_2n)$, where d is the depth of the dendrogram describing the community structure. This increase in speed allows community-finding to be applied to larger networks, as the speed of past algorithms limited its use to graphs of at most a few thousand vertices.

Blondel et al propose a greedy algorithm that optimizes the calculation of modularity in order to achieve a runtime of $O(n \cdot log(n))$. While the mathematical definition of modularity is the same as that of Clauset et al, the Louvain algorithm (named after the university in which it was developed) optimizes the use of modularity within the algorithm.

3 Data Set

The data was obtained from the work of Ivan Brugere at the Laboratory for Computational Population Biology, University of Illinois at Chicago [1]. The dataset we will use in this project is a blockchain snapshot from April 7, 2013. Figure 1 describes the relationship between the various data files:

[1] http://compbio.cs.uic.edu/data/bitcoin/

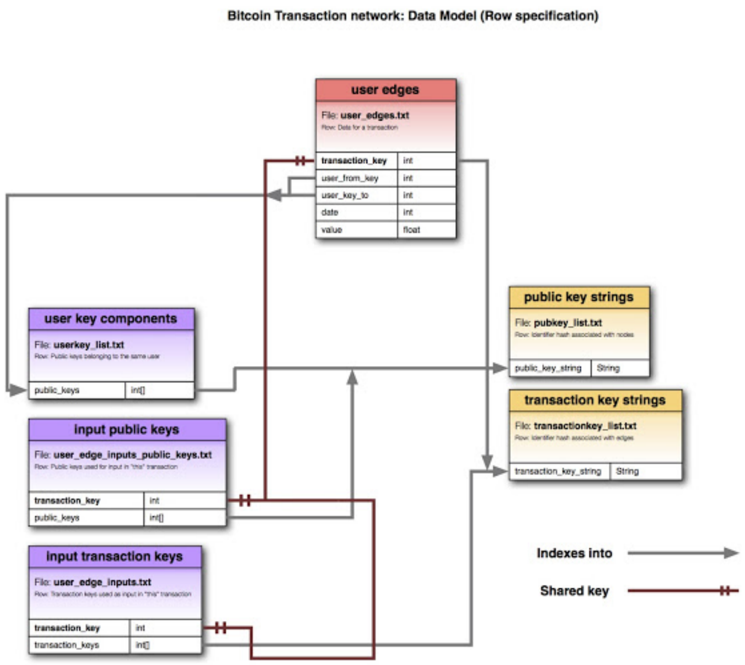

Bitcoin Transaction network: Data Model (Row specification)
user edges
File: user_edges.txt
Row: Data for a transaction
transaction_key
int
user_from_key
int
user_key_to
int
date
int
value
float
user key components
File: userkey_list.txt
Row: Public keys belonging to the same user
public_keys
int[]
input 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 Dataset Structure

The edges for the graph are obtained from the ”user_edges.txt” file. This is the main file for our implementation but it also references information in other files. The transaction_key in user_edges.txt refers to a row (i.e. a transaction) in transactionkey_list.txt. Additionally, the user_from_key and user_key_to refer to rows in the userkey_list.txt file, which lists public keys that belong to the same user. Both the transaction and public keys can be queried on blockchain.info for more information.

4 Dataset Stats

The dataset described above has 11.8 million unique public keys and 15.9 million unique transaction keys. From this, we end up with a graph with 6.3 million nodes and 37.4 million edges. The nodes are unique users (which can be comprised of multiple public keys) and the edges are unique transactions between the users. We explain more about the network model below. When first examining our dataset, we discarded any transactions whose value was below 1 BTC and then examined the resulting graph. The core graph statistics are displayed in Table 1.

Table 1: User Graph Statistics

Nodes	4399335
Edges	6625879
Zero Deg Nodes	0
Zero InDeg Nodes	21901
Zero OutDeg Nodes	145316
NonZero In-Out Deg Nodes	4232118
Unique directed edges	6625879
Unique undirected edges	6527472
Self Edges	166672
BiDir Edges	363486
Closed triangles	717347
Open triangles	3539064462
Frac. of closed triads	0.000203
Connected component size	0.988909
Strong conn. comp. size	0.711372
Approx. full diameter	6926
90% effective diameter	655.643277

To avoid the bottleneck that such a large dataset would pose on our experiments, we isolated a subset of the nodes and edges, representing the transactions for a single day. This is the dataset on which we ran our various algorithmic implementations. The small dataset statistics are displayed in Table 2.

Table 2: Statistics for Single-Day Transactions

Nodes	43615
Edges	85913
Zero Deg Nodes	0
Zero InDeg Nodes	0
Zero OutDeg Nodes	0
NonZero In-Out Deg Nodes	43615
Unique directed edges	171826
Unique undirected edges	85913
Self Edges	0
BiDir Edges	171826
Closed triangles	23197
Open triangles	19180463
Frac. of closed triads	0.001208
Connected component size	0.919661
Strong conn. comp. size	0.919661
Approx. full diameter	23
90% effective diameter	6.767017

Additionally, in this single-day dataset, we removed all the self-edges for all nodes to optimize the runtime of our various algorithms.

5 Implementation

Our analysis consists of three separate steps. The first step is to construct a network of Bitcoin users. For this we use the structure of Bitcoin transactions to cluster addresses that with high likelihood are controlled by the same user. This first step is already a significant aggregation of addresses.

The second step is using network analysis tools to find communities in the user to user transaction graph. Here we leverage sophisticated graph analysis tools to find users that often interact with each other.

The final step is analyzing communities and identifying patterns and real-world entities that are connected to them. This allows us to link real-world identities to Bitcoin addresses and determine usage patterns in Bitcoin.

5.1 Network Construction and User Grouping

One of the main challenges of any network analysis is how to model the problem at hand as a network. This challenge is also present in the Bitcoin analysis literature. In general two approaches have been considered and analyzed.

In the introduction we explained that in bitcoin money is sent from address to address. While this is practically accurate, it omits an important underlying technical property of Bitcoin. Each transaction takes as input multiple still-unspent transactions and creates a set of outputs (each potentially belonging to a different address). Transactions have to be spent in full but can be split up. Assuming that Alice has received a transaction of 1 BTC from Bob she can send .5 BTC to Charlie by taking the 1 BTC transaction as output and sending .5 BTC to Charlie and .5 BTC back to herself.

The Bitcoin blockchain is a network of transactions. One can, thus, intuitively model the blockchain using transactions as vertices and creating a directed weighted edge from each input transaction to an output transaction with the value of the transaction being the weight. This graph is stylized in Figure 2.

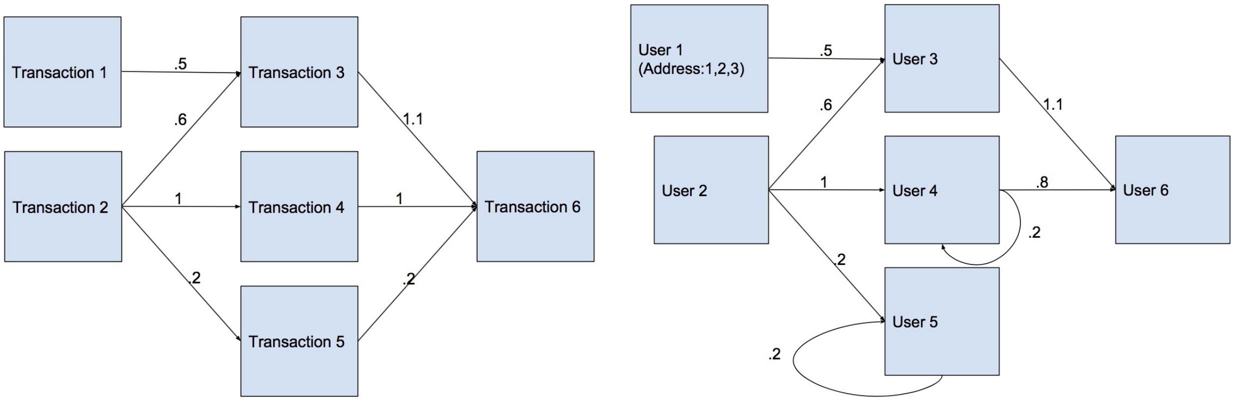


Figure 2: Transaction and User Network

While this approach is accurately modeling the underlying structure of the Bitcoin network, it drops important information which are vital when analyzing the structure of the Bitcoin network. Most importantly it does not model the fact that multiple transactions can be associated with the same address that is, by definition, controlled by a single user. In the stylized example above, transaction 3, 4 and 5 could all be associated with the same address. Additionally an address can at any time be associated with many unspent outputs. If a user wants to send funds from one address to another he will choose outputs somewhat arbitrarily from his available unspent outputs. The edges are thus created to some degree arbitrarily and other edges that could have been created with equal probability are not present.

With the limitations of the transaction network in mind we consider a different network that connects the interaction between users rather than transactions. Each address belongs to a single user. This is the case because every address is associated with a unique private key. Sharing that key is equivalent to sharing control over an address, at which point the two entities have shared control over the funds at that address and can, thus, be viewed as a single user. A simple model would thus create one node per public key. However a single transaction can take money from multiple addresses and move it to multiple different addresses. Without the use of hyper-edges, this becomes difficult to model. To circumvent this we can use the heuristic described by Reid and Harrigan [2013] to combine different public keys. Specifically we can assume that a transaction is constructed by a single user and thus that all the inputs are controlled by the same entity. For each output of a transaction we can then add a directed edge between the input user and the output user. The edge is weighted by the value of the transaction. The graph can have self edges when a user sends money to himself. Additionally there can be more than one edge between two users. We describe in the future work section how to combine them.
Another heuristic that can be used to combine users is looking at so called change addresses [Androulaki et al., 2013]. Since in Bitcoin transactions have to spent full inputs, there is often change associated with a transaction. This change usually stays in controll of the sender. It is common practice to generate a new and previously unused address for this change. If a transaction, thus, has multiple outputs and only one of them is unused then we can assume that this is the change address which is under the controll of the sender.
The aggregated user network is stylized in Figure 2.

5.2 Community Detection algorithms

We implemented the algorithms from the 3 papers discussed in the "Related Work" section following the user grouping detailed above.

We began with the Girvan-Newman algorithm, in which we removed edges on the betweenness criteria based on the shortest path. Since betweenness favors edges that lie between communities as opposed to those that lie within them, removing them tends to disjoint the communities and make their separation evident.

Following the shortest path betweeness criteria, our algorithm operates by finding the shortest paths between all pairs of vertices, and counting how many run along each edge. We then compute these betweenness scores for all edges in the network, find the edge with the highest score and remove it, and then recalculate the betweenness for all remaining edges. Then we find the edge with the next highest score and repeat. To choose at which iteration to stop, we, as following the proposal by Newman and Girvan, utilize the modularity of the network (Q). By calculating Q at each step of the dendrogram and finding the local maximum, the algorithm finds the best natural division of the network.

This algorithm, with the betweeness criteria based on the shortest path, runs in O(m*n) on a graph with m edges and n vertices. Given that even our small dataset has 43615 nodes and 85913 edges, we were unable to successfully run this algorithm on our dataset without utilizing all available memory. Therefore, while we explored this algorithm to its fullest potential, we were unable to utilize it for our experiments.

We then implemented the Louvain algorithm. As with Clauset et al, the algorithm begins by considering each node in the network as its own community. Then, for each node i, the change in modularity is calculated by removing i from its own community and moving it into the community of each neighbor j:

$$\Delta Q = \left[\frac{\sum_{in} + deg(i,in)}{2m} - \left(\frac{\sum_{tot} + deg(i)}{2m}\right)^2\right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m}\right)^2 - \left(\frac{deg(i)}{2m}\right)^2\right] \tag{1}$$

where $\sum_{in}$ is the sum of all the weights of the edges inside the community i is joining, $\sum_{tot}$ is the sum of all the weights of the edges to nodes in the community, $deg(i,in)$ is the sum of the weights of the edges between i and the other nodes in the community and m is the sum of the weights of all edges in the network. Once the change in modularity is calculated for all communities i is connected to, node i is placed in the community that results in the largest increase in modularity for the network. If there is no possible increase, i remains in its original community. This is repeated until no modularity increase is possible within the network.

Whether by programmer error or simply the nature of our network, this algorithm was unable to divisively separate our network into smaller, more understandable communities. With these results, we turned to the Clauset et al algorithm to run our experiments.

Finally, we implemented the Clauset et al algorithm. This depends on the property of modularity, which measures the number of edges within a community against the number of edges between communities. We followed their proposal and calculated modularity to be:

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v \cdot k_w}{2m}\right] \delta(c_v, c_w) \tag{2}$$

Utilizing this definition of modularity, our algorithm starts with each node representing a single community of one and repeatedly joins together two communities whose combination optimizes for the greatest increase in modularity value. The algorithm tracks the modularity values to avoid costly computations that simply merge elements with value 0.

This implementation of the Clauset et al algorithm had the fastest runtime of the three algorithms we implemented. Because of this and the various complications we encountered with our implementations of the Girvan-Newman and Louvain algorithms, we utilized this algorithm in particular when running our experiments in trying to utilize the communities within the bitcoin network to de-anonymize the blockchain.

5.3 Attaching Identities

There are many ways to attach identities to Bitcoin addresses. Another way to link addresses to real world identities is to monitor the IP address that broadcasted a transaction originating from an address. Especially with police capabilities it can be easy to use IP addresses to determine location and identity of a sender. On the other hand, it is also easy to mask once ip address using VPN, proxies or the TOR network.

Another common way to attach an address to an identity is to interact with said identity. If we can get a person or company to send us some bitcoin, we can easily link the sending address to the user. Additionally we can also send money to a user and trace it's further movement. This methodology has been used to identify funds belonging to bitcoin exchanges.

While the first two options were not easily available to us, we focused on a third one. Some Bitcoin services have publicly known addresses that are commonly used. For instance, gambling sites intentionally create addresses that have a common prefix, such as (**1dice**2pxmRZrtqBVzixvWnxsMa7wN2GCK) for marketing purposes. Another instance of public addresses is when they are publicly posted by the user himself. This can be the case if a single address for donations shall be used. These special addresses can be used to link users and in a secondary step whole communities to a real world entity.

6 Results and Discussion

6.1 Aggregating Users

As described in Section 5 we used an aggregated user-user network to find communities in Bitcoin. This first aggregation step, proved to be highly useful as it reduce the number of nodes in the network by almost 50%. Additionally we could use this domain specific knowledge to identify strong connections between address and reliably aggregate them without even having to use a network-analysis tool.

6.2 Community Detection

While the Girvan-Newman and Louvain algorithms were not useful in our experiments, the Clauset et al algorithm was able to separate out 1436 communities within the smaller dataset of a single day's worth of the blockchain. A majority of the communities were small, with 2-3 members. We believe that these represent invidiaul bitcoin users who were transferring currency or making payments that day. However, there were a number of large community networks, comprised of hundreds of thousands of individual users that were distinguished by the algorithm. The distribution of community sizes is detailed in Figure 3.

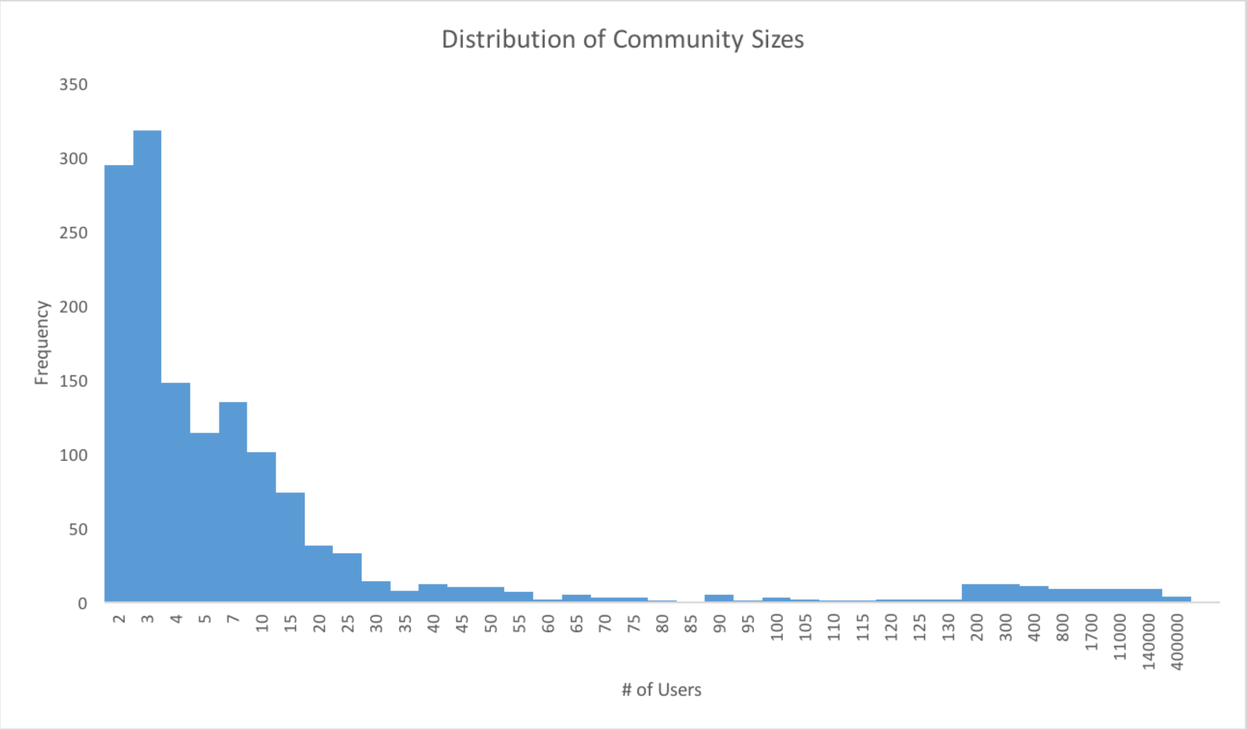
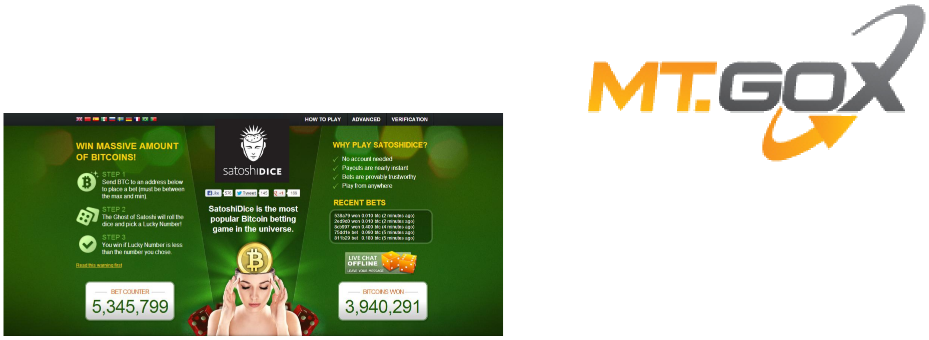


Figure 3: Histogram of Community Sizes

6.3 Attaching Identities

In the final step we analyzed the largest found communities and tried to find important identities associated with them. For one of the largest clusters containing 4,761 users and over 300,000 addresses we found that multiple addresses were owned my MtGox (Figure 4). MtGox was at the time the largest Bitcoin exchange so it seems reasonable that there was a large interaction with it. Interestingly none of the MtGox addresses from our database was associated with any other community which indicates the validity of our methedology.

The second community we identified had 2,186 users and over 120,000 addresses. We found that most of the address belonging to Satoshi Dice were associated with this cluster. Satoshi Dice is the largest Bitcoin gambling site (Figure 4). It is particularly interesting that we were able to link a community to this identity as gambling is considered to be an illegal activity in some countries.



WIN MASSIVE AMOUNT OF BITCOINS!

STEP 1
Send BTC to an address below to place a bet (must be between the max and min).

STEP 2
The Ghost of Satoshi will roll the dice and pick a Lucky Number!

STEP 3
You win if Lucky Number is less than the number you chose.

Read this warning first

BET COUNTER
5,345,799

SatoshiDice is the most popular Bitcoin betting game in the universe.

WHY PLAY SATOSHIDICE?

✓ No account needed
✓ Payouts are nearly instant
✓ Bets are provably trustworthy
✓ Play from anywhere

RECENT BETS

```
538a79 won 0.010 btc (2 minutes ago)
2ed9d0 won 0.010 btc (2 minutes ago)
8cb997 won 0.400 btc (4 minutes ago)
75dd1e bet  0.090 btc (5 minutes ago)
811b29 bet  0.180 btc (5 minutes ago)
```

BITCOINS WON
3,940,291

Figure 4: We identified a Satoshi Dice and a MtGox communtity

References

Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. Evaluating User Privacy in Bitcoin. In *Financial Cryptography*, 2013.

Michael Fleder, Michael S Kester, and Sudeep Pillai. Bitcoin transaction graph analysis. *arXiv preprint arXiv:1502.01657*, 2015.

S Nakamoto. Bitcoin: A peer-to-peer electionic cash system. Unpublished, 2008.

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.

Dorit Ron and Adi Shamir. Quantitative Analysis of the Full Bitcoin Transaction Graph. In *Financial Cryptography*, 2013.