Structure and Evolution of Bitcoin Transaction network

Lucas Liu (zliu2@stanford.edu); Aaron Brackett (abrack74@stanford.edu); Yixin Cai (yixincai@stanford.edu)

1. Introduction

Bitcoin is a new cryptocurrency that has slowly been on the rise. While it started more as a niche project, it has since evolved into a currency with actual value used to perform actual transactions in the real world. A user on Bitcoin has one or more public addresses, each with their own private address attached to them. In order to receive Bitcoins, the user gives out one of their public addresses. To send them, they sign a transaction with their private address. The transactions of all users are stored in the block chain, a chronologically ordered list of snapshots of the Bitcoin network taken every ten minutes. Because of this, Bitcoin gives us a unique opportunity to study financial systems in a way that was impossible before, as transactions were private affairs with no public records. We examined Bitcoin transactions throughout time to find the mechanisms of network growth, and a way to generate a synthetic model with similar properties. We pursued our project in three steps. First, we took detailed measurements at uniform instances of time. From there, we created different graphs to see if we could isolate out different components of the network that may be growing differently. Second, we tried to understand the evolutionary mechanism underlying the transaction network. Last, we developed a mathematical model based on our metrics from the first step and generated a simulation network using this model.

2. Related Work

The researchers who gathered our data examined several key statistics of the Bitcoin network to learn more about them from an economic standpoint. They discovered that the degree distribution follows the power law, and that the clustering coefficient is more than it would be in a randomly created network. They also studied the preferential attachment model of the network to discover that the rich do indeed get richer[1]. In our study of the bitcoin network, we replicate their results on the degree distribution and connectivity of the graph, and find a preferential attachment model we can use to simulate a new bitcoin graph. They also use some linear algebra to perform Principal Component Analysis on some matrices they constructed from the dataset[2], but we will not be focusing too much on this aspect of their research and will instead be exploring some ideas of network analysis and simulation methods used by different teams of researchers studying other networks.

We also looked at other papers examining other networks. One paper we looked at analyzed the network data from FLICKR, LinkedIn, Answers and Delicious, and proposed an entire model for evolution of social network. The evolution contains three steps - node arrival process, edge initiation process and edge destination selection process[3]. The models and methods discussed in this paper were relatively simple but powerful, so we decided to use them to analyze our data, studying patterns in node arrival and edge initiation in our Bitcoin network and coming up with our own model for these processes. Another paper detailed the process of studying an evolving network[4]. Using this paper as a guideline, we decided which measurements to take, what graphs to make, and what types of analysis we should do. In addition, they described a method to generate a model with the same properties, on which we based our process for simulating another bitcoin network. And our final paper investigated the dynamic characteristics of an evolving social network to uncover the underlying mechanism of evolution in a dynamic social network, breaking it down into three distinct steps. First, find measurements to understand how the network evolves. Second, extract useful characteristics from the metrics and construct a network model. Lastly, run a simulation on the network model and summarize its behavior and discover interesting properties[5]. This paper provided detailed guidance in understanding the evolution of dynamic networks, and we applied the same approach to analyzing the evolution of transaction network on Bitcoin.

3. Data Collection

We directly downloaded our data from ELTE BitCoin project[6].

There are two sets of data. The first dataset contains all BitCoin transactions starting from the first BitCoin transaction on 1/3/2009 to 12/28/2013. In this dataset we are using the following files:

1. txtime.txt, which includes all transaction IDs and timestamps.
2. txedge.txt, which tells us the sending address and receiving address for each transaction.
3. degree.txt, which contains the node degree of all nodes

There is another set of data which contains all transactions before 10/19/2014. We decided to not use this data set because it does not contain timestamps and there is no way for us to split it into smaller chunks. It is impossible for us to simulate all transactions from 12/28/2013 to 10/19/2014 and fit the graph in memory.

For the original dataset, due to computation limitation, we decided to only consider transactions between 5/28/2013 to 12/28/2013 for simulation. We initialize the base graph using transactions from 5/28/2013 to 11/28/2013, and transactions between 11/28/2013 to 12/28/2013 as the test set to evaluate our simulation performance against. The data prior to 5/28/2013 are used in network analysis, but not in model simulation.

4. Problem Definition and Network Statistics

We define the problem as the following: The BitCoin transaction network is a directed graph because each transaction has senders and recipients. Nodes are defined as address IDs, and edges are defined as the transaction between two addresses.

Starting from an empty graph, when a transaction arrives there could be multiple senders and recipients involved. We connect each sender to each recipient. The edges are the cross product between senders and recipients. For sender nodes and recipient nodes in each transaction, “new nodes” are defined as nodes that are not in the existing graph; “old nodes” are nodes that already exist in the graph. After each transaction, we add all nodes and edges to the graph and repeat the process until there are no more transactions.

The graph of the whole dataset has 129178908 edges for 29771436 transactions. 59.1% of the edges were old nodes sending to old nodes. 40.4% of the edges were old nodes sending to new nodes. 0.339% of the edges were new nodes sending to old nodes, and 0.132% of the edges were new nodes sending to new nodes.

Figure 1 and Figure 2 show in-degree and out-degree distribution in BitCoin network on a log-log scale. It is very clear that in-degree strictly follows the power law. The out-degree distribution has a peak on degree 2, which suggests that the out-degree is most likely to be 2 rather than 1. This pattern will later affect how we model the number of recipients in each transaction.

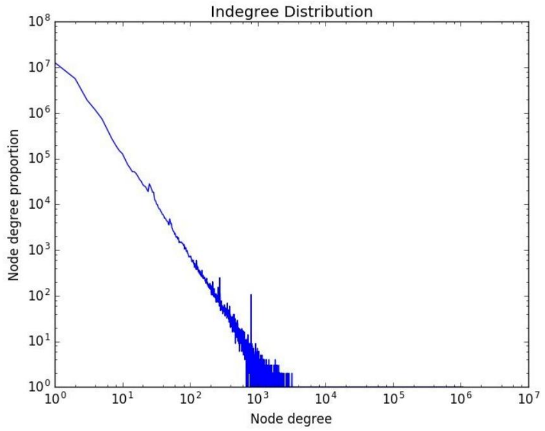

Figure 1

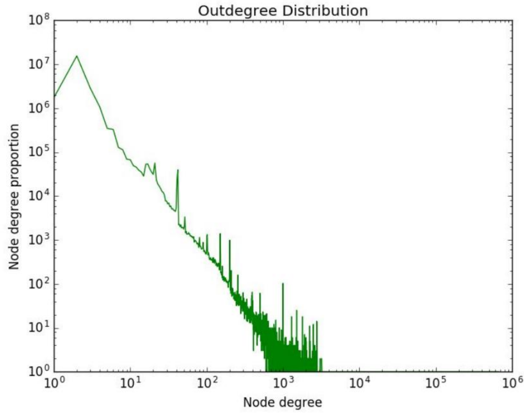

Figure 2

5. Network Analysis, Findings and Model Construction

5.1 Transaction generation

The first thing we want to measure is how fast the network grows. For the BitCoin network, this can be measured by how many transactions occur in a month. The reason we use a monthly graph is because it is the largest scale that we can analyze - the yearly graph is way too large and the weekly graph has too many fluctuations. Figure 3 is a monthly graph for the number of transactions where the y-axis uses a log scale. As we can see, this graph roughly follows a line, and we can make the assumption that the network grows exponentially.

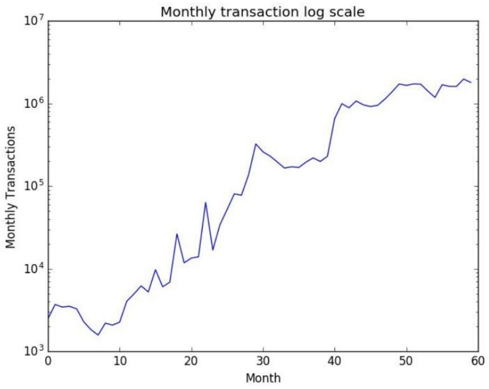

Figure 3

As a result, we fit a linear model $y = kx + b$, where y is the logarithm of number of monthly transactions, and x is the number of months passed since the first BitCoin transaction. After further investigation, we decided to fit the model only using the data in the last 7 months because the network grows at different rates throughout time. The actual number of transactions is 1975945, and our predicted number of transactions is 1948456. There is a small difference of 1.4%.

5.2 Determining number of senders

After determining the number of transactions, we examined how many senders and recipients are involved in each transaction. Figure 4 shows the distribution for sender counts in one transaction, and Figure 5 shows the distribution of recipient counts in each transaction. We decided to first select the number of senders for a transaction because a straight line is easier to model.

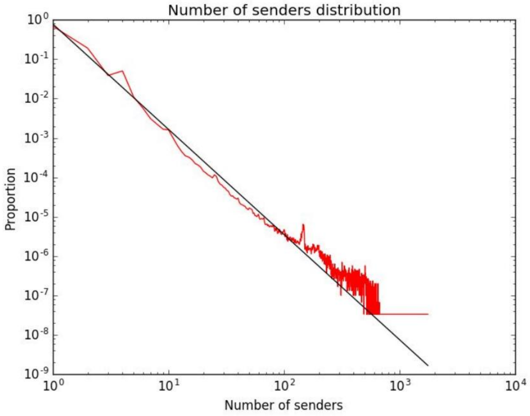

Figure 4

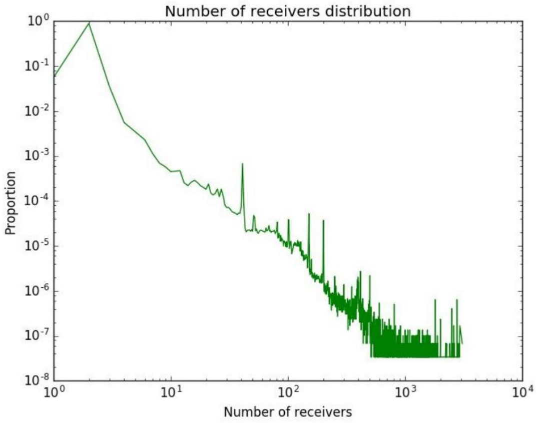

Figure 5

We model the distribution using the power law; the log-probability of select a sender number is linear with regard to logarithm value of the number.

The equation is $log(P(X=x)) = k * log(x) + b$

We used tail pruning, cumulative distribution and expectation maximization to find the values of k and b. The fitting result is k = -2.6711961619 and b = -0.533973482704, which is shown as the black line in Figure 4.

5.3 Determining number of recipients given sender count

After we have determined the number of senders, we need to figure out the number of recipients. The first question we want to ask is - is the distribution for number of recipients dependent or independent on number of senders? We have generated three graphs (Figures 6-8) for recipient number distribution given 1, 2 and 3 senders, and all of them seem to follow a similar pattern. Interestingly, this pattern is similar to node out-degree distribution and number of recipients distribution. As a result, we can be quite confident that the number of recipient distribution will be independent from the number of senders. Also, it follows the same pattern as Figure 5, just with different maximum values.

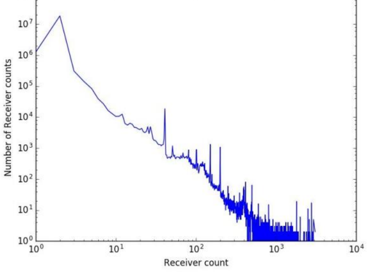

Figure 6

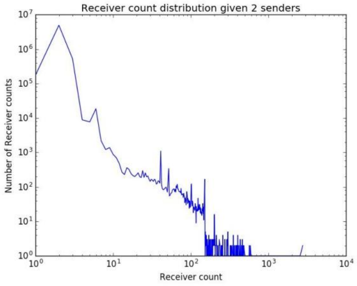

Figure 7

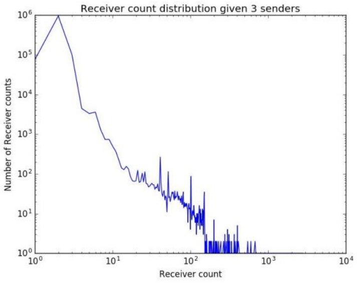

Figure 8

Another thing that we have realized is that the minimum and maximum number of recipients changes as the number of senders change. We can see from the graphs above that the slopes are getting steeper as number of senders increase. We have created two graphs (Figures 9 and 10) for the range that number of recipients can take. (Note that we have found and fixed a bug in our code, so the graphs look different from the ones in milestone.)

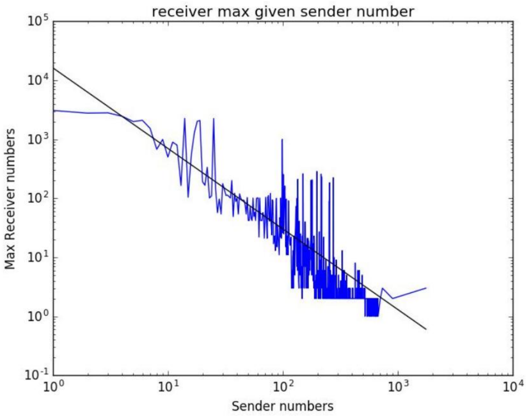

Figure 9

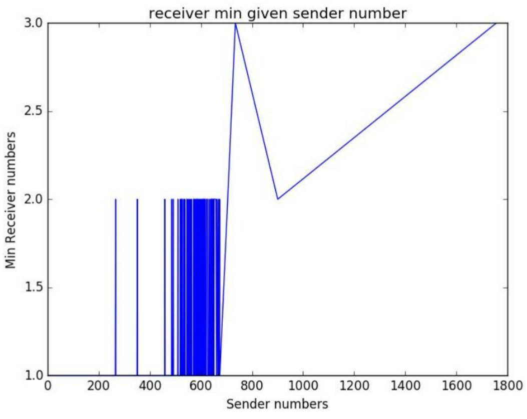

Figure 10

As we can see, the maximum number distribution (log-log scale) follows a straight line. But the minimal number is always close to 1. We have fit a linear model for maximum number $log(P(X = x)) = k * log(x) + b$ where the coefficients are k = -1.36426904827, b = 9.69072177648.

As a result, our model would be: given a number of sender, we generate the max number of recipients based on power law, and set the min number to be 1. The number of recipients are selected based on out-degree distribution in Figure 5 normalized by maximum number.

5.4 Determine number of new senders and new recipients

Given the number of senders and recipients, we want to figure out what is the proportion of new nodes. However, given number of senders and recipients, the probability of new senders and new recipients seems to be very noisy. Some example data points are presented in Table 1.

Sender recipient pair	New sender and new recipient distribution
(3, 527)	(0, 21): 1
(7, 47)	(0, 1): 2, (0, 4): 1
(72, 42)	(0, 42): 1
(641, 2)	(0, 1): 2
(5, 1495)	(0, 133): 1

Table 1

It seems like there is no good model for such joint distribution, thus we have to make two assumptions. The first is that the number of new senders and the number of new recipients are independent. The second is that each new node are independent from each other. Then we started to look at the distribution of sender count and recipient counts given a new sender node or a new recipient node. Figures 11 and 12 show the distribution of sender count and recipient count for new sender nodes. Figures 13 and 14 show the distribution of sender count and recipient count for new recipient nodes.

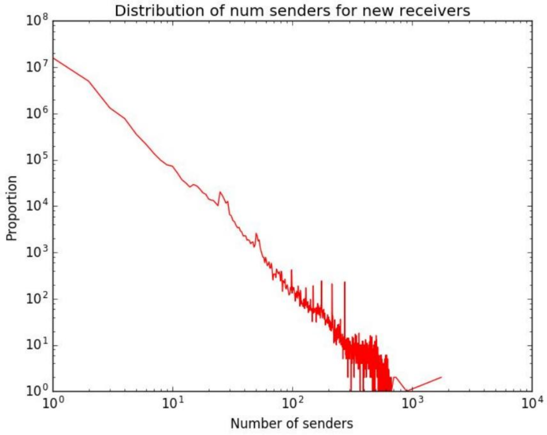

Figure 11

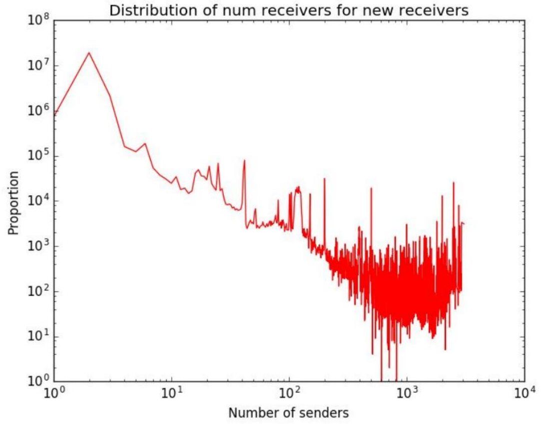

Figure 12

Interestingly, given a new sender, the distribution of sender count and recipient count is very similar to Figure 4 and 5, which is the number of senders and number of recipient distribution. As a result, we come to a conclusion that the probability of a sender to be new is independent from sender and recipient count.

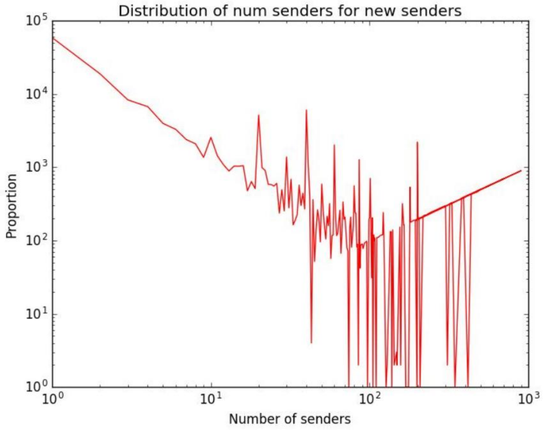

Figure 13

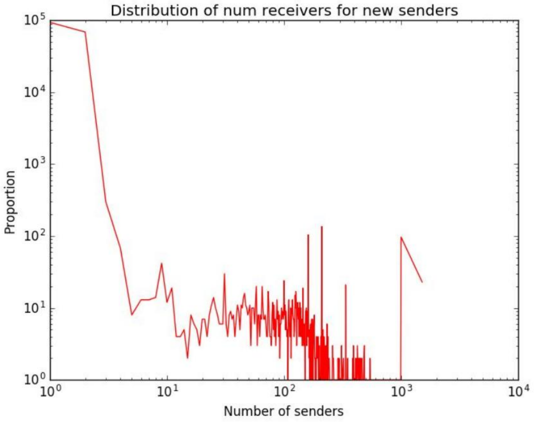

Figure 14

On the other hand, given new recipient, the sender count distribution and recipient count distribution are very different. It is more likely for new recipient to show up when the number of senders is either very large or very small. The distribution seems to follow a downward line and then an upward line. However, for our model, we decide to still use uniform distribution. One reason is that the distribution is quite noisy in the middle and two lines may not be good enough. The second reason is that since most transactions only have a small number of senders, the uniform distribution is good enough as long as it applies to most transactions with few nodes involved. Finally, this provides us with a simple model that would not be hard to implement and test.

As a result, by learning from the existing graph, we decided to use 0.00316093237904 as the probability of a sender to be new and 0.336722278848 as the probability for recipient node to be new.

5.5 Preferential attachment

In this section, we will study how new edges are introduced into the network. Similar to the experiment in section 4.1 in Leskovec et al[3], we want to determine, when a new node joins the network, whether the destination node of a new edge is chosen proportional to the destination's degree. We decide to plot $p_e(d)$ as the probability that a new edge chooses a destination node with degree d. The formula in Leskovec et al[3] is given as:

$$p_e(d) = \frac{\sum_t [e_t=(u,v) \wedge d_{t-1}(v)=d\,]}{\sum_t |\{u{:}d_{t-1}(u)=d\}|}$$

We divide our data into monthly graphs with $t \in [0,6]$ where $t=0$ corresponds to 05/28/2013. We then compute the degree distribution as displayed in Figure 15. We can clearly observe a linear relationship between $p_e(d)$ and d. Therefore, we can conclude that the Bitcoin network follows the preferential attachment model.

Given the Bitcoin network is a directed graph, similarly, we also need to study the relationship between source node of a new edge and its degree. We decide to compute the degree distribution $p_e(d)'$ using the same formula as above but with edge direction flipped:

$$p_e(d)' = \frac{\sum_t [e_t=(u,v) \wedge d_{t-1}(u)=d\,]}{\sum_t |\{u{:}d_{t-1}(u)=d\}|}$$

The distribution in Figure 16, still exhibits a weakly linear relationship between $p_e(d)'$ and d.

In summary, both source and destination nodes are more likely to get attached to an edge with higher degrees in the Bitcoin transaction network. Hence, in the simulation, specifically step 5, we decide to choose the source and destination nodes, in our case senders and recipients, proportional to their degrees.

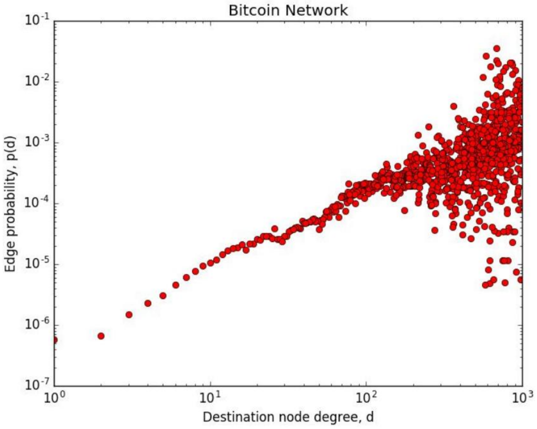

Figure 15

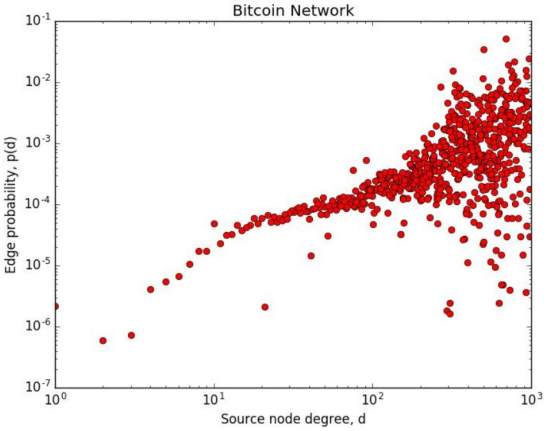

Figure 16

Using an approach similar to the one in 5.2, we fit the equation $log(P(X=d)) = k * log(x) + b$, where $k=0.95577536$, $b=-13.39918393$ for Figure 15. In Figure 16, $k=0.96687774$, $b=-13.16216966$.

6. Simulation Algorithm

Here is the algorithm we use to simulate BitCoin network between 11/28/2013 and 12/28/2013:

1. We generate the total number of transaction based on the exponential model.
2. For each transaction, we randomly select the number of senders according to power law.
3. Given number of senders, we determine max number of recipient from our power law model and set min number as 1. We normalize the number of recipient distribution based on the max number, then select the number of recipients randomly according to the distribution.
4. For each sender node, it has 0.00316093237904 probability to be a new node. For each recipient, it has 0.00316093237904 probability to be a new node.
5. We choose existing senders and existing recipients proportional to node degree. Ideally we should have a p where with probability p the node is chosen uniformly random and with (1-p) node is chosen according to node degree, but we do not have time to tune with different p values due to the large number of transactions.
6. After the sender nodes and recipients nodes are selected, we add new nodes to the existing graph, and add edges between each pair of sender and recipient..
7. Repeat for the number of transaction times.

7. Simulation Results and Analysis

Although we wanted to simulate multiple times and take the average, we were unable to do so because the number of transactions is very large and each simulation would take hours to complete. Eventually we were only able to simulate once for a 1 month period instead of 10 times for 10 months.

7.1 Number of transactions

As mentioned in 5.1, the actual number of transaction is 1975945, and our predicted number of transaction is 1948456. We have achieved a very close prediction, which only differs from the real number by 1.4%.

7.2 Nodes, edges and network diameter

	Real graph	Simulated graph
Nodes	3031288	3441156
Edges	10986324	7655108
Diameter	25.5428182905	8.24465871597

Table 2

Table 2 shows the difference between simulated value and the real values. We can see that the number of nodes are close, but there is a huge difference between number of edges and network diameter. Our simulated network is more connected and more condense. The reason is that all old nodes are selected based on node degree. As a result, there are more repeated edges in our simulation than the real network. If we have more time, we will try with different p values for uniform node selection, and it will definitely increase number of edges and increase network diameter.

7.3 in-degree and out-degree distribution

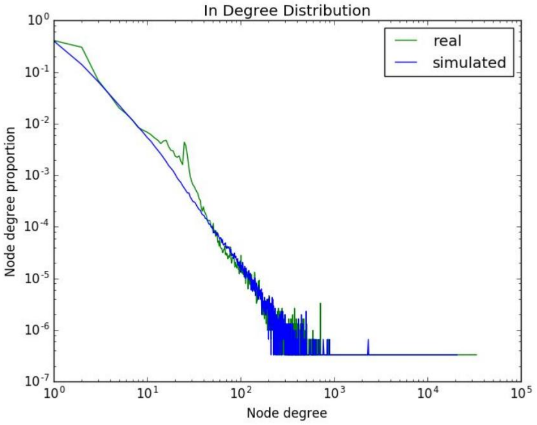

Figure 17

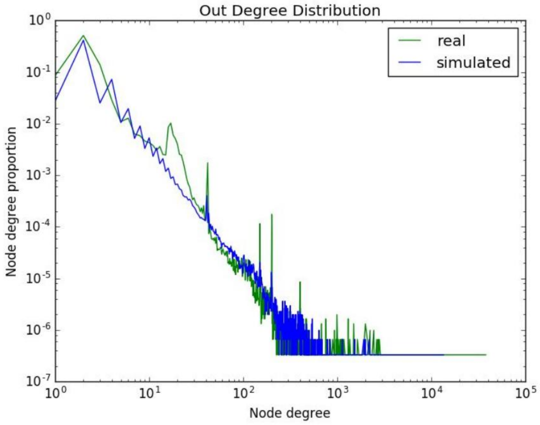

Figure 18

Figures 17 and 18 shows in-degree distribution and out-degree distribution for real network and our simulated network. It is clear that our simulation does represent the real world degree distribution.

7.4 Number of sender and number of recipient distribution

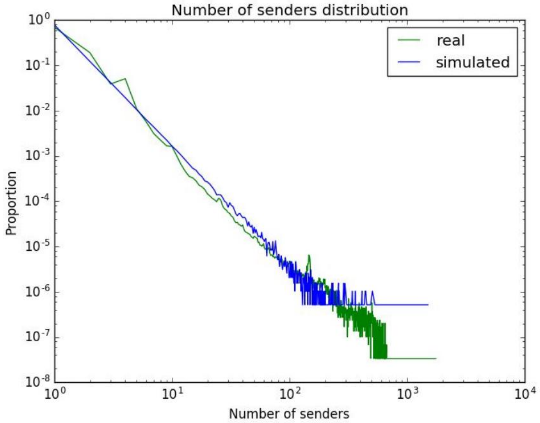

Figure 19

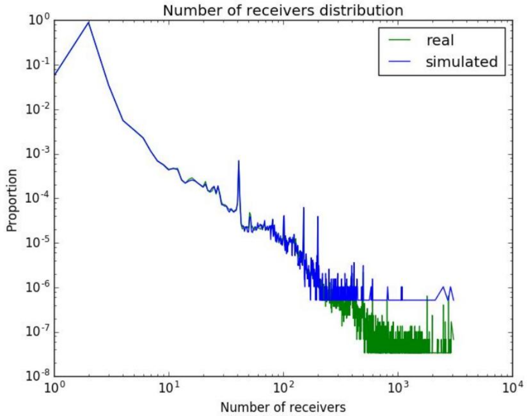

Figure 20

Figures 19 and 20 shows number of sender distribution and number of receiver distribution for the real network with all transactions and our simulated model. It is not surprising that our model successfully captures characteristics in both graphs. Note that the real graph (green) has lower value for minimum proportion because the total number of transactions is much larger than the number of transactions in our simulation.

8. Conclusion and Future Work

In conclusion, our model has successfully captured how a transaction affects the growth of the BitCoin network. We have made close prediction on the amount of growth, number of transactions, senders and recipients of transactions and how new nodes appear for BitCoin network. Even though we made many independence assumptions in our model, we are still able to achieve a good result for number of nodes, in-degree distribution and out-degree distribution.

The aspects that our model does not take into consideration are mostly the internal structure of the BitCoin network. We have not modeled the V-shaped sender number distribution given new receivers, and the probability *p* that leads to uniform distribution in node selection. Other things that we can explore in the future include seasonal fluctuation, community structure, node BitCoin balance and edge occurrence counts.

9. References

1. D. Kondor, M. Pósfai, I. Csabai and G. Vattay. Do the Rich Get Richer? An Empirical Analysis of the Bitcoin Transaction Network. *PLOS One,* 2014.
2. D. Kondor, I. Csabai, J. Szüle, M. Pósfai and G. Vattay. Inferring the interplay between network structure and market effects in Bitcoin. *New Journal of Physics,* 2014.
3. J. Leskovec, L. Backstrom, R. Kumar, A. Tomkins. Microscopic Evolution of Social Networks. In Proc. KDD 2008.
4. R. Kumar, J. Novak, A. Tomkins. Structure and evolution of online social networks. In Proc. KDD, 2006.
5. A.L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, T. Vicsek. Evolution of the social network of scientific collaborations. Physica A: Statistical, 2002.
6. http://www.vo.elte.hu/bitcoin/downloads.htm

10. Contribution

Yixin Cai: Analysis for network growth, sender and recipient distribution and new node appearance. Algorithm design, simulation and result analysis.

Lucas Liu: Data preprocessing, preferential attachment implementation and analysis.

Aaron Brackett: Introduction, Related Work, References, Poster, Presentation, Proofreading, Organization