Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Exploring Blockchain Peer-to-Peer Networks

Bachelor's Thesis

Alain Gautschi

alainga@ethz.ch

Distributed Computing Group
Computer Engineering and Networks Laboratory
ETH Zürich

Supervisors:
Zeta Avarikioti, Ye Wang
Prof. Dr. Roger Wattenhofer

March 14, 2021

Acknowledgements

I thank my supervisors Zeta and Ye for their help and fresh ideas. I am grateful for my family's endless support.

Abstract

The rise of cryptocurrencies like Bitcoin and Ethereum attracts a lot of attention, also from attackers. In order to secure blockchain systems, one first needs to understand their networks. And this is what we try to do in this thesis. We compare degrees, clustering coefficients, shortest paths and closeness centrality of simulated Bitcoin and Ethereum networks. Furthermore, we build a theoretical model to calculate these properties in theory and compare them to the results of simulations. We find that the centrality of a node is dependant on the time it was added and that Ethereum-like networks have a linear construction process, which leads to high clustering.

Contents

Chapter 1

Introduction

1.1 Motivation

Bitcoin is considered to be one of the most valuable electronic currencies currently. As of March 2021, the price of each bitcoin is as high as $50,000. Driven by such interests, tens of thousands of people are incentivized to attack the Bitcoin network every day, which has caused great concern for the security of the Bitcoin network. Other blockchain systems also face the same security threat. Research on blockchain has mainly focused on Layer 1 and Layer 2 technologies, i.e., on designing consensus protocols and off-chain scaling solutions. However, for these layers to work properly, the network layer (Layer 0) has to maintain certain properties and guarantees, since all the data between the nodes of a blockchain system are exchanged via peer-to-peer communication.

1.2 Related Work

We were able to build upon the work of [1], which provided a theoretical model for the degree distribution of unbounded networks and the basis of the simulations. A great overview on the topic of networks can be found in the work of Albert et al. [2]. Marcus et al. described the Ethereum peer-to-peer network [3] and demonstrated attacks on Ethereum which together with [4], [5], and [6] emphasized the importance of the topology of blockchain networks. The recursive paths approach in [7] delivered analytical results for the shortest path distribution in random graphs, which we adjust to specific node pairs in growing peer-to-peer networks. Path lengths in configuration model networks were discussed in [8], which showed not to be of use for this thesis, as it does not account for the construction mechanism of a network.

1.3 Contribution

In this thesis, we want to understand what properties these networks have, what mechanism constitute these properties and how they may affect the security and fairness of a blockchain peer-to-peer network. We simulate Bitcoin and Ethereum networks and compare their properties. Furthermore, we build a theoretical model which allows us to compute the degree distribution, clustering coefficients, shortest paths and closeness centrality of such networks. We show the consequences of parameter choice on the network topology. Moreover, we exhibit that the connectivity of nodes in Ethereum-like networks is strongly dependant on the time when they were added.

Chapter 2

Background & Simulation Setup

In peer-to-peer networks, computers directly connect and communicate with each other without the need of a central instance. This enables powerful technologies like blockchains. An outgoing connection is a connection which a node initiates itself, whereas an incoming connection was initiated by another node. The inbound and outbound distribution specifies the choice of bounds on the number of incoming and outgoing connections. When the connection was successful, we speak of peers or neighbours. We describe the default setting in the following.

2.1 Bitcoin's connection strategy

A Bitcoin node tries to reach 8 outgoing connections, which it chooses from its address table at random, and allows for a total of 125 connections, limiting the incoming connections at 117 [1].

2.2 Ethereum's connection strategy

An Ethereum node is allowed to have up to 13 outgoing connections and accepts by default at most 17 incoming connections [9]. In Ethereum, the mechanism of choosing which nodes to connect to, differs quite a bit from Bitcoin's.

In a first step, an Ethereum node chooses 6 random addresses from all addresses known and additionally fills the *lookup buffer* with up to 16 addresses.

It then tries in a second step to connect iteratively to those 22 nodes until it reaches a max of 13 outgoing connections. So the number of outgoing connections a node can initiate is dependant first, on how many addresses it can put into the *lookup buffer*, which we can safely assume is always 16 for a sufficient large network, and second, how many of those nodes accept the connection.

If the *lookup buffer* is smaller than (13 - #outgoing connections), it starts a new discovery task which fills up the *lookup buffer* with new addresses [3]. The

more strictly limited Ethereum nodes, in terms of connection bounds, leads to many full nodes which can not accept any more connections. This makes it harder for new nodes to find other nodes which still accept incoming connections. As we will see later, this strongly affects the topology of the Ethereum network.

2.3 Simulation setup

In the following chapters, we refer to different models and simulations. First, we run full simulations called *full sim bitcoin* and *full sim ethereum*.

The notions *theoretical bitcoin* and *theoretical ethereum* represent the respective theoretical models descriped in Chapter 4.

As the theoretical model does not suffice for the calculations of centrality, we construct random networks called *random bitcoin* and *random ethereum*. In the random networks, we iteratively add one node, choose 8, respectively 13 random nodes from the network and connect them, if they do not already have 125, respectively 25 connections.

The properties of those random networks are then compared with results from the simulations *lite sim bitcoin* and *lite sim ethereum*. These lite simulations are limited to the functionality of the initial connection establishing in order to deliver comparable results. So nodes only have one chance to connect with peers, which is at the time when they are added to the network.

All results are from networks of 3000 nodes. We run 10 simulations for each strategy and show the average results. The code used in this thesis was taken from [1] and modified and completed. The repository can be found here [10].

Chapter 3

Simulations

We thoroughly test the simulation of [1] and apply improvements and changes where necessary. At each timestep in the simulation, some new nodes are added and all nodes process their connection requests. The *id*s range from 1 to 3000, where the node with *id* 1 is the oldest in the network.

Additionally, we have that some nodes go offline and online again. Biryukov et. al. [11] showed that a node has a 15% chance to go offline after 4 hours. To model this behaviour, a random amount of nodes between 0% and 3% of the network disconnect and reconnect again with a 90% chance in the coming timesteps. The result, as we will see later, is a more mixed up network, as nodes do not connect to their previous peers after a loss of connection. This means that Ethereum nodes have peers from the full range of *id*s. With this we account for reboots of nodes in the real world networks due to updates or power outages. The loss of a peer's internet connection may also pose a cause of connection teardown.

The degree distribution, clustering coefficients per node, shortest paths between every node pair and closeness centrality per node of the simulated Bitcoin and Ethereum networks are evaluated.

Note that a lower node *id* corresponds to an earlier introduction to the network of said node. Hence most of the plots reveal also the dependency on time. Outliers may exist, if a node gets dis- and reconnected towards the end of the simulation and does not have enough time to fully connect again.

3.1 Degree distribution

Figure 3.1 shows the comparison of the degree distribution between fully simulated Ethereum and Bitcoin networks. Bitcoins degree distribution mirrors one of a scale-free network and shows a power law distribution, where the oldest nodes get the most incoming connections [2]. On the contrary, Ethereum prohibits this behaviour by having a rather small upper bound on the total amount of connections. Leading to a much smaller variance in the degree where the majority of nodes reach a full degree of 25. The drop of degrees of the youngest nodes, i.e.,

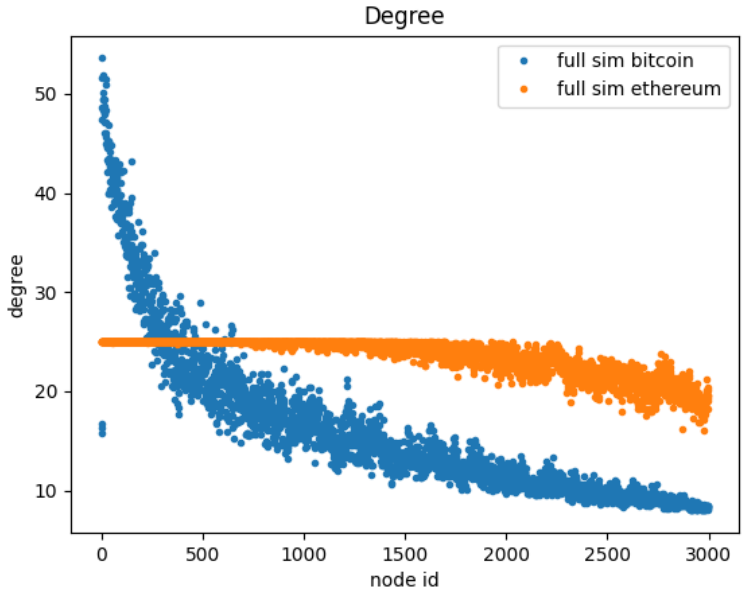


Figure 3.1: Comparison of degree distribution

the nodes added last, is due to the lack of even younger nodes providing incoming connections.

3.2 Clustering

The clustering coefficient measures how well connected the neighbours of a node are with themselves. The local clustering coefficient for node i in an undirected, unweighted graph is

$$c_i = \frac{2T(i)}{deg(i)(deg(i)-1)}$$

where $T(i)$ denotes the number of triangles through node i [12].

Figures 3.2 and 3.3 illustrate the the local clustering coefficient of fully simulated Ethereum and Bitcoin networks. One immediately detects that the clustering coefficient of Ethereum nodes have a much higher average and variance than Bitcoin nodes. We believe that this is due to the linear construction of the network. Meaning that, if one thinks of the network on a plane, it would grow in one direction, so to speak, as a new node can only connect to nodes which just have been added to the network too, because older nodes are already full.

Furthermore we notice that there are Bitcoin nodes with a clustering coefficient of zero. With 3000 nodes and an average degree of 16, it is expected to have some nodes with non-connected neighbours.

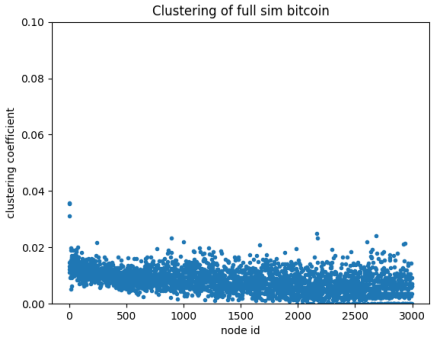


Figure 3.2: Clustering coefficient in Bitcoin.

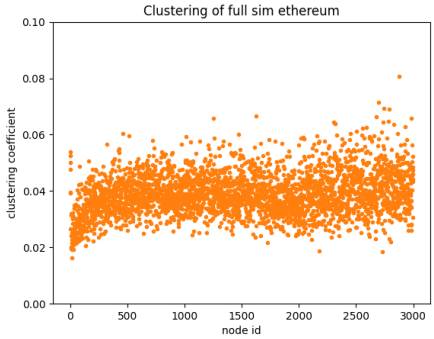


Figure 3.3: Clustering coefficient in Ethereum.

3.3 Shortest paths

Figure 3.4 depicts the comparison of shortest paths between fully simulated Ethereum and Bitcoin networks. The average shortest path in the Ethereum network is shorter than in the Bitcoin network. We believe that this is due to the higher average degree of Ethereum nodes. Both networks have almost no shortest path with a length of 5 or longer, similar to "small-world networks" in which according to [12], the average distance between any two nodes grows logarithmically with the number of nodes.

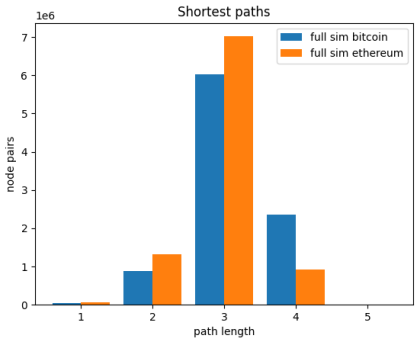


Figure 3.4: Shortest path lengths.

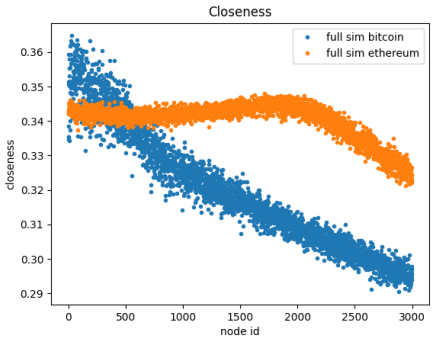


Figure 3.5: Closeness centrality.

3.4 Closeness

Closeness centrality gives us a measurement on how close a node is to all other nodes on average. Formally, closeness centrality of a node i is the reciprocal of the sum of the shortest path distances from i to all $N - 1$ other nodes. Since

the sum of distances depends on the number of nodes in the graph, closeness is normalized by multiplying with $N-1$ [13, 14].

$$C(i) = \frac{N-1}{\sum_{j=1}^{n-1} d(i,j)}$$

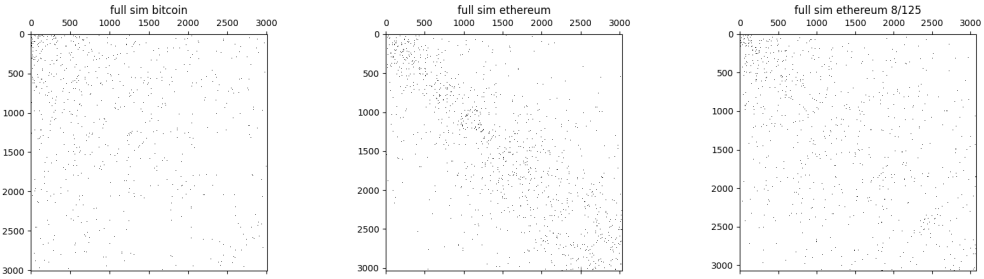
where $d(i,j)$ is the shortest-path distance between i and j. Note that higher values of closeness indicate higher centrality.

Figure 3.5 demonstrates the comparison of closeness centrality between fully simulated Ethereum and Bitcoin networks. One observes that the closeness centrality of both Bitcoin and Ethereum look similar to their degree distribution, having the same drop in time. We learn from this that the oldest nodes are more central in the network than younger nodes, especially in Bitcoin. Also, Ethereum has an overall higher average closeness, which we attribute to the higher average degree. We believe this is due to the correlation between degree and closeness. The correlation coefficient of degree and closeness is 0.4 according to [15].

3.5 Connection distribution

In order to get a picture of the connections in the networks, we use matrix plots. Figure 3.6 beautifully shows every connection in the networks as a dot. The corresponding x- and y-coordinates of a dot correspond to the node *id*s of the connected peers. This clearly demonstrates that nodes in the Ethereum network connect preferably with nodes which have been added shortly before or after themselves.
Figure 3.6c is the result of an Ethereum simulation, but with the connection upper bounds of Bitcoin, i.e., 8 for outgoing and 125 for total connections. We still can observe a slight grouping of dots along the diagonal. This implies that not only the connection bounds, but also the peer choosing mechanism of Ethereum is responsible for this linear construction of the network.

In order to confirm this with a quantitative measurement, we compute the average difference between a node's *id* to the *id*s of its peers for all nodes in the network. Figure 3.7 shows the results of typical Bitcoin and Ethereum networks and of a network with Ethereum's connection mechanism but with Bitcoin's connection bounds. The x-axis depicts each node's id and the y-axis is the average distance to the peer *id*'s. Indeed, not only the difference between Ethereum and Bitcoin is significant, but also the difference between Bitcoin and Ethereum with Bitcoin's upper bounds.



(a) Bitcoin. (b) Ethereum. (c) Ethereum 8/125.

Figure 3.6: Matrix plots.

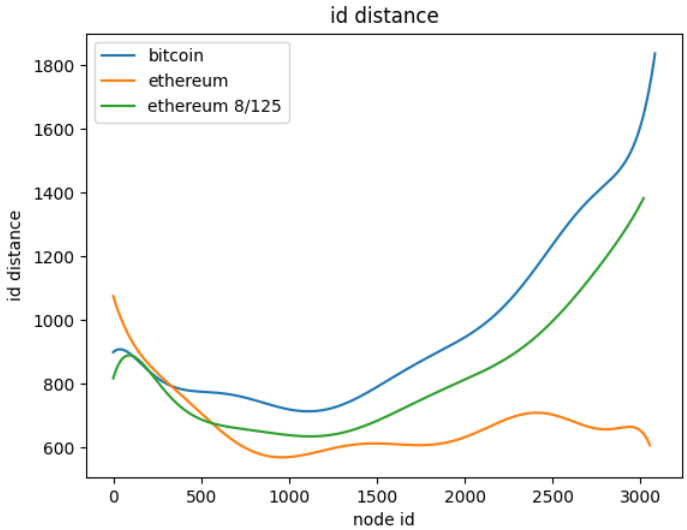


Figure 3.7: Distances of id's

Chapter 4

Theoretical Model

In this chapter, we build a theoretical model for Bitcoin- and Ethereum-like networks in an attempt to understand the outcomes of different bounds on incoming and outgoing connections. When compared to the simulations, this model also sheds light on the effects of both connection mechanisms.

The model abstracts the behaviour of nodes when choosing peers for outgoing connections, but lack extended functionality like retrying to establish outgoing connections to other nodes if some of the initially chosen nodes refuse the connection.

We calculate the degrees, clustering coefficients, shortest paths and closeness centrality of up to 3000 nodes in total and compare it to the results of the lite simulations.

4.1 Preliminaries

The parameters of the model are m and C, which are set to 8 and 125 respectively to 13 and 25 to model the upper bounds on outgoing and total number of connections in Bitcoin and Ethereum.

The network initially has m fully connected nodes and grows by iteratively adding one node at each time step. Node i is added at time i. After a node is added to the network, it tries to establish m outgoing connections. We abstract the process of choosing peers to connect to and treat it to be completely at random. We further assume that a node has perfect information about all other nodes, i.e., their IP-address, which it needs to know in order to establish a connection.

A node is allowed to have up to C total connections. Hence, each node accepts at least $C - m$ incoming connections. $N(t) = t - 1$, denotes the amount of nodes at time t when the node with $id = t$ joins the network, while $deg(i, t)$ is node i's degree at time t.

The function $f(t)$ expresses the amount of full nodes at time t, i.e., the num-

ber of nodes which do not accept anymore incoming connections. By the nature of the construction process, we have that a node i will not accept new incoming connections at time t if $i \leq f(t)$. As we will see, this only has an effect with Ethereum parameters, as in a network of reasonable size with the Bitcoin parameters, no node reaches the upper bound of allowed connections.

f(t) can be computed as follows.

$$f(t) = \sum_{i=1}^{t-1} D_i^{t-1} \text{ , where } D_i^{t-1} = \begin{cases} 1 & \text{if } deg(i, t-1) \geq C \\ 0 & \text{else} \end{cases}$$

In the following we state everything explicitly for Ethereum, but plugging in $f(t) = 0$ gives the results for Bitcoin.

4.2 Degree

In Ethereum, the probability that a random node is not full and hence accepts a new incoming connection at time t is $\frac{N(t)-f(t)}{N(t)}$. Therefore, the number of connections grows by $m\frac{N(t)-f(t)}{N(t)}$ with each new node. These new connections are distributed among all nodes in the network, which still accept incoming connections. Thus, the changing rate of the degree of a node i at time t is

$$\frac{\partial deg(i,t)}{\partial t} = \frac{m\frac{N(t)-f(t)}{N(t)}}{N(t)-f(t)} = \frac{m}{N(t)}$$

if it is not full yet.

$$\frac{\partial deg(i,t)}{\partial t} = \frac{m}{N(t)} \iff \int \partial deg(i,t) = \int \frac{m}{N(t)} \partial t \iff deg(i,t) = m\ln(N(t)) + A$$

Using $deg(i,i) = m\frac{N(i)-f(i)}{N(i)}$ we get $A = m\left(\frac{N(i)-f(i)}{N(i)} - \ln(N(i))\right)$ and hence

$$\begin{aligned} deg(i,t) &= m\ln(N(t)) + m\left(\frac{N(i)-f(i)}{N(i)}\right) - \ln(N(i)) \\ &= m\left(\ln(N(t)) - \ln(N(i)) + \frac{N(i)-f(i)}{N(i)}\right) \\ &= m\left(\ln\left(\frac{N(t)}{N(i)}\right) + \frac{N(i)-f(i)}{N(i)}\right) \end{aligned}$$

Adding that nodes do not accept more connections if they are full we get

$$deg(i,t) = \min\left\{m\left(\ln\left(\frac{N(t)}{N(i)}\right) + \frac{N(i)-f(i)}{N(i)}\right), C\right\} \tag{4.1}$$

The plot of the degrees in the theoretical Bitcoin and Ethereum networks compared with their lite simulations with 3000 nodes can be seen in Figure 4.1.

We see that no node is full in the Bitcoin network, whereas in the Ethereum network about one third of all nodes get full.

Note that for a failing connection, we only considered if a node is already full or not. But In reality, there are many more reasons for a connection to fail, e.g., different software versions, blacklists, etc.

The plot shows as expected, that the earlier a node is introduced to the network, and hence the lower the *id* is, the higher the probability to reach a large degree. The reason for this is that older nodes receive many connection requests from younger nodes. The difference between the lite sim and theoretical bitcoin at the beginning can be explained with the fact that we have DNS nodes in the lite sim, which all other nodes try to connect to.

4.3 Clustering

In this section, we compute the expected local clustering coefficient for each node. From Section 4.2 we already have the degrees. What remains is the expected value of $T(i)$, the number of triangles through node i. We use an indicator variable $X_{i,j,k}$, which is 1 iff the nodes i, j, k form a triangle, and sum over all possible constellations. Then $\mathbb{E}[T(i)]$ is

$$\mathbb{E}[T(i)] = \mathbb{E}\left[\sum_{j\neq i}^{1\leq j\leq N}\sum_{k\neq i}^{j<k\leq N} X_{i,j,k}\right]$$
$$= \sum_{j\neq i}^{1\leq j\leq N}\sum_{k\neq i}^{j<k\leq N} \mathbb{E}[X_{i,j,k}] = \sum_{j\neq i}^{1\leq j\leq N}\sum_{k\neq i}^{j<k\leq N} Pr[X_{i,j,k}=1]$$

from the linearity of expectation.
But what is $Pr[X_{i,j,k}=1]$? Let's say we have nodes i, j and k where $i<j<k$ w.l.o.g. Then $Pr[X_{i,j,k}=1]$ is the probability that node j chose to make an outgoing connection to i and node k chose to make outgoing connections to both i and j. Those two events are independent. Note that nodes are being iteratively added to the network and initially choose m random peers they connect to. The probability that node j makes an outgoing connection to i and succeeds is

$$\frac{m}{N(j)}, \text{ if } i > f(k), \text{ else } 0$$

And similarly for k to make outgoing connections to i and j is

$$\frac{m(m-1)}{N(k)(N(k)-1)}, \text{ if } i,j > f(k), \text{ else } 0$$

In general, there are three cases: $i < j < k$, $j < i < k$ and $j < k < i$. With this, we get

$$\begin{aligned} E[T(i)] = &\sum_{j=i+1}^{N} \sum_{k=j+1}^{N} \frac{m}{N(j)} \frac{m(m-1)}{N(k)(N(k)-1)} \\ &+ \sum_{j=1}^{i-1} \sum_{k=i+1}^{N} \frac{m}{N(i)} \frac{m(m-1)}{N(k)(N(k)-1)} \\ &+ \sum_{j=1}^{i-2} \sum_{k=j+1}^{i-1} \frac{m}{N(k)} \frac{m(m-1)}{N(i)(N(i)-1)} \end{aligned}$$

Figure 4.2 shows a plot of the clustering coefficients of the theoretical Bitcoin and Ethereum networks compared with their lite simulations with 3000 nodes. Both theoretical and lite simulation Ethereum have very high clustering coefficients in the beginning, whereas in Bitcoin, the first few nodes have only a slightly higher clustering coefficient than younger nodes. This is due to the fact that those old Bitcoin nodes still get many incoming connections from much younger nodes, whereas the old Ethereum nodes can not accept anymore connections from an early time on and we start with m initially connected nodes.

The difference between theoretical and lite sim Ethereum indicates that the nodes in the latter do not have sufficient information about the network, i.e., nodes from the full range of *id*s. We believe the reason to be twofold. First, the peer choosing mechanism has an influence on the topology and second, lite sim Ethereum does not manage to mix up the network enough due to lacking functionality.

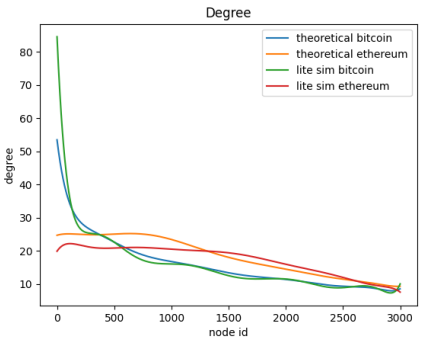


Figure 4.1: Comparison of degree distribution

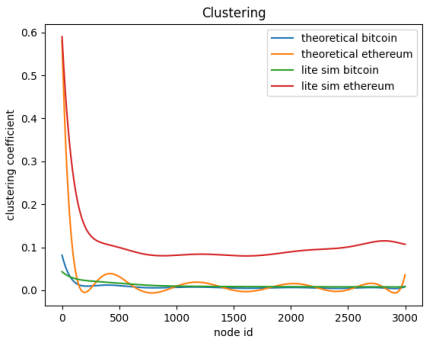


Figure 4.2: Clustering coefficient per node.

4.4 Centrality

In this chapter we develop a formula to compute the probability distribution of shortest path length for each pair of nodes in the network. These results can then be used to compute the closeness centrality for each node. Closeness centrality measures the average distance to all other nodes in the network. Closeness is significant for blockchain networks because a low closeness expresses the delay in receiving information about transactions and new blocks which can be directly translated to a loss of computation power and money.

4.4.1 Analytical shortest paths

Let us fix a node i and estimate the probability distribution of the shortest path length from i to a random node j. Let P_{ij}^l denote the event that a path from i to j of length l exists. Note that $P_{ij}^l = P_{ji}^l$. From now on we assume that $i < j$ w.l.o.g.
For the path from i to j to be of length 1, i and j need to be neighbours.

$$Pr[P_{ij}^1] = \begin{cases} \frac{m}{N(j)} & \text{, if } i > f(j) \\ 0 & \text{, else} \end{cases}$$

If the shortest path is of length 2, then there exists a path of length 2 and also i and j are not neighbours. By conditional probability we get

$$Pr[P_{ij}^2 \cap \neg P_{ij}^1] = Pr[P_{ij}^2 | \neg P_{ij}^1] * Pr[\neg P_{ij}^1] \tag{4.2}$$

We employ an indicator variable X_{ijk} which is 1, iff k connects i and j given that i and j are not neighbours. There are 3 different constellations:

1. $i < j < k$: k is the youngest node and connects to both i and j.
2. $i < k < j$: k connects to i whereas j connects to k.
3. $k < i < j$: both i and j connect to k.

$$Pr[X_{ijk} = 1] = \begin{cases} \frac{m}{N(k)} * \frac{m-1}{N(k)-1} & \text{, case 1.} \\ \frac{m}{N(k)} * \frac{m}{N(j)-1} & \text{, case 2.} \\ \frac{m}{N(i)-1} * \frac{m}{N(j)-1} & \text{, case 3.} \end{cases}$$

If $i, j < m$, X_{ijk} simply is 0, because we start the network with m fully connected initial nodes.
We can reformulate equation 4.2 as

$$Pr[P_{ij}^2 \cap \neg P_{ij}^1] = (1 - \prod_{k \neq i,j} Pr[X_{ijk} = 0]) * (1 - Pr[P_{ij}^1])$$

Given $m < i < j$:

$$\begin{aligned}\prod_{k \neq i,j} Pr[X_{ijk} = 0] = &\prod_{k=j+1}^{N} 1 - \left(\frac{m}{N(k)} * \frac{m-1}{N(k)-1}\right) \\ &* \prod_{k=i+1}^{j-1} 1 - \left(\frac{m}{N(k)} * \frac{m}{N(j)-1}\right) \\ &* \prod_{k=1}^{i-1} 1 - \left(\frac{m}{N(i)-1} * \frac{m}{N(j)-1}\right)\end{aligned}$$

As the estimation of the probability distribution gets much more complicated for longer paths, we take the recursive paths approach from [7] and adapt it for our purposes.
Let the tail-distribution $F_{ij}^N(k) = Pr(d_{ij} > k)$ denote the probability that the distance between nodes i and j in a network of size N is larger than k. For any two nodes we have that $F_{ij}^N(0) = 1$ and $F_{ij}^N(1) = q_{ij} = 1 - p_{ij}$ where p_{ij} is the probability that nodes i and j are neighbours. This probability is dependent on the size of the network at the time when the younger node was added and whether the younger node still accepts incoming connections. We get the probability distribution $P_{ij}^N(k)$ from $F_{ij}^N(k-1) - F_{ij}^N(k)$. The authors of [7] formulated the following expression

$$F_{ij}^N(k) = F_{ij}^N(1) \prod_{l=2}^{k} P_{ij}^N(d_{ij} > l | d_{ij} > l - 1) \tag{4.3}$$

where we adapt the right hand side as follows to account for the connection probabilities of specific node pairs.

$$P_{ij}^N(d_{ij} > k | d_{ij} > k-1) = \prod_{l \neq i,j}^{N} \Big[q_{il} + p_{il} * P_{lj}^{N \setminus i}(d_{lj} > k-1 | d_{lj} > k-2) \Big] \tag{4.4}$$

We can close the recursive equation with $P_{ij}^{N'}(d_{ij} > 1 | d_{ij} > 0) = F_{ij}^{N'}(1) = q_{ij}$. Note that one step in the recursion leaves i out, as it would otherwise form a loop in the path.
The algorithm to compute this has a time complexity of $O(kN^{2+k})$ where k is the maximum path length computed. To make it faster, we only consider a random 1% subset of node pairs. Comparisons show that the results are close to the ones received from calculating over all pairs.
Furthermore, the probability for the path to be longer than $\lceil \log(N) \rceil$ is small and

such a path almost never appears in the simulations. Hence we only calculate the paths up to a length of 4. Because of limited resources, we only compute the shortest paths for a network of size 1000. The results can be seen in Figure 4.3. Note that the y-axis shows the percentage of all node pairs. E.g., 20% of all node pairs in the theoretical Bitcoin network with 1000 nodes are separated by a shortest path of length 2. This approach does not account for correlations between shortest paths which share at least one edge. Whereas the results for path lengths of 1 and 2 are similar to the results from equation 4.2, this approach does not provide reasonable results for longer path lengths.

4.4.2 Numerical shortest paths

In order to get numbers for larger networks, we simulate networks with the same outbound and inbound distributions but with a random choice of connections. The results of this method are in agreement with the analytical results for the degree distribution and clustering coefficients. We measure the shortest paths for all pairs and use these results to compute the closeness centrality. The average shortest paths of the random Bitcoin and Ethereum networks compared with the lite simulations can be seen in Figure 4.4. We observe that the random and lite simulation results of Bitcoin are in harmony.

But there is a large discrepancy between both random and lite simulation Ethereum results. We attribute this to the same reasons described in Subsection 4.3

4.4.3 Closeness

We can plug in the results from Section 4.4.2 and get the results in Figure 4.5. Axis x depicts the *id*s, while axis y illustrates the closeness centrality per node.

The results of both Bitcoin-like networks imply that nodes which have been around for a while have an advantage. Higher closeness implies that nodes receive new information, e.g., on the blockchain state, earlier than others. This is important in terms of fairness in the network.

Just like with clustering and shortest paths, we observe here again a difference between both Ethereum-like networks. Furthermore, we see that the closeness first rises and then slightly drops again. This implies that the network grows in a linear manner, where the oldest and youngest nodes are far away from all other nodes. This is due to the fact that new nodes only connect to nodes which just have been added to the network too, because the older nodes are already full.

The large difference between the first few Bitcoin and Ethereum nodes can be explained with the large difference regarding their degrees.

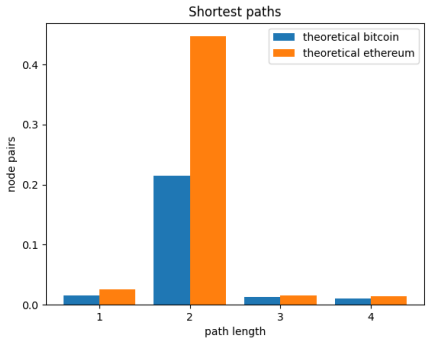


Figure 4.3: Shortest paths with 1000 nodes calculated with theoretical model.

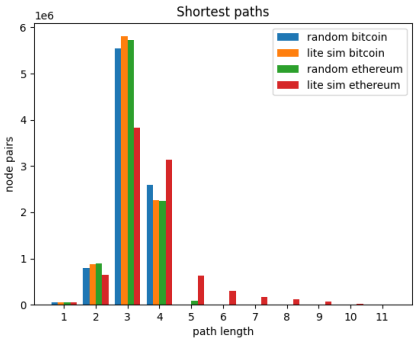


Figure 4.4: Shortest path lengths.

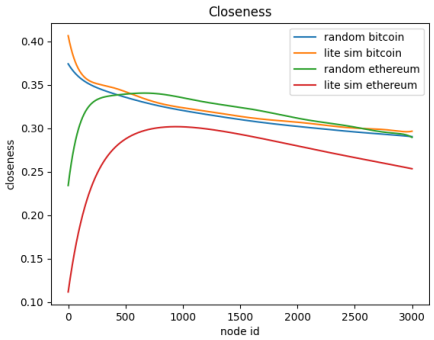


Figure 4.5: Closeness per node.

Chapter 5

Discussion

5.1 Discussion

In networks which allow for high degrees, like in Bitcoin, we have seen that the old nodes receive the most connection requests and hence, have a higher degree than nodes which have been added at a later time. Furthermore, no node gets full in a network with 3000 nodes. This leads to a more mixed up network which results in overall low clustering coefficients. Bitcoin networks have a low average of shortest paths. Because of their high degrees, old nodes have a higher closeness centrality than their younger peers.

In Ethereum-like networks, due to the low upper bound on total connections, many nodes reach a full degree of 25. Therefore, Ethereum achieves an even degree distribution. Even with the mixing up mechanism in the full simulations, Ethereum nodes evince a higher clustering coefficient than Bitcoin nodes. Whereas the theoretical model shows longer shortest paths, the full Ethereum simulation reaches low values for shortest paths and a rather even closeness distribution, with exception of the youngest nodes.

We attribute those findings to the linear construction of the Ethereum network, which maintains even after many dis- and reconnections.

Questionable is the role of the virtual time steps in the simulations, which affect the degree of dynamism and dictate the rate of dis- and reconnections in the network.

5.2 Security and Fairness

What do the findings imply from a security and fairness perspective? Having a high clustering coefficient, like in the Ethereum network, implies not having many peers from other parts of the network. This translates, same as a low closeness centrality, to a possible delay when it comes to receiving information about new blocks or transactions. In Bitcoin, old nodes are given a lot of power, as they get

connected to the most. This is also not what is best from a security and fairness perspective. If shortest paths are long, it gives a higher chance of succeeding in a front runner attack [4]. Ethereum-like networks should have a mechanism to mix the network up. It is essential to have an upper bound on connections, especially incoming ones, to mitigate eclipse attacks [5, 6]. A connection bound between the ones of Bitcoin and Ethereum seems best.

Chapter 6

Conclusion & Future Work

6.1 Conclusion

In this thesis, we compared degrees, clustering coefficients, shortest paths, closeness centrality and the connection distribution of simulated Bitcoin and Ethereum peer-to-peer networks. Furthermore, we built a theoretical model to calculate those properties. Then, we compared them to the results of lite simulations, which have been adapted to observe the effects of the different peer-choosing mechanisms.

From our study, we find that the Ethereum network has a linear construction, which leads to an even degree and closeness distribution and high clustering, but still manages to have short average shortest paths. From a security and fairness perspective, its better to have an even degree- and closeness-distribution like Ethereum, low clustering coefficients like Bitcoin and short average shortest paths, which is reached by both.

6.2 Future work

An interesting future direction would be extensions to the theoretical model to account for connection dynamics, where nodes go offline, reconnect, and retry to establish connections to other nodes upon failure, as well as shortest path results for bigger networks. Interesting would be to apply analysis of more network properties, considering, e.g., state and location of nodes, have single nodes deviate from the protocol or even attack the network. How much the dynamics in connections affect the topology poses the most important open question.

Bibliography

[1] D. Spielmann, "Evaluation of network connection strategies in cryptocurrencies," Oct. 2019, bachelors thesis, ETH Zurich.

[2] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of Modern Physics*, vol. 74, no. 1, p. 47–97, Jan 2002. [Online]. Available: http://dx.doi.org/10.1103/RevModPhys.74.47

[3] Y. Marcus, E. Heilman, and S. Goldberg, "Low-resource eclipse attacks on ethereum's peer-to-peer network," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 236, 2018.

[4] S. Eskandari, S. Moosavi, and J. Clark, "Sok: Transparent dishonesty: front-running attacks on blockchain," 2019.

[5] G. Xu, B. Guo, C. Su, X. Zheng, K. Liang, D. S. Wong, and H. Wang, "Am i eclipsed? a smart detector of eclipse attacks for ethereum," *Computers & Security*, vol. 88, p. 101604, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404818313798

[6] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *24th USENIX Security Symposium (USENIX Security 15)*. Washington, D.C.: USENIX Association, Aug. 2015, pp. 129–144. [Online]. Available: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/heilman

[7] E. Katzav, M. Nitzan, D. ben Avraham, P. L. Krapivsky, R. Kühn, N. Ross, and O. Biham, "Analytical results for the distribution of shortest path lengths in random networks," *EPL (Europhysics Letters)*, vol. 111, no. 2, p. 26006, Jul 2015. [Online]. Available: http://dx.doi.org/10.1209/0295-5075/111/26006

[8] M. Nitzan, E. Katzav, R. Kühn, and O. Biham, "Distance distribution in configuration-model networks," *Physical Review E*, vol. 93, no. 6, Jun 2016. [Online]. Available: http://dx.doi.org/10.1103/PhysRevE.93.062309

[9] "Ethereum source code," https://github.com/ethereum/go-ethereum, 2021.

[10] "Code repo," https://github.com/alainga/networksim, 2021.

[11] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of clients in bitcoin p2p network," 2014.

[12] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, Jun 1998. [Online]. Available: https://doi.org/10.1038/30918

[13] A. Bavelas, "Communication Patterns in Task-Oriented Groups," *Acoustical Society of America Journal*, vol. 22, no. 6, p. 725, Jan. 1950.

[14] G. Sabidussi, "The centrality of a graph." *Psychometrika.*, vol. 31, no. 4, pp. 581–603, Dec. 1966.

[15] T. W. Valente, K. Coronges, C. Lakon, and E. Costenbader, "How correlated are network centrality measures?" *Connections (Toronto, Ont.)*, vol. 28, no. 1, pp. 16–26, Jan 2008, 20505784[pmid], PMC2875682[pmcid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/20505784