# POLITECNICO DI TORINO

Master's Degree in Ingegneria Informatica (Computer Engineering)

Master's Degree Thesis

# Topological analysis and simulations on centrality in the Lightning Network


Supervisors

Prof. Antonio VETRÒ

Dr. Marco CONOSCENTI

Candidate

Angelo PILI


July 2021

# Summary

Bitcoin is the first decentralized cryptocurrency ever created, and due to the transaction limit of the blockchain it is not scalable.

To overcome the scalability problem, payment channel networks have been developed. Payment channel networks reduce the number of transactions loaded on the blockchain, allowing unbounded off-chain payments in almost instant time.

Lightning Network is the most prominent payment channel network built for Bitcoin. With its time-based smart contract called HTLC, it allows off-chain payments across channels without trusting other participants, with small fees to route payments. Payments are routed with a few hops from peers across the network.

Topological studies show that since channel creation takes funds from the blockchain nodes tend to create few channels by connecting to central nodes in the network rather than creating ad hoc ones.

A very important research field is understanding if there are nodes that may have an important or central position in the network or if there are hubs. These nodes can compromise the decentralization of the network and if they are not cooperative, they can increase the rate of payments failure.

This work concerns the research of these nodes on the Lightning Network. There are different centrality measures in graph theory that define different concepts of centrality of a node based on several criteria. Main centrality measures were presented, and the Lightning Network nodes that are central to these measures were identified. For each centrality measure, the central nodes were removed, and the topology of the network was studied. A random node removal was also performed to test the network's resistance and behavior to random attacks or offline nodes. In addition to the topological analysis, in this work simulations on the Lightning Network were performed: for each centrality measure, central nodes were removed and the resulting network was simulated. To conduct simulations, CLoTH was used, a simulator of the Lightning Network: it simulates payments on the Lightning Network and produces performance measures, such as probability of payment success and average payment time.

The topological results showed that removing a few central nodes was sufficient

to detach many nodes in the network. Most of these nodes were isolated, confirming that nodes had few connections to central nodes. Payments needed on average more hops to reach the destination.

CLoTH simulations confirmed the increase of average number of hops, and showed also that the average payment time and the rate of payment failure increase when removing central nodes.

The results showed that Lightning Network has some nodes that are important to the network. The removal of these central nodes creates inconveniences in the success of the payments not comparable to a random removal of nodes. However, despite their removal, the network is still fairly well connected.

# Table of Contents

# List of Tables

# List of Figures

# Acronyms

**LN**
: Lightning Network

**LCC**
: Largest Connected Component

**Deg**
: Degree Centrality

**Eig**
: Eigenvector Centrality

**Clo**
: Closeness Centrality

**Bet**
: Betweenness Centrality

**VRank**
: Vote Rank

**Load**
: Load Centrality

**Bin**
: Binary Network

**W**
: Weighted Network

**HTLC**

Hash Time Locked Contracts

# Chapter 1

# Introduction

Bitcoin[1] is the first decentralized cryptocurrency that was ever developed. Its main feature is that it allows transactions without the help of a trusted central entity to control them.

The blockchain is the distributed ledger of Bitcoin where all transactions are reported and replicated on all nodes of the peer-to-peer network. Every change must be reported on all copies of the blockchain on the various nodes.

The blockchain requires a large amount of storage space and bandwidth. It also requires time to ensure that the blockchain is propagated to as many users as possible. These factors put a cap on the number of transactions per second. This transaction throughput does not allow Bitcoin and others cryptocurrencies to scale and consequently limits their large-scale use.

To solve the scalability problem and reduce the information load on the blockchain, payment channels have been proposed. Payment channels allow for off-chain transactions between two parties that do not need to be transcribed to the blockchain and therefore are not subject to the blockchain throughput limitation.

Links between multiple payment channels create a payment channel network. Payment channel networks allow for the exchange of money between two nodes that do not have a direct channel between them.

Lightning Network[2] is the most popular Bitcoin payment channel network. At the time of writing Lightning Network has approximately 10,440 nodes and 47,005 channels and has recently surpassed 1,500 Bitcoin in total capacity (approximately $49.5 million). Lightning Network uses a smart contract called Hashed Timelock Contract (HTLC) that allows off-chain payments to take place on a network across channels without trusting other participants, with small fees to route payments.

Topological studies[3, 4, 5, 6, 7] showed that since channel creation takes funds from the blockchain nodes tended to create few channels by connecting to central nodes in the network rather than creating ad hoc ones.

This allows nodes to communicate with a larger window of the network with few channels created.

Since channel creation takes funds from the blockchain nodes tend to create few channels by connecting to central nodes in the network rather than creating ad hoc ones.

In this way, nodes are able to route their payments through a few intermediaries without open new ad-hoc channels.

Nodes that have a strategic position in a network are called central nodes.

A very important field of research is trying to recognize central nodes that may have an important position in the network.

These nodes are very influential in the network given their position and can compromise the decentralization of the network or if they are not cooperative they could reduce the robustness of the network, reduce the number of successful transactions or increase the number of hops to route the payment.

The objective of this work was to study centrality on the Lightning Network. To accomplish with this goal, both a topological analysis and simulations were conducted in this work.

In the topological analysis, central nodes of the LN were idenitified, according to the centrality measures provided by graph theory. For each centrality measure, central nodes were removed, and the effect of the removal on the topology of the network was studied.

The simulations were performed using CLoTH[8], a Lightning Network simulator. CLoTH simulates payments on the LN and provides performance results such as probability of payment success, average payment route length and average payment time. For each centrality measure, simulations were performed, where the Lightning Network without central nodes was given in input to CLoTH.

Topological results showed that removing a few central nodes can reduce the number of nodes in the network. In addition, there was an increase in network diameter and average distance between two nodes.

The simulations on CLoTH confirmed the results obtained from the topological analysis performed previously such as the average increase in the number of hops per payment, adding details regarding the increase in the average time for payments and more payment failures.

The obtained results denote that the central nodes have indeed some influence on the network. However, despite their removal, the network is still fairly well connected.

The thesis work is structured as follows. The chapter 2 discusses the background of Bitcoin, the blockchain, Lightning Network and introduces the main measures of centrality. The chapter 3 presents the topological analysis of various centralities on Lightning Network and the removal of central nodes. The chapter 4 presents the CLoTH simulator and the simulations performed with the various versions of the network. Finally, the chapter 5 provides conclusions.

# Chapter 2

# Background and Related Works

The follow-up Chapter will present the background regarding Bitcoin and Lightning Network and also present centrality measures presented in the `networkx` library to give a context to the thesis.

The section 2.1 will discuss Bitcoin transactions and the blockchain. In the section 2.2 introduces the concept of scalability and in 2.3 will be presented a possible solution that is payment channel network of which Lightning Network is the most popular. At the end, the section 2.4 will be discussed the various characteristics of the main measures of centrality of `networkx` library.

## 2.1 Bitcoin, cryptography, and transaction

This section will briefly introduce Bitcoin, transactions, and the blockchain. The following readings [1, 9, 10, 11] are recommended where these topics will be covered in more detail.

**Bitcoin** (₿) is the first implementation of a decentralized digital currency. It is decentralised because, in contrast to most of the other payment systems, it does not need a trusted third part like banks or governments for keep track of transactions between participants in the Bitcoin network.

The Bitcoin network is a peer-to-peer network composed by all the nodes, such as personal computers or smartphones, with a running open-source software, the Bitcoin protocol.

Since Bitcoin is decentralised, this peer-to-peer network regulates transactions and minting of new currency according to the consensus of the network. Transactions are accepted through cryptography by nodes and inserted in the distributed public ledger of Bitcoin, the blockchain.

**Asymmetric cryptography** gives the security in the untrusted peer-to-peer Bitcoin network. It consists in creation of a pair of keys, the public key, and the private key.

The public key is the address of the Bitcoin node. It can be shared online and it is utilized for sending money to the relative Bitcoin user. The private key proves that a transaction belongs to the owner of the corresponding public key and gives the possibility to spend bitcoins.

This method of encryption by the private key and decryption by the public key is called digital signature.

**Transactions** consist of a change in bitcoin ownership from previous transactions. It consists of one or more inputs and one or more outputs.

In the output there is the quantity of bitcoins that are transferred to another user and a locking script that indicates certain condition required to spend the relevant amount of bitcoins. An input refers to a previous output transaction via a pointer and include the unlocking script that satisfies the conditions set in the output locking script.

Typically, the locking script is the recipient's public key (address). The owner that wants to spend the money needs to unlock the locking script with his digital signature on the hash of the relative transaction.

Exist complex locking scripts like the multi-signature script where N public keys are recorded, and M is the minimum number of signatures relative to these public keys required for validation. These transactions are called M-of-N multi-signature transactions since they require M signatures between N to be validated.

The signature allows to verify the ownership, but it is impossible to verify that this transaction was already spent.

**Double spending** is a digital payment fraud that consist of a malicious user that make two transactions using the same money. In centralised digital payment system this problem is simple to avoid because the central authority has a complete view of all transactions and can therefore deny the second one. The Bitcoin network utilizes the blockchain to solve this problem.

**The blockchain** is the public ledger of the Bitcoin network, and it is distributed in all the Bitcoin nodes of the network. The blockchain differs from common payment system ledgers since it does not record the total amount of bitcoins relating to each owner but is a long chain of blocks within which transactions are present.

These blocks recreate all the Bitcoin's transaction history since the genesis block, and they also include the minting of new currency. A block is chained by the hash of the previous block. The hash is useful to prevent modification in transactions inside a block because this changes the block hash that become invalid such as the following blocks and hashes.

When nodes receive a transaction that refers to an already spent transaction, they discard this last one to prevent the double spending. The chronological order is given by the hash of the previous block that must already exist at the creation of the next one. Every node in the Bitcoin network can add new blocks in the blockchain, and they are called miners.

**Mining** is the process used to the nodes for add new blocks in the blockchain. This process consists in arbitrarily collecting transactions broadcasted in the peer-to-peer network into a block. Any node can create a block, but only one is added at the blockchain. The choice of the next block of the blockchain is given by the resolution of a proof-of-work.

The mining process rewards miners with the minting of new coin and the fees of the transactions inside new blocks added in the blockchain.

This process incentivizes the nodes to behave correctly and to create valid blocks since the invalid blocks would not be accepted by the other nodes on the network and the attacker would not receive the reward.

In average, the resolution of a proof-of-work is every 10 minutes, and the Bitcoin protocol adjusts its difficulty to maintain this average time. This time is necessary to allow the nodes of the network to be notified of the creation of the new block and update their version of the blockchain.

The **proof-of-work** is a mathematical puzzle and consists in finding the hash of the new block. In the block there is a nonce which is a random number, the proof-of-work algorithm consists in a continuous search for a value of this nonce until the calculated hash is less than a given number.

The first node that complete this task, add his block in the blockchain and notify all the network of the change. This process requires a lot of computing power to increase chances of successfully finding the nonce before others. The probability of success is directly proportional to the fraction of the computational power of the network owned by the miner.

**Fork** When two miners find the solution of the proof-of-work and simultaneously create their version of the new block, a fork occurs in the blockchain, two blocks pointing to the same block.

The same situation would happen if an attacker rewrote an alternate block to substitute the last one. This situation create ambiguity, break the consensus and potentially a double-spending problem, so only one block must be chosen.

Bitcoin nodes to resolve this problem, choose the block that lies in the longest path of the blockchain ignoring the other branches, because this branch is the one that is considerate valid by most nodes.

In the case a fork is at the top of the chain, both branches are potentially valid.

At the creation of the new block, the miner arbitrarily chooses one of the two blocks in the branch, concatenate the new block with the hash of the chosen block. Transactions inside this block now have a confirmation and that ensure that these transactions are accepted by the network.

The other block now is in a shortest branch, so this block is discarded and transaction inside are not considerate confirmed.

After six confirmation the block is considered irrevocable. All transactions inside this block are considered accepted by whole the network. A malicious user that wants to invalidate this block by create a longer alternative branch require a huge amount of computational power to recalculate six blocks.

## 2.2 Scalability problem

An important question regarding Bitcoin and the blockchain regards the scalability. Payment systems like Visa can process 1,700 transaction per second on average.

In contrast, Bitcoin network can process an average of only 4 transactions per second with a peak of 7.

With the increase in the number of users and consequently in transactions, this low throughput creates a bottleneck for large-scale use of Bitcoin.

This throughput is set by two main factors of the Bitcoin protocol. The size of the blocks, less than 4MB and the generation time of new blocks[12].

**Reparameterization** The community can try to adjust these two parameters to attempt to increase the throughput of the network like the Visa one. The block size would need to be increased to 377.5 MB or the generation time reduce from 10 minutes to 1.6 seconds.

If the block size increase to much, it is difficult to run Bitcoin in common personal computer, due to an increased storage space required for the blockchain and to store transactions and for the bandwidth needed for the communication with other nodes.

This solution affects the decentralization of the network because only some nodes could manage these amounts of data at their best.

Reduce block generation time create some problem with the propagation time of the new block. In just 1.6 seconds, the new mined block does not reach all the nodes of the network.

In average a block needs 14 seconds to reach the 99% of the network. Setting the new block generation time under 14 seconds leads to more forks and double-spending attacks in the network.

If the block generation time is set at the time limit to reach the 99% of the network, the throughput is about 188 transaction per seconds with block size of

1MB, increasing this size affects the block propagation time[12].

**Payment Channel Network** One solution to the scalability problem is to reduce the number of transactions on the blockchain. Payment channel networks have been designed on this concept.

These transactions are just as valid as if they were reported on the blockchain. Since they stay off-chain they do not have to undergo the limit given by block space and latency times, allowing high throughput with short times.

## 2.3 Payment Channel Network

A payment channel network is a network of payment channels that allow off-chain payments. These payments are not subject to the throughput imposed by the Bitcoin network.

A payment channel is a two-way channel through which two parties can exchange payments off-chain. For this reason, payment channel networks are considered layer 2 protocols.

The following will illustrate how a payment channel is used. At first, two users Anna and Bob open a payment channel with each other by funding the channel with a portion of their money, for example each with 0.5 ₿.

The channel will have a total capacity of 1 ₿ and the respective balances of the two participants of 0.5 ₿ each.

Now that the channel is established, Anna and Bob can make transactions with each other without the need to notify the network and transcribe into the blockchain.

Their balances will be updated accordingly. For example, if Anna wants to pay Bob 0.1 ₿, the channel balances will be updated to 0.4 ₿ to Anna and 0.6 ₿ to Bob. Channels are bidirectional so a transaction from Bob to Anna is also possible.

The payment channels network is established by connecting these channels. This avoids the creation of ad hoc channels with all participants in the network.

For example, Anna has to pay Carl but there is no payment channel between them. Anna and Carl both own a payment channel with Bob. Anna transfers the amount of bitcoin that she has to pay to Bob. Bob sends the same amount to Carl via their payment channel.

Lightning Network is the mainstream implementation of a payment channel network in the Bitcoin network.

### 2.3.1 Lightning Network

Lightning Network[2] is the most widely used payment channel network that relies on the blockchain. The Lightning Network protocol defines how to create new channels and how to route payments through the network. In the following sections, channel creation and payment routing will be explained.

**Channel opening** is initialized by a funding transaction by the two owners, Anna and Bob. The funding transaction funds the newly created channel with some bitcoins from the participants. This transaction must be committed to the blockchain. Funding transaction will take in input bitcoins from both participants. For example, Anna and Bob create a 1 ₿ channel by funding each with 0.5 ₿. At this time both will have 0.5 ₿ of initial balance in this channel.

The output of the funding transaction is the total initial amount, in this case 1 ₿. The output is one locking script a 2-of-2 multi-signature, and a transaction needs both the signatures of Anna and Bob in input to spend this money.

When the funding transaction is committed to the blockchain, the channel is open. Both parties also create a commitment transaction where the two signatures of the participants are present as input. The commitment transaction has in input the funding transaction output. The output returns balances to their respective owners. Anna signs the commitment transaction and send it to Bob who also signs it.

The commitment transaction is not written to the blockchain until a party wants to close the channel.

**Commitment transaction** is used to make payments in the payment channel. When one party wants to send bitcoins to the other, it creates a commitment transaction and sends it signed. The receiver in turn signs the transaction and sends it back.

Output balances are updated according to this new transaction. Commitment transactions can be done by both parties instantly without updating the status on the blockchain. An off-chain transaction of this type is done for every exchange of money between the two parties.

**Closing a channel** to close a channel the two parties need to broadcast the most recent commitment transaction at the blockchain. The two balances are refund according to the commitment transaction to the two parties. Once this transaction is confirmed, the channel is closed.

Channel closing can be done even if one of the two participants is uncooperative or unresponsive.

**Punishment** Anna pays some bitcoins to Bob via a commitment transaction and the balances are updated. Anna could later cheat Bob and close the channel by transmitting a previous transaction to the network as it would bring her more money back.

Lightning Network's protocol allows to punish misbehaviour by malicious users. If a user sent an old transaction to the blockchain, the protocol allows to the counterpart to demonstrate the malicious behaviour of the first one. The malicious user will lose all the fund in the channel. The protocol allows the punishment of a malicious user. The cheated user can prove that the submitted transaction is older than another one. The protocol will then punish the first user by depriving him of all his funds in the channel that will go to the cheated user.

This mechanism is not automatic, users must actively check the blockchain. If a user does not check the blockchain within a certain timeout from the transmission of the transaction, the malicious user will succeed in his intent.

This mechanism allows for the creation of channels without necessarily trusting the other party.

**Hashed Time Lock Contracts (HTLC)** is a smart contract used in Lightning Network. HTLC is a locking script that allows secure transaction through the payment channel network without the need for trusted participants.

HTLC consist in a hash of a secret number $R$ and a time lock that is an expiration time. The secret number $R$ is a preimage create by the final recipient given to a hashed function.

A user that wants to redeem the transaction must show the secret $R$ within the expiration time, otherwise the payment is denied. In fact, the time lock ensures a refund if the secret was not revealed within this expiration time.

Now it will be illustrated a payment using HTLC. Let us assume that exist 2 channels, 1 between Anna and Bob and one between Bob and Carl. All these channels have 1 ₿ of capacity and all the balance are 0.5 ₿ in the initial state.

Anna wants to pay 0.1 ₿ to Carl trough the network without opening a new direct channel ad hoc to him. Both Anna and Carl have a channel with Bob, so exist a route between them.

Carl, that is the final recipient of the transaction generate a random number $R$ and keep it secret. Carl calculates the hash $H$ of the random number and share this hash with Anna, the initial payer.

Anna creates a HTLC, payable to the hash $H$, and a timeout $TL$ that there is a number of block in the future, i.e. 6 blocks. The total amount of this payment counts not only the amount that Anna wants to pay at Carl but also some fees for the intermediary Bob and send it to him for instance 0.101 ₿ . This transaction scales from Anna's balance in the channel with Bob this amount. The HTLC will pay Bob if he proves that he knows the secret R, otherwise Anna will get the bitcoins back after the timeout.

The balance of the channel in the commitment transaction that represent the state of the channel now has 3 outputs, 0.5 ₿ balance of Bob, the balance of Anna 0.399 ₿, and 0.101 ₿ committed in the Anna's HTLC.

Bob can redeem this commitment only with the secret number generated by Carl before the timeout of 6 blocks.

Now Bob establish a HTLC with Carl. The commitment transaction has 3 output with 0.4 ₿ balance of Bob, 0.5 ₿ balance of Carl, and 0.1 ₿ committed in the Bob's HTLC. The timeout in the HTLC is reduced, i.e. 5 blocks in the future.

Carl knows the secret number $R$ because it is created by him, so he can claim the HTLC offered by Bob. Carl sends the secret number $R$ to Bob that can pay Carl and claim the HTLC of Anna.

If $R$ is not revealed after the timeout, HTLCs failed and refunds the payer of each channel. The timeout is decreasing so that each participant has time to receive the secret and be able to use it.

The balances of the two channels now are 0.399 ₿ balance of Anna and 0.601 ₿ balance of Bob in the channel between Anna and Bob, and 0.4 ₿ balance of Bob and 0.6 ₿ balance of Carl in the channel between Bob and Carl.

The payment was successful, and Bob earned a small amount of money for participating. These fees incentive the network's users to cooperate.

Lightning Network also allows these payments to be routed through multiple intermediaries. The only limits are that there must be a route between the payer and the receiver and that these channels must have sufficient capacity to route the payment and the fees.

One of the most interesting studies that can be done on Lightning Network is to find nodes that can be central to the network.

We can consider the Lightning Network as an undirected graph. The nodes are the users of Lightning Network, edges represent channels. The graph is an undirected graph because the channels are bidirectional, and we cannot make assumptions about the balances.

There are different criteria with which to define central a node. Some centrality measures from the `networkx` library[13] and their criteria for defining a node as central will be discussed below.

## 2.4 Centrality Measures

A very important concept for the study of complex networks, like the Lightning Network, is the identification of the most important nodes. The importance of a node in network theory is given by a centrality measure.

There are many different centrality measures, each one considers a different criterion to evaluate the relative topological importance for a node in the network. Because of this there is not a best centrality measure a priori, but one could describe very well one network feature and poor another[14].

On following will be presented various centrality measures and what features of nodes are considered to the computation of the relative measure, and the Adjacency Matrix[15] that is very useful in some centrality measure.

For a more in-depth reading see *"The structure of complex networks: theory and applications"* Estrada(2012)[16].

**Adjacency Matrix**: each element $a_{ij}$ of the adjacency matrix A indicates if exist an edge between node $i$ and node $j$. If exist the connection between the two nodes, the element $a_{ij}$ is equal to 1 and the two nodes are nearest-neighbors, 0 otherwise.

The elements on the diagonal of the matrix are all zeroes if there are no self-loops in the network. If the network is a directed network, the corresponding adjacency matrix may not be symmetrical.

An interesting fact is that multiply k times the adjacency matrix by itself, the elements element $a_{ij}$ of the resulting $A_k$ matrix give us the number of k-length paths between nodes $i$ and node $j$.

**Degree Centrality**: the degree centrality[17, 18] for a node in a network is defined by the number of edges that are incident upon the node itself.

For this centrality more a node has connections with other nodes, more is important and bigger his degree is. Often this value is standardized by dividing it by the number of network's nodes minus one.

The degree centrality most important feature is that it takes under consideration only the node's influence for its nearest neighbors. We can deduce the degree for a node in a network from the corresponding row of the Adjacency Matrix.

This centrality measure tells us that a node more is connected with other nodes of the network, more for the network is important.

In the case where the network that we are taking under consideration is a directed network we can discern two different centrality measures, indegree and outdegree centralities. Each one of these centralities take in consideration respectively only the numbers of edges incident to a node and the number of edges incident from a node.

The principal weakness of this method is that is common that more than one node has the same degree.

**Eigenvector Centrality**: the eigenvector centrality[19, 20], in contrast to the degree centrality that accounts only the influence of nearest neighbors to a given node, counts the effects of the influence also for the other nodes.

The degree centrality awards 1 point for the centrality for each edge of the node, the principle behind eigenvector centrality is that nodes are not all equally relevant in a network. In fact, a node that have a high degree does not necessarily have a high eigenvector centrality and vice versa.

Bonacich [21] proposed the eigenvalue centrality in which a node's centrality is its summed edges to others, weighted by their centralities. The method behind is to find eigenvalues $\lambda$ of the adjacency matrix $A$ and find the non-zero vector $x$ that satisfies the equation $\lambda x = Ax$. If it is defined in the function the parameter "weight" it is used this attribute of the edge instead of the elements of adjacency matrix.

This vector is an eigenvector of $A$ and exist only one eigenvector for each eigenvalue. Usually, the largest eigenvalue is the preferred one. This eigenvalue gives us the principal eigenvector.

Since $\lambda$ and $x$ are both unknown, to resolve this problem is used the power method to compute the eigenvector, but the convergence is not guaranteed.

One important characteristic of the eigenvector centrality is that can differentiate two or more nodes with the same degree, except in case of regular networks where all the nodes have the same number of neighbors.

For directed networks, similarly at the degree centrality, there are two different measures, left eigenvector centrality and right eigenvector centrality. The first one is an extension of the indegree centrality and is also known as "prestige", the second one an extension of the outdegree.

Apply these two centralities methods could give rise some difficulties if the network presents some nodes with indegree/outdegree equal to zero, because the corresponding node has left-eigenvector/right-eigenvector centrality equal to zero and nodes with edges pointing to this node would not receive any score for point to it.

**Katz Centrality**: like the eigenvector centrality, the computation of the Katz centrality[22] for a node takes in consideration centralities of its nearest neighbors and the other nodes of the network that are reachable through them.

For compute The Katz centrality we need two important parameters: the attenuation factor $\alpha$ and $\beta$ that represents an extra weight to provide to the immediate neighbors.

The attenuation factor $\alpha$ takes consideration of the topology of the net and so attenuate the importance of the node by this value for each step. The value of this parameter should be positive and strictly minor to the inverse of the principal eigenvalue of the adjacency matrix.

To give at nodes a small amount of centrality and to avoid the problem of the directed networks presented of the eigenvector centrality, it is used the bias $\beta$.

The Katz centrality is a generalization of the eigenvector centrality (if $\alpha$ is equal to the inverse of the principal eigenvalue of $A$ and $\beta = 0$ we have the eigenvector centrality).

**Closeness Centrality**: in the closeness centrality nodes of interest are those that are relatively close to all other nodes of the network with few steps. For this measure is not relevant the number of edges of a node or if the nodes are connected with others important nodes, a node is central if can reach the other nodes with short paths.

On average these nodes could communicate with the others in the shortest number of steps. This centrality measures the mean distance from a node to the others. The closeness centrality is the inverse of this mean distance of shortest paths. Nodes with a high closeness centrality score have shortest distances to all other nodes.

The shortest path distance of two nodes $i$ and $j$ in a network is the number of edges in the shortest path between the two nodes.

In directed networks the distance between $i$ and $j$ could be different to the distance between $j$ and $i$.

A problem situation in networks is that because this is an average sum this method cannot distinguish two nodes that have the same closeness centrality, but one could be more central because needs fewer steps to reach nodes rather than the other.

We need a fully connected network or component to calculate this centrality.

**Information Centrality / Current-Flow Closeness Centrality**: the information centrality[23] is equivalent to the Current Flow variant of the Closeness Centrality. The closeness centrality assumes that the information flows along only in the shortest path, without that the possibility of split this information. The current flow closeness centrality let the information flow and split like current in an electrical network.

Current Flow closeness centrality uses the resistance distance notion that differs from the typically distance used on graphs like the shorter path.

The resistance distance is used to avoid some drawbacks in the other distance measures like for instance the other paths, longer than the shorter one, give no contribution to the measure.

Resistance distance has some interesting characteristic. This measure considers the existence of multiple paths between two nodes. More paths of the same length there are between two nodes more the two nodes are closer. The resistance distance considers also that two nodes separated by paths than taken in pairs do not share edges are closer than two nodes that have redundant paths. Similarly at the shortest path, for this distance two nodes are closer if the path is shorter.

**Harmonic Centrality**: the harmonic centrality is a variant of the closeness centrality used to solve the problem with unconnected graph of the closeness centrality.

This variant instead of the inverse of the sum of shortest paths, uses the sum of the inverses of those distances for the calculation of the average distance[24].

**Betweenness Centrality**: the betweenness centrality measures how often a node lies on shortest paths between the other nodes. These nodes are central for the network because they have influence in the network by virtue of the control over the information passing between others.

Nodes that are central for this measure are very important in the communications. In fact, removal of one of the most important of these nodes could disrupt communication between other nodes. This because they lie on the largest number of paths between nodes.

Like for the closeness centrality, exist a current flow version of this measure that use an electrical current model for the spreading of the information in the network[25].

**Communicability Betweenness Centrality**: the concept behind this measure is the same of the betweenness centrality[14], but in this case to calculate the importance of nodes, this measure takes in consideration not only shortest paths, but all possible routes with the introduction of a scaling factor such that longer walks carry less importance.

This measure is very useful not only to see what happens in the network when edges of a certain node are removed, but also to measures the sensitivity of a node's communicability when its edges are subject to infinitesimally changes.

**Group (Betweenness/Closeness/Degree/Indegree/Outdegree) Centralities**: all the previous centralities measures are applied for individual nodes.

This group of centralities are the same kinds of centralities presented before but are version that take in consideration set of nodes and the relationships between this group and all the other individual nodes of the network.

In fact, if the group is composed by only one node the result is the same of the relative one-node counterpart[26].

**Load Centrality** the load centrality is often considered erroneously to be the same measure as the betweenness centrality.

The difference of these two measures is that in the load centrality each node sends a unit amount to each other node. This amount from the respective source is passed to the node's neighbors closest to the target, and in the case that these nodes are more than one this amount is divided equally among them.

The total amount passing through a node during all this process defines its load[27].

**Subgraph Centrality**: the importance of a node for the subgraph centrality is characterise by its participation in all the closed walks that start and end from the node itself.

We can find for a node $i$ the number of closed walks of length $k$ by the $i^{th}$ element of the diagonal of the $k^{th}$ power of the Adjacency Matrix.

The subgraph centrality of a node is defined by the sum of all its closed walks of different length with decreased contribution in function of the length of the relative closed walk. This is possible by the Estrada index that also guarantee the convergence[28].

**Dispersion**: this centrality measure is an extension of the embeddedness of two nodes. Embeddedness tells us how related two nodes are based on the number of neighbors they have in common. The underlying concept considers that if two nodes have a correlation between them, common neighbors of them are part of the same relationship.

The dispersion of the $i$-$j$ link is defined to be the sum of all pairwise distances between nodes in the set of common neighbors of the two nodes $i$ and $j$.

The dispersion tells how much two nodes can reach nodes that are not directly related to each other. The distance function used give us 1 when the two nodes are not directly connected and they do not have common neighbors in the same set, 0 otherwise[29].

**Reaching Centrality**: local reaching centrality of a node $i$ is the portion of all nodes of a directed network that are reachable starting from the node $i$ itself through its outgoing edges to its neighbors. This measure tells us the number of nodes that are reachable with a finite and positive distance.

The average over all nodes of the difference between the local and the highest local reaching centrality of any node give us the global reaching centrality of the network[30].

**Percolation Centrality**: the percolation centrality is peculiar for the study of percolation scenarios. A percolation scenario can be a change of nodes' status like the spreading of a virus through a computer network. The state of a node can change due a spread of the information over the link between the node and his neighbor.

Usually, the state is a real number between 0.0 and 1.0 or binary values. This measure quantifies the relative impact of nodes based on their topological connectivity.

The percolation centrality is an extension of the betweenness centrality, if the state of all nodes is the same the two measures are equivalent[31].

**Second Order Centrality**: The Second Order centrality is a measure meant for overcome the issues relative to the distribution of centrality computation.

The relative importance of the node in the topology of the graph it is like the betweenness centrality. The measure is time based. The algorithm consider a single random walk visiting the network, starting by an arbitrary node in the network and choosing the next node with uniform probability among its neighbors.

To compute their centrality, nodes record the return time of that walk, and the standard deviation of this return time is the second order centrality of a node. So lower value of second order centrality indicates higher centrality for the node.

The intuition behind this measure is that bridge nodes are visited more regularly in a random path than other nodes.

Comparing standard deviations is also useful to detect the presence of critical paths, or traps in the topology. To compute meaningful deviation results each node needs to be visited a few times by this random walk[32].

**Trophic Level**: the trophic level is used in the food chains to give a score to various organism in the chain and the transfer of energy from one part of the ecosystem (producer) to another (consumer). Producers are edges incident of a node, like the in-degree centrality. Consumers are the equivalent of out-degree centrality. In a directed network the trophic level of a node $i$ is defined as the average distance between producers and consumers of a given node plus one. In fact, producers by convection have trophic level equal to 1 (like plants).

**VoteRank**: VoteRank is an iterative method used to identified top influential nodes of the network that can spread the information very well.

VoteRank search influential nodes by a voting algorithm. This algorithm looks for important nodes that do not have a sphere of influence that overlaps too much.

VoteRank identify a set of decentralized influential nodes that have the best spreading ability of the information. These influential nodes are elected by scores given by their neighbor.

In subsequent interactions, the elected spreader nodes cannot vote and the vote of their neighbors will be decreased by a factor. The number of interactions is equal to the number of spreaders that we want[33].

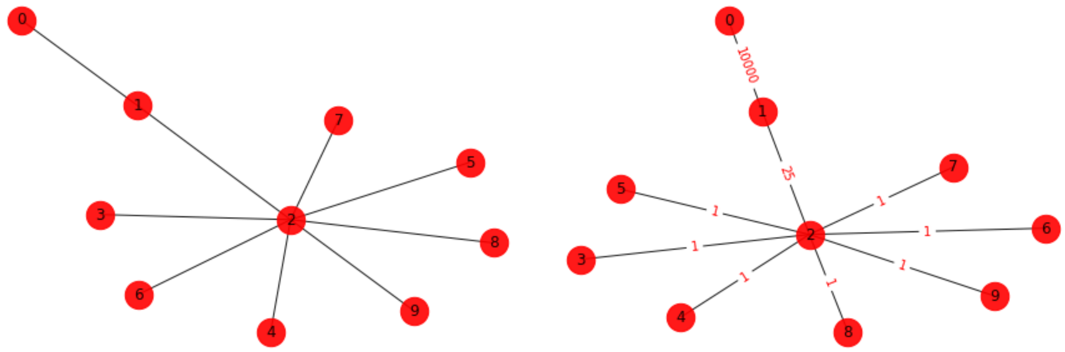
**Weight parameter and Mixed network**: it is important to specify what the weight parameter represents in the various measures.

It is important to specify what the weight parameter represents. The main measures as explained above refer to the case in which the weight parameter is not present. The measures considered all edges equals and is seen as a binary case in which the presence of an edge or not is given respectively by the value one or zero.

The meaning of centrality measures considering the weight parameter change drastically. Second order and Vote rank does not accept the weight parameter in the graph.

For the **degree centrality** if the weight parameter is defined, a node is more central if the sum of the weights of its edges is high. It is also called node strength.

A simple example can be made with degree centrality in the figure 2.1, in the binary scenario, node 2 is the one with the most connections and according to the degree centrality it is the more central node. In the weighted scenario the best node would be the 1 because the sum of the weights of its connections is greater.




**(a)** Degree Binary **(b)** Degree Weighted

**Figure 2.1:** Degree Binary-Weighted differences

The weight parameter of an edge in degree centrality and **eigenvector centrality** is represented by the capacity of the corresponding channel.

Therefore, in the weighed case the number of connections is no longer important, but how strong they are, in fact it is also known as strength of a node. So the measure in the weighted case is no longer connected to the original idea of the degree centrality, and the information about how much connection they have is also lost.

This argument is discussed by Opsahl et al. (2010)[34] where they said that this generalizations proposed also for the betweenness and the closeness have solely focused on the edge weight and non in the number of connections which was the initial concept behind these measures.

They also discuted that it is possible to **combine the two different measures**, the generalized case with the weight parameter and the binary network case by a positive tuning parameter $\alpha$, which indicates the relative importance of the number of ties compared to the weights.

The degree centrality measure proposed by Opsahl et al. is the product of the nearest neighbors of a focal node, and the average weight to these nodes adjusted by $\alpha$ [34].

$$C_D^{w\alpha}(i) = k_i \times \left(\frac{s_i}{k_i}\right)^{\alpha}$$

Where $k_i$ is the degree and $s_i$ the strength of a $i^{th}$ node. If $0 < \alpha < 1$ then having high degree is advantageous, else if $\alpha > 1$ then are preferred nodes that have low degree and high strength.

In the two borderline cases of $\alpha = 0$ and $\alpha = 1$, the formula is equivalent respectively at the degree and the strength [34].

For the **betweenness centrality** and the **closeness centrality** weighted versions compute the shortest path length using Dijkstra's algorithm. In this case the weight parameter it's considered as a cost or a distance.

In the case of Lightning Network, the weight parameter cannot be considered a cost for these two centralities since a channel with high capacity would be considered negative. In fact, nodes connected to high-capacity channels are more likely to be used for making payments than smaller-capacity channels.

According to the study of Opsahl et al. in this scenario the capacity is a parameter to be considered as a tie strenght and not a cost, so before analysis, it is appropriate to compute the **reciprocal of the capacity** that will be used in the Dijkstra's algorithm to find the shortest-paths using this as a cost. High values depict weaks channels and low values represent strongers channels.

Also for these measures they proposed a **mixed measure** where the inverted weights are transformed by a similar tuning parameter used in the degree centrality before using them as distances in the Dijkstra's algorithm [34].

$$d^{w\alpha}(i,j) = \min\left(\frac{1}{(w_{ih})^{\alpha}} + \cdots + \frac{1}{(w_{hj})^{\alpha}}\right)$$

Also in this case the positive tuning parameter $\alpha$ adjusts the proportion by weighted and the binaries distances. With a value of $\alpha$ closely to 0, the algorithm chooses a shortest-path with weak ties, while with a value of $\alpha$ close to 1, is preferred a longer-path but composed of strongest ties [34].

The study of the important nodes of the Lightning Network is one of the most interesting studies to understand the behavior of the network. These important nodes are defined as such according to various criteria.

Most researchers investigating nodes' centrality in Lightning Network to understand the behavior of the network and possible problems on critical issues have commonly utilized various centralities' measures mainly the degree centrality, the between centrality, the eigenvector centrality, and the closeness centrality.

## 2.5 Recent Works

A significant analysis and discussion on the subject was presented by Martinazzi (2019). He studied the evolution of Lightning Network's topology during the first year by its introduction and he showed that the Network present a diassortativity. Diassortativity indicates that nodes with lower degree usually open new channels with higher degrees nodes.

This could potentially lead to the formation of hubs nodes. This is justified by the fact that it is expensive to open new channels, so for new nodes it is easier to connect to these hubs to have a large window of reachable nodes[3, 4].

Some of these hubs, according to Jian-Hong Lin et al.(2020)[6] have many connections, the largest one has 121 of degree, and is linked on the 34,2% of nodes of the Network.

Furthermore, to study the robustness of the Network, Martinazzi(2019) implements various attack strategies for the removal of important nodes based on topological centrality, like degree centrality (most specifically the strength of a node, determined as the weighted sum of all its edges), betweenness centrality and random node removal.

This analysis shows that Lightning Network is very resistant to random faultiness and the attack based on the weighted betweenness centrality is the most effective.

In fact, the removal of the best 50 nodes for this centrality decrease the size of the Largest Connected Component of the network by more the 35% against about 20% of Strength and Eigenvector.

The evaluation of random failures is important because emulate the case in which a peer is offline or the shutdown of a channel between two nodes and they obviously cannot participate to the routing of payment[3, 4].

Elias Rohrer et al. (2019) studied the state of Lightning Network's topology and assess its resilience to random failures and target attack. They found that the Lightning Network's central point dominance (maximum betweenness centrality of all nodes) is of the same order of a scale-free graph, a graph characterized by a few nodes with high degree and many nodes having low degree with a degree distribution similar to a power law distribution where the fraction of nodes is $P(k) \sim k^{-\alpha}$ (with k degree and $\alpha$ between 2 and 3) suggesting that Lightning Network relies in few central nodes in order to process payments.

In fact, a peculiar factor that distinguishes a Scale-free networks is that a new node can choose with freedom its neighbors and normally prefers to create a connection with well-connected nodes, as demonstrated by the previously analysis carried out by Martinazzi (2019).

These networks are generally robust to random failures than a random graph, but they are disposed to targeted attacks. In this study is also showed that in comparison with a random network, the Lightning Network is more clustered showing the typical characteristic of Small-world Networks with nodes that tend to cluster and a high density of edges[5].

Similar results were found in Topological Analysis of Bitcoin's Lightning Network performed by I. A. Seres et al. (2019) for the diassortative property and the robustness of the network regarding random failure and targeted attacks.

Going into details, they show how the removal of the highest degree node fragments the graph of Lightning Network into 37 connected components and the removal of the 30 largest hubs causes the shredding into 424 components, among which most are singles nodes that are completely isolated from the network.

They also show that targeted attacks to the hubs greatly affect the length of shortest paths and the available liquidity of the network and this cause the increasing of the failed payments ratio, in fact the removal of 37 of these nodes decrease the available funds by more than 50% [7].

# Chapter 3

# Topological Analysis on Centrality

The following chapter is structured as follows: the section 3.1 will discuss the Lightning Network dataset where the information needed to understand the structure of the network will be taken, in section 3.2 will discuss the degree distribution of the Lightning Network and the composition of the graph as diameter, average shortest path length and central point dominance, articulation nodes and connected components, in section 3.3 will present the choices regarding the centrality measures taken, in the section 3.4 the algorithms used for node removal are introduced and finally in the section 3.5 the results obtained are discussed.

## 3.1 Dataset

From the `"channels_ln.csv"` file, the information needed to understand the topological structure of Lightning Network was extracted. The csv file consists of various fields among which the most important are the ids of the nodes connected by the channel and the total capacity of the channel itself.

From the snapshot of the network it can be seen as shown in Table 3.1 that Lightning Network is composed of 6,006 nodes connected together through a total of 30,457 channels. The lowest channel capacity is 1,100 satoshi or 0.000011 ₿ and maximum capacity is 500,000,000 satoshi or 5 ₿. The total capacity of the network is 104,055,781,879 satoshi or 1040.56 ₿ with an average capacity of 0.039 ₿.

Some nodes have multiple channels between them, for the study done for each of these situations an equivalent channel will be considered and its capacity the sum of the capacities of the channels between the two nodes. The creation of equivalent channels reduces the total number of channels by 3,357, or 11% of total channels. A total of 27,100 channels will be considered for analysis.

| Lightning Network | |
|---|---|
| Total Nodes | 6,006 |
| | |
| Channels: | |
| Total Channels | 30,457 |
| Channels after merging | 27,100 |
| Merged Channels | 3,357 (11%) |
| | |
| Max Capacity | 5.0 ₿ |
| Min Capacity | $1.1 \times 10^{-5}$ ₿ |
| | |
| Total Capacity | 1040.56 ₿ |
| Average Capacity | 0.038 ₿ |
| | |
| Diameter of LCC | 7 |
| Central Point Dominance | 0.15 |
| Average Shortest Path | 3.17 |

**Table 3.1:** Lightning Network Dataset

## 3.2 Topological features of the Lightning Network

Studying the composition of the network it is possible to deduce important data regarding the nodes and the connections between them including the distribution of the degree that gives indications on the number of connections of the various nodes.

In the figure 3.1 it is represented the distribution of the degree in Lightning Network in a log-log plot. The x-axis shows the degree of the nodes, and the ordinates show the number of nodes per degree. The figure shows that Lightning Network follows a power law distribution with a negative trend, so the network consist in few nodes with high degrees and many nodes with low degree that prefer to connect with first ones to partecipate at the Network, because it is less expensive than open many others channels.

The table 3.2 shows more precisely the number of nodes with a certain degree and the relative percentage of the network. It can be seen that 2,588 nodes in the network possesses only one channel and represent 43% of the total number of nodes. More than half of the nodes, 3,448 or 57% of the network has 2 or less channels. Considering the nodes with 5 or less channels we have 76% of the network with

4.573 nodes while the 90% of the network (5.394 nodes) is reached considering the nodes with 14 or less channels. The node that has the maximum number of connection is the node with ID 2 that has degree 1,185, and in the ranking of nodes' degree the seventh node has about half of first node's neighbors with 571, and only 283 the 5% of total nodes have a neighborhood of at least 30 nodes.

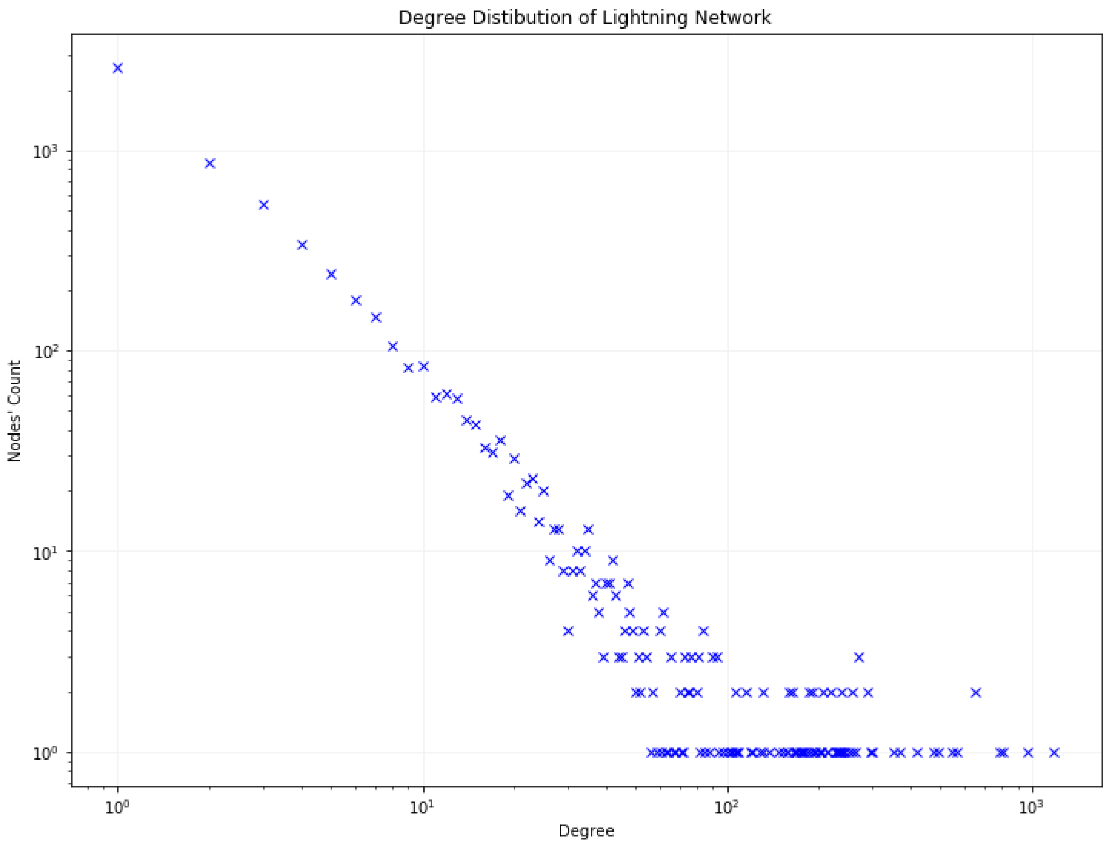



**Figure 3.1:** Degree distribution of Lightning Network

Other important network data can be found by studying the dataset such as the number of connected components, diameter, center point dominance, and average shortest paths.

**Connected components** are subgraphs of the network whereby nodes are reachable from each other. The snapshot of the network has 8 connected components, seven of which consist of only two nodes. For the purposes of this study, the 7 connected components composed of two nodes each represent two users who have created a channel between themselves but without participating in the collective network. For this reason they are irrelevant and will not be considered. The

| Lightning Network | | |
|---|---|---|
| | # of nodes | % |
| Total Nodes | 6,006 | |
| Nodes in the LCC | 5,992 | 99.8% |
| Nodes with 1 degree | 2,588 | 43% |
| Nodes with 2 degree or less | 3,448 | 57% |
| Nodes with 5 degree or less | 4,573 | 76% |
| Nodes with 14 degree or less | 5,394 | 90% |
| Nodes with 30 degree or more | 283 | 5% |

**Table 3.2:** Lightning Network Distribution

remaining 5,992 nodes in the network form the largest connected component (LCC) and as reported in the table 3.2 it is 99.8% of the Lightning Network.

**Diameter** in a network represents the maximum distance of short paths between two nodes. The diameter of the Lightning Network graph is 7. This measure is interesting in that it indicates the maximum number of hops a transaction would take to get to its destination. The path in question does not consider channel capacity and any fees but only the connections between nodes. The **average shortest path length** between nodes is 3.17.

**Central point Dominance** of the network is represented by the node that lies more often on the short paths among all the other nodes, that is the maximum betweenness centrality of the nodes of the network.

A value closely to one, indicate that almost all the payments pass through this node, therefore it would indicate a potential centralization of the network. The central point dominance in Lightning Network is 0.15 and is relative to the node with ID 2, the same node with the highest degree. Having a low value of central point dominance is preferable to avoid a few nodes having high control of the payment flow.

These three parameters will be analyzed after removing nodes in each centrality measure to see how they change in the network.

**Articulation Nodes** another important analysis is the search for the articulation points of the Lightning Network graph. An articulation point is a node that removed separates the graph into multiple parts. So removing them will form connected components. By analyzing the largest connected component there are 338 articulation nodes in Lightning Network.

| Articulation Nodes | | |
|---|---|---|
| Node ID | LCC | Connected Components (size) |
| 4254 | 5,889 | 1(3) 2(2) 95(1) |
| 5239 | 5,989 | 1(2) 0(1) |
| 928 | 5,989 | 1(2) 0(1) |
| 468 | 5,953 | 1(2) 36(1) |
| 350 | 5,977 | 1(2) 12(1) |
| 177 | 5,960 | 1(2) 29(1) |

**Table 3.3:** Articulation Nodes

It is interesting to understand which of these nodes upon removal create connected components composed of multiple nodes and not just isolated nodes, and to see if they turn out to be among the central nodes in subsequent analyses.

Most of the articulation nodes (332) form only isolated nodes. The remaining 6 nodes are shown in the table 3.3 showing the composition of the remaining largest connected component upon removal of the articulation node and the connected components that are created. Five of them, create only one connected component composed of a couple of nodes, and the other one break the network in 4 but two of the connected components are maded up by only 2 nodes and the last one by 3 nodes.

## 3.3 Centrality Measures

An undirected graph will be used to represent Lightning Network since it is impossible to know the state of the two balances between the two edges of a channel.

The centrality measures that can be used only for direct graphs presented in the previous chapter or the in and out versions of degree centrality, local reaching centrality and trophic levels will be discarded. In addition, the analysis is carried out on single nodes so all versions of group centrality will not be considered.

As in the past work the main measures of centrality will be used, such as degree centrality, eigenvector centrality, closeness centrality and betweenness centrality. Other measures that have not been used previously will be tested on the network as the second order centrality, load centrality, the current flow betweenness centrality and Vote rank.

To represent Lightning Network and study the centrality measures will be used two different graphs based on the use or not of the weight attribute of the edges. The two graphs will be called binary network and weighted network.

In the `binary network` the weight of the edges connecting two nodes will not be considered, if there is a channel between two nodes the corresponding edge attribute is 1, 0 otherwise. The binary network considers only the links between nodes. This graph will be used on all analyzed measures.

In the `weighted network` instead there are two different edge attributes according to the measure of centrality with which the network is analyzed. The capacity of the channel will be used in degree centrality and eigenvector centrality, while for closeness centrality, betweenness centrality and its current flow version will be used the reciprocal of the capacity.

This graph will not be used in the second order centrality and vote rank because they do not accept weighted graphs for their calculation.

The weighted network will also be used to analyze the network using mixed versions of the centrality measures presented in the previous chapter.

The $\alpha$ paramenter will be tuned by degree centrality and used the same value for closeness centrality and betweenness centrality.

## 3.4 Algorithm for Central Nodes Removal

For each version of the network and for each centrality measure, several tests were performed, the main ones being one with elimination of the list of central nodes in one iteration (One iteration), and one with iterative computation and removal of central nodes (Iterated).

There are also different approaches based on the centrality measure considered. The various approaches used on the various centrality measures will be explained below.

**One iteration**: One iteration means one calculation of the centrality measure. For each measure, the centrality of all nodes in the network is calculated and a list of central nodes is created. Then the nodes of the list are removed from the graph only if they are present in the largest connected component created by the previous removal.

If the central node considered was an isolated node or belonged to a different connected component than the current largest, it would not be removed. This will be done until the number of nodes within the largest connected component is less than half the number of nodes in the original network, or 3,003. The One iteration will be performed on all the centrality measures taken into account, both binary and weighted.

**Iterated**: in the Iterated version of the algorithm, the centrality measure was recalculated at each iteration. Therefore every time a node was removed the largest connected component was found again and the centrality on it was recalculated.

The Iterate was used on the four main measures in both the binary network and the weighted network and load centrality. The Iterate algorithm has not been used on the Second order and the current flow betweenness centrality because they required too much time to carry out the calculations.

Since Vote Rank uses a specific algorithm so using the Iterated removal would give identical results to those obtained from the degree centrality iterated on the binary network.

**Vote Rank Iteration**: with One iteration vote rank is not able to succeed in halving the largest connected component, and the Iterative is identical to degree centrality. For this reason, two different iterations were done. The first one recalculating the centrality on the remaining network, the second one finding a number of removed central nodes so increasing again the voting ability of the nodes that had voted the previous central nodes after a certain number of iterations. This hyperparament was tuned on the smallest number of central nodes needed to halve the largest connected component and was found 49. After 49 removals therefore the centrality was recalculated. These two methods for vote rank are called 2-Iteration and 49-Iteration.

**Mixed analysis**: a mixed analysis was also performed that considers both the binary and weighted measure. This was done by tuning the hyperparameter $\alpha$ which indicates whether to give more importance to the binary measure or the weighted measure.

The hyperparameter has been tuned on the degree centrality Iterated. It were used for the other centrality measures for which the mixed distance was defined, i.e. betweenness centrality and closeness centrality.

The choice of the $\alpha$ parameter fell on the value for which the necessary number of central nodes is minimized. In the case two or more values gave the same number of nodes, it has been chosen the value of $\alpha$ closer to zero, giving therefore more importance to the number of connections.

For each test, the topology of the network was studied. Specifically, the following topological characteristics were measured: the number of connected components and isolated nodes, the total capacity of the network, and the size of the largest connected component. Other data that were considered in the results were network diameter, average short path length, and central point dominance.

The size of the largest connected component was used as a limit to removal to see how many central nodes are needed to halve the number of nodes in the network.

Finally, nodes that were important for all centrality measures were found and the same analysis was performed. A random removal of nodes was also performed to test the resilience and behavior of the network to random attacks or offline nodes.

## 3.5 Results

This section will discuss the main results obtained from the various centrality measures shown in the table 3.4.

The table 3.4 shows the main results obtained after the various removals of the central nodes. For each measure the number of central nodes to be removed to halve the largest connected component, the composition of the remaining part of the nodes detached from the LCC in connected components of size 2 nodes, 3 nodes and other sizes as appropriate and the percentage of isolated nodes compared to the number of initial nodes.

| Centrality Measure | Nodes Removed | Connected Components Sizes | | | | Channels | | | | | Diam | Central Point Dom | Avg Short Paths |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 2 | 3 | other | 1 | Tot | Merged | Max | Tot ₿ | Avg | | | |
| $\text{Deg}_{(Bin,1)}$ | 80 | 37 | 6 | 1(11) | 47% | 7,890 | 7,306 | 3.00 ₿ | 164.52 ₿ | 0.023 ₿ | 12 | 0.05 | 4.39 |
| $\text{Deg}_{(Bin,iter)}$ | 75 | 37 | 6 | 1(11) | 47% | 8,273 | 7,672 | 3.00 ₿ | 173.50 ₿ | 0.023 ₿ | 12 | 0.06 | 4.31 |
| $\text{Deg}_{(W,1)}$ | 162 | 50 | 5 | | 47% | 6,100 | 5,634 | 0.20 ₿ | 52.53 ₿ | 0.009 ₿ | 12 | 0.24 | 4.60 |
| $\text{Deg}_{(W,iter)}$ | 113 | 47 | 5 | | 47% | 6,808 | 6,323 | 0.25 ₿ | 71.29 ₿ | 0.011 ₿ | 11 | 0.12 | 4.54 |
| $\text{Deg}_{(\alpha=0.2,iter)}$ | 74 | 39 | 6 | 1(11) | 47% | 8,438 | 7,792 | 3.00 ₿ | 158.39 ₿ | 0.020 ₿ | 12 | 0.08 | 4.26 |
| $\text{Eig}_{(Bin,1)}$ | 187 | 52 | 8 | 1(4) 1(6) | 47% | 5,241 | 4,807 | 3.00 ₿ | 84.03 ₿ | 0.017 ₿ | 13 | 0.18 | 5.12 |
| $\text{Eig}_{(Bin,iter)}$ | 101 | 46 | 5 | | 47% | 6,710 | 6,224 | 3.00 ₿ | 109.53 ₿ | 0.018 ₿ | 13 | 0.12 | 4.71 |
| $\text{Eig}_{(W,1)}$ | 342 | 62 | 7 | 1(4) 1(6) | 42% | 5,418 | 4,934 | 0.59 ₿ | 40.26 ₿ | 0.008 ₿ | 12 | 0.28 | 4.49 |
| $\text{Eig}_{(W,iter)}$ | 144 | 47 | 5 | | 46% | 6,779 | 6,178 | 0.25 ₿ | 55.85 ₿ | 0.009 ₿ | 11 | 0.20 | 4.44 |
| $\text{Bet}_{(Bin,1)}$ | 71 | 34 | 4 | 1(11) | 47% | 9,469 | 8,674 | 3.00 ₿ | 239.33 ₿ | 0.028 ₿ | 12 | 0.07 | 4.05 |
| $\text{Bet}_{(Bin,iter)}$ | 76 | 38 | 6 | 1(11) | 47% | 8,272 | 7,624 | 3.00 ₿ | 189.60 ₿ | 0.024 ₿ | 12 | 0.05 | 4.32 |
| $\text{Bet}_{(W,1)}$ | 83 | 37 | 4 | 1(11) | 47% | 9,114 | 8,412 | 0.59 ₿ | 130.06 ₿ | 0.015 ₿ | 12 | 0.15 | 4.01 |
| $\text{Bet}_{(W,iter)}$ | 97 | 42 | 5 | 1(11) | 46% | 7,784 | 7,103 | 0.59 ₿ | 90.95 ₿ | 0.013 ₿ | 12 | 0.15 | 4.28 |
| $\text{Bet}_{(\alpha=0.2,1)}$ | 75 | 37 | 4 | 1(11) | 47% | 9,186 | 8,469 | 3.00 ₿ | 162.60 ₿ | 0.019 ₿ | 12 | 0.10 | 4.07 |
| $\text{Bet}_{(\alpha=0.2,iter)}$ | 79 | 40 | 6 | 1(11) | 47% | 8,244 | 7,592 | 3.00 ₿ | 140.61 ₿ | 0.019 ₿ | 12 | 0.09 | 4.27 |
| $\text{Clo}_{(Bin,1)}$ | 296 | 50 | 5 | 2(4) 1(5) 1(6) | 43% | 5,095 | 4,676 | 3.00 ₿ | 75.25 ₿ | 0.016 ₿ | 13 | 0.32 | 4.96 |
| $\text{Clo}_{(Bin,iter)}$ | 113 | 45 | 4 | 1(6) | 46% | 6,536 | 6,023 | 3.00 ₿ | 106.11 ₿ | 0.018 ₿ | 13 | 0.12 | 4.81 |
| $\text{Clo}_{(W,1)}$ | 294 | 61 | 9 | 1(4) | 42% | 5,582 | 5,108 | 0.20 ₿ | 41.68 ₿ | 0.008 ₿ | 11 | 0.28 | 4.42 |
| $\text{Clo}_{(W,iter)}$ | 155 | 49 | 6 | | 47% | 5,891 | 5,440 | 0.51 ₿ | 54.95 ₿ | 0.010 ₿ | 14 | 0.14 | 4.79 |
| $\text{Clo}_{(\alpha=0.2,1)}$ | 221 | 58 | 6 | | 44% | 5,609 | 5,169 | 0.25 ₿ | 46.26 ₿ | 0.009 ₿ | 12 | 0.29 | 4.63 |
| $\text{Clo}_{(\alpha=0.2,iter)}$ | 121 | 48 | 4 | | 46% | 6,394 | 5,911 | 3.00 ₿ | 78.06 ₿ | 0.013 ₿ | 11 | 0.15 | 4.72 |
| $\text{VRank}_{(iter)}$ | 75 | 37 | 6 | 1(11) | 47% | 8,273 | 7,672 | 3.00 ₿ | 173.50 ₿ | 0.023 ₿ | 12 | 0.06 | 4.31 |
| $\text{VRank}_{(2-iter)}$ | 420 | 30 | 4 | | 42% | 12,413 | 11,336 | 3.00 ₿ | 409.37 ₿ | 0.036 ₿ | 9 | 0.06 | 3.38 |
| $\text{VRank}_{(iter,49)}$ | 73 | 37 | 4 | 1(11) | 47% | 9,721 | 8,965 | 3.00 ₿ | 232.24 ₿ | 0.025 ₿ | 12 | 0.08 | 4.00 |
| $\text{Load}_{(W,1)}$ | 83 | 37 | 4 | 1(11) | 47% | 9,114 | 8,412 | 0.59 ₿ | 130.06 ₿ | 0.015 ₿ | 12 | 0.15 | 4.01 |
| $\text{Load}_{(W,iter)}$ | 97 | 42 | 5 | 1(11) | 46% | 7,784 | 7,103 | 0.59 ₿ | 90.95 ₿ | 0.013 ₿ | 12 | 0.15 | 4.28 |
| $\text{Bet}_{(flow)}$ | 72 | 32 | 5 | 1(11) 1(14) 1(15) | 47% | 10,054 | 9,226 | 5.00 ₿ | 336.51 ₿ | 0.036 ₿ | 12 | 0.09 | 3.92 |
| SecondOrder | 124 | 52 | 4 | 1(6) | 46% | 6,103 | 5,616 | 3.00 ₿ | 97.52 ₿ | 0.017 ₿ | 13 | 0.16 | 4.85 |
| Random | 2,107 | 4 | | | 15% | 11,436 | 10,143 | 5.00 ₿ | 452.82 ₿ | 0.045 ₿ | 8 | 0.23 | 3.17 |

**Table 3.4:** Centrality Measures Results

Total channels of the remaining network are also shown with the number of equivalent channels, maximum, minimum and average capacity per channel. The last three columns show the network diameter, the average shortest path length, and the central point of dominance of the network. In general, the network diameter averaged over the various measures increased to 12. Most tests on the measures result in the total number of channels being reduced by one-third after removing the center nodes.

| Nodes with one degree isolated from the Network | | | | | |
|---|---|---|---|---|---|
| Measure | Isolated | Deg(1) | Deg(1) & Iso | % Deg(1) | % Isolated |
| $\text{Deg}_{(Bin,1)}$ | 2,848 | 2,588 | 2,017 | 78% | 71% |
| $\text{Deg}_{(Bin,iter)}$ | 2,816 | 2,588 | 2,010 | 78% | 71% |
| $\text{Deg}_{(W,1)}$ | 2,812 | 2,588 | 1,855 | 72% | 66% |
| $\text{Deg}_{(W,iter)}$ | 2,808 | 2,588 | 1,917 | 74% | 68% |
| $\text{Deg}_{(\alpha=0.2,iter)}$ | 2,815 | 2,588 | 2,013 | 78% | 72% |
| $\text{Eig}_{(Bin,1)}$ | 2,843 | 2,588 | 1,785 | 69% | 63% |
| $\text{Eig}_{(Bin,iter)}$ | 2,828 | 2,588 | 1,918 | 74% | 68% |
| $\text{Eig}_{(W,1)}$ | 2,496 | 2,588 | 1,463 | 57% | 59% |
| $\text{Eig}_{(W,iter)}$ | 2,737 | 2,588 | 1,833 | 71% | 67% |
| $\text{Bet}_{(Bin,1)}$ | 2,845 | 2,588 | 2,084 | 81% | 73% |
| $\text{Bet}_{(Bin,iter)}$ | 2,843 | 2,588 | 2,026 | 78% | 71% |
| $\text{Bet}_{(W,1)}$ | 2,832 | 2,588 | 2,066 | 80% | 73% |
| $\text{Bet}_{(W,iter)}$ | 2,792 | 2,588 | 1,965 | 76% | 70% |
| $\text{Bet}_{(\alpha=0.2,1)}$ | 2,820 | 2,588 | 2,061 | 80% | 73% |
| $\text{Bet}_{(\alpha=0.2,iter)}$ | 2,811 | 2,588 | 2,013 | 78% | 72% |
| $\text{Clo}_{(Bin,1)}$ | 2,566 | 2,588 | 1,489 | 58% | 58% |
| $\text{Clo}_{(Bin,iter)}$ | 2,773 | 2,588 | 1,858 | 72% | 67% |
| $\text{Clo}_{(W,1)}$ | 2,542 | 2,588 | 1,489 | 58% | 59% |
| $\text{Clo}_{(W,iter)}$ | 2,815 | 2,588 | 1,877 | 73% | 67% |
| $\text{Clo}_{(\alpha=0.2,1)}$ | 2,657 | 2,588 | 1,623 | 63% | 61% |
| $\text{Clo}_{(\alpha=0.2,iter)}$ | 2,760 | 2,588 | 1,861 | 72% | 67% |
| $\text{VRank}_{(iter)}$ | 2,816 | 2,588 | 2,010 | 78% | 71% |
| $\text{VRank}_{(2-iter)}$ | 2,537 | 2,588 | 2,000 | 77% | 79% |
| $\text{VRank}_{(iter,49)}$ | 2,833 | 2,588 | 2,072 | 80% | 73% |
| $\text{Load}_{(W,1)}$ | 2,832 | 2,588 | 2,066 | 80% | 73% |
| $\text{Load}_{(W,iter)}$ | 2,792 | 2,588 | 1,965 | 76% | 70% |
| $\text{Bet}_{(flow)}$ | 2,802 | 2,588 | 2,054 | 79% | 73% |
| SecondOrder | 2,778 | 2,588 | 1,778 | 69% | 64% |
| Random | 876 | 2,588 | 774 | 30% | 88% |
| elitenodes | 1,630 | 2,588 | 1,104 | 43% | 68% |

**Table 3.5:** Isolated Nodes with one degree after removal

An important study was also to understand which nodes result isolated from the network after the various removals after the various tests, specifically if there were many nodes with degree one among them.

The table 3.5 shows the number of isolated nodes in the network in the various tests and specifically those with degree one. The first column shows the number of isolated nodes, the second column shows the number of degree one nodes in the network, and the third column shows the number of nodes that had degree one and remained isolated from the network after central node removals. The last two columns show the percentage of isolated nodes with degree one compared to the total number of nodes with degree one, and the percentage of isolated nodes with degree one compared to the total number of isolated nodes.

From the percentages it can be deduced that most of the nodes isolated after the removal of the central nodes in the various tests were nodes with only one channel.

In fact, in all measures, the results show that more than half of the isolated nodes are nodes with degree one, confirming the tendency of these nodes to connect with central nodes to communicate with the network.

This might indicate that transactions between the remaining nodes in the network could still be performed without too many problems since most of the isolated nodes were nodes with one degree.

**Degree Centrality Binary Network**: with One Iteration, the removal of the 1$^{st}$ best node in this case caused the isolation of 68 nodes from the network. After the removal of the 4$^{th}$ best node the number of individual nodes increased up to 534 and there was also a formation of a first connected component consisting by a pair of nodes.

The number of isolated nodes increased up to 1,197 after the removal of the 11$^{th}$ best node and the number of connected components disjointed by the largest connected component was about 11 composed by 2 nodes each one.

At the removal of the 50$^{th}$ node the largest connected component was the 60% of the original size with 3,580 nodes, the other nodes were divided into 26 connected components consisting of 2 nodes and 2 connected components consisting of 3 nodes.

At the end of the analysis the original LCC was dismembered into 37 connected components consisting of 2 nodes, 6 of 3 nodes and one of 11 nodes and 2,848 isolated nodes while the remaining 2,961 constituted the new LCC after removal of 80 nodes.

Similar results were obtained in the case of the iterative removal of nodes. In fact, for the first 24 nodes removed the results were exactly the same, as the nodes removed in the two scenarios were the same.

Despite this, it performed better overall after removing the 50$^{th}$ node, managing to dismantle the LCC in half with only 75 nodes. Both results showed a very large reduction in the number of network nodes with few nodes removed, with the 47% of the nodes were completely isolated and the 2% of the LCC is scattered in small connected components.

There was also an increase in the diameter and average shortest path of the network. The most distant nodes, in order to perform a transaction, had to perform 12 hops to reach the most distant node in the best case, and the average of these hops increased slightly from 3.17 to 4.39 and 4.31, respectively. The network also had a reduction in central point dominance from 0.16 to 0.05 and 0.06 respectively indicating a lower centrality.

The total capacity of the network had been reduced to 16% of the initial amount of 1040.56 ₿ with a total of 164.52 ₿ and 173.50 ₿. If all nodes in the network were removed except for those 75 or 80 central nodes chosen by the algorithm, the total capacity would be about double 325 ₿, and 345 ₿ respectively. This showed that these nodes had some control over the network, and had many important channels, especially between them.

**Degree Centrality Weighted Network**: After removal of the $51^{\text{th}}$ strongest node both the One Iteration and the Iterated reduced the LCC size to 80% and 70% of the original size, respectively, with 1,130 individual nodes and 7 connected components formed by 2 nodes for the One Iteration versus 1,676 individual nodes and 19 connected components of size 2 for the Iterated removal.

To complete the task, the two measures needed the removal of 162 and the parts detached from the LCC were a total of 55 connected components including 5 of size 3 nodes and 2,812 disconnected nodes for the One Iteration, while the Iterated needed the removal 113 stronger nodes with 2,808 isolated nodes and 52 connected components including 5 of size 3.

The diameter increased in both cases, slightly more in the One Iteration to 12 than in the Iterated to 11. There was a larger increase in the average shortest paths length compared to the binary network calculations, with a value of 4.60 with one iteration and 4.54 in the iterated. In the one iteration case there was also an increase in network center point dominance from 0.15 to 0.24 while in the Iterated it was 0.12.

The total funds within the network after the removal of these nodes decreased significantly, up to 5% of the initial value in One Iteration. Also in the weighted network, the Iterated perform slightly better.

**Degree Mixed Network**: after the two previous methods, it is interesting to try to mix them and use the hybrid measure provided by Opsahl et al. and the formula that they proposed.

The tuning of the $\alpha$ parameter was performed with various values between 0 and 1 inclusive, and choose the value of $\alpha$ which minimizes the number of central nodes needed to halve the LCC. Since the parameter could be even greater than

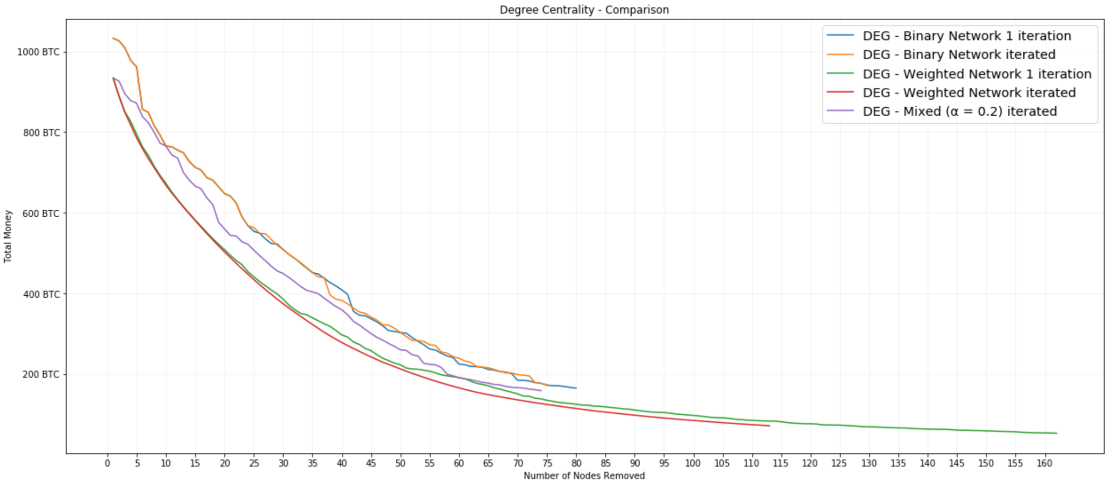



**Figure 3.2:** Degree Centrality: Total amount of ₿

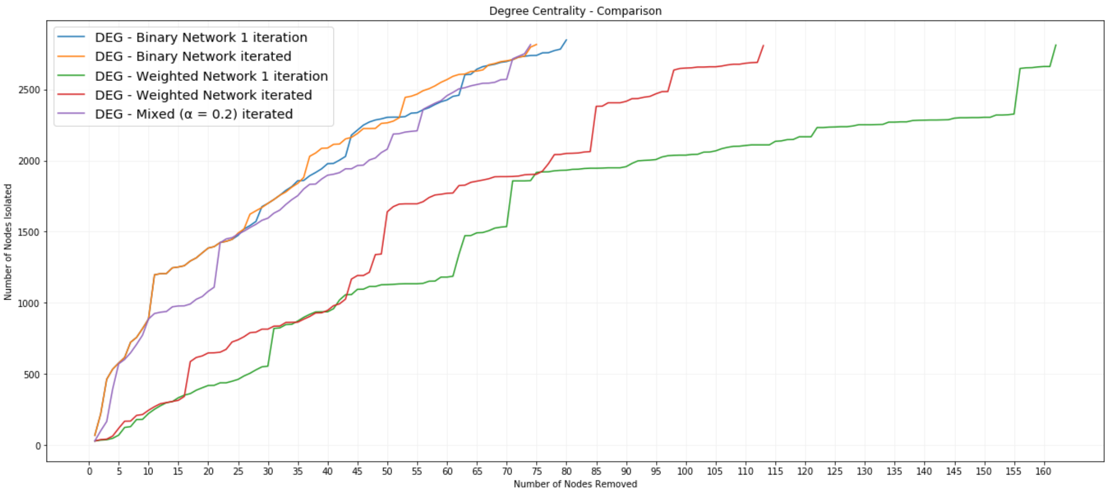



**Figure 3.3:** Degree Centrality: Isolated Nodes

one, also values greater than 1 was tried. For $\alpha > 1.0$ the number of central nodes needed for the task increased, so this could indicate that strongest nodes with few connection were not necessarily the best nodes for this task. Because of this the choice of the $\alpha$ parameter fell on a value between 0 and 1. It was found that the value that satisfies these conditions is for $\alpha = 0.20$ with 74 nodes for complete the task.

Since many values of $\alpha$ gave the same result, the lowest one was chosen, this is to give more importance to the number of ties than to their strength. The results were very similar to those of the Iterated case of the binary network, but due to the consideration of the weight parameter the total funds decreased to 158.39 ₿ compared to 173.50 ₿ of the Iterated.

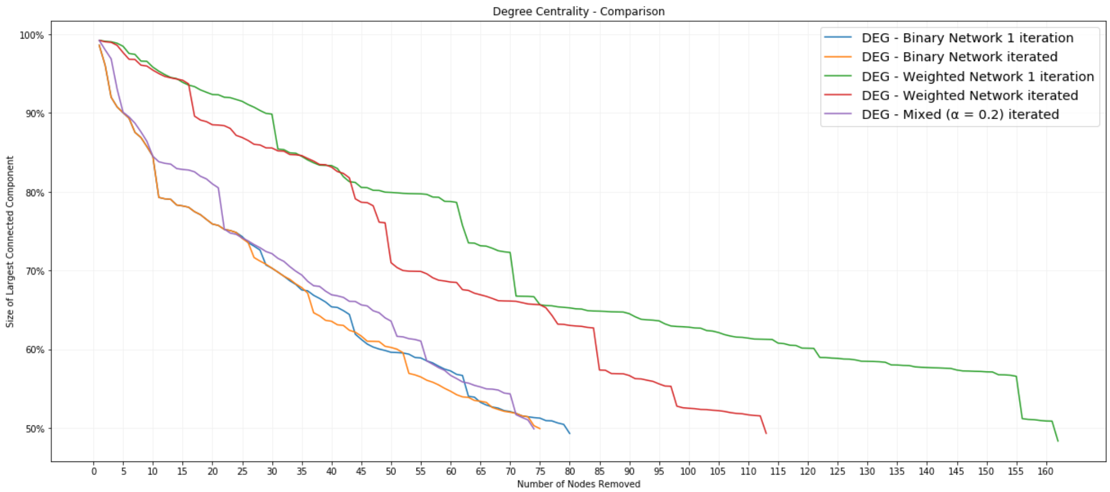



**Figure 3.4:** Degree Centrality: Size of LCC

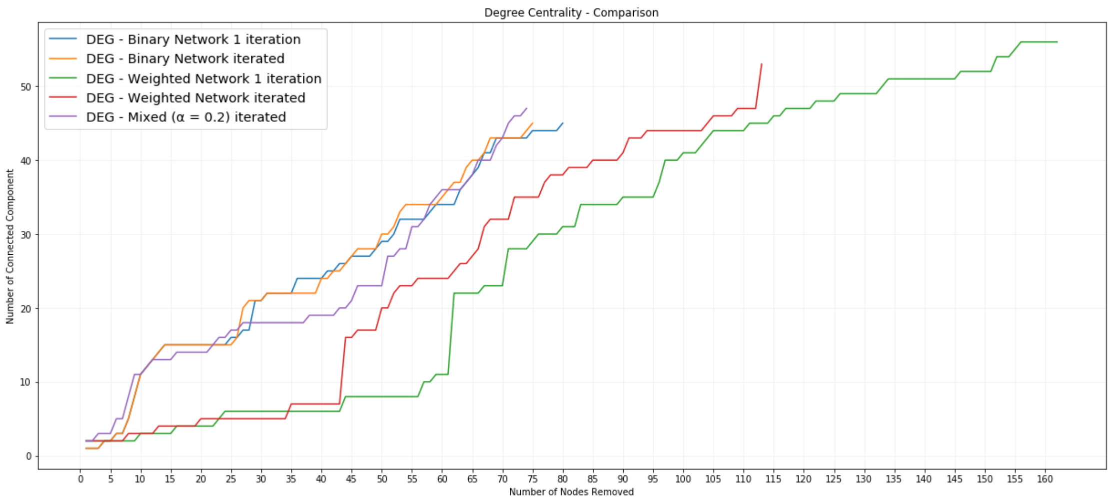



**Figure 3.5:** Degree Centrality: number of Connected Components

The figures represent the trend of the network at the removal of the various nodes according to the degree centralities taken into consideration: in figure 3.2 the total number of Bitcoins present in the remaining network, in figure 3.3 the number of isolated nodes accumulated at the removals, the size of the largest connected component in figure 3.4 and the number of connected components that were created in figure 3.5.

**Eigenvector Centrality Binary Network**: the One Iteration test with eigenvector centrality needed 187 central nodes to complete the task, while the Iterated needed 101 central nodes.

With the remotion of central nodes the LCC separated into 52 connected components consisting of two nodes, 8 of 3 nodes, 1 of 4 and 1 of 6 and 2,483 isolated nodes in the One Iteration test. On the other hand, in the Iterated test the LCC was divided into 46 connected components of 2 nodes, 5 of 3 nodes and 2.828 isolated nodes.

The total capacity of the network was reduced to 84.03 ₿ in the One Iterated and 109.53 ₿ in the Iterated with a similar average per channel of 0.017 and 0.018 ₿ respectively.

The diameter was almost doubled with 13 hops to connect the most distant nodes in both tests. The average length of the shortest paths in the One Iteration case was 5.12 hops, the largest of all the tests performed, while 4.71 in the Iterated case. The central point of dominance increases in the One Iteration to 0.18 while in the Iterated test the central point of dominance decreases to 0.12.

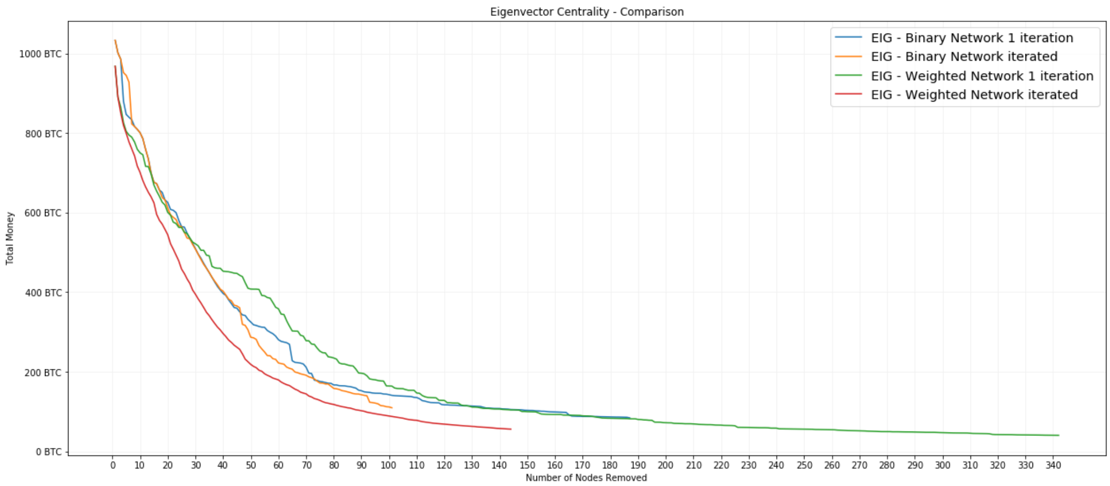



**Figure 3.6:** Eigenvector Centrality: Total amount of ₿

**Eigenvector Centrality Weighted Network**: in the One Iteration needed 342 central nodes to halve the LCC and 144 in the Iterated. The connected components formed were 62 of size 2 nodes, 7 of size 3, 1 of size 4 and 1 of size 6 and a total of 2,496 isolated nodes in the One Iteration, while in the Iterated there were 47 connected components of size 2, 5 of size 3 and 2,737 isolated nodes.

The total capacity of the network in both tests was reduced to about 5% in both tests with 40.26 ₿ in the One Iteration test and 55.85 ₿ in the Iterated.

The diameter increased to 12 in the One Iteration while in the Iterated case it

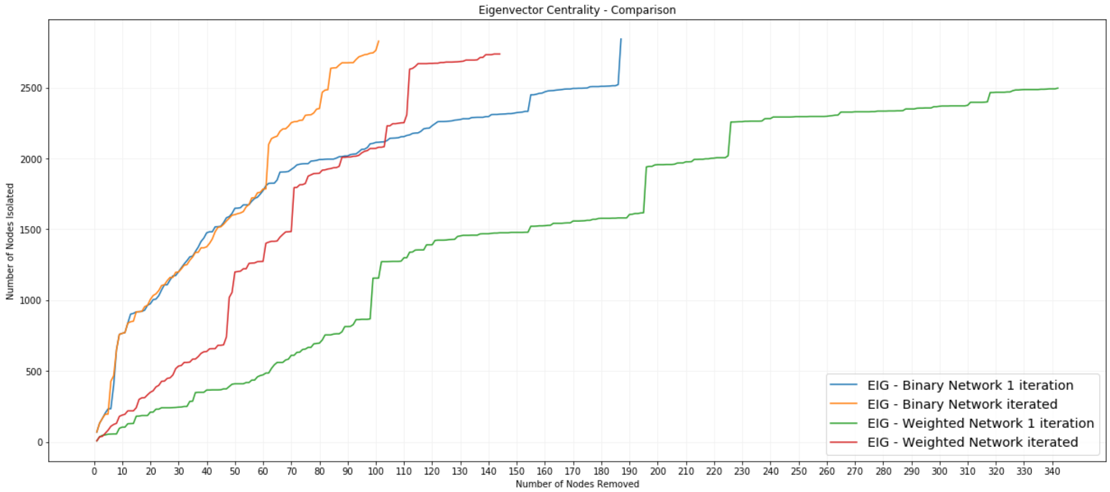



**Figure 3.7:** Eigenvector Centrality: Isolated Nodes

increased to 11. The mean length of the shortest paths increased to 4.49 in the One Iteration and 4.44 in the Iterated. The central point of dominance increased to 0.28 and 0.20 respectively.

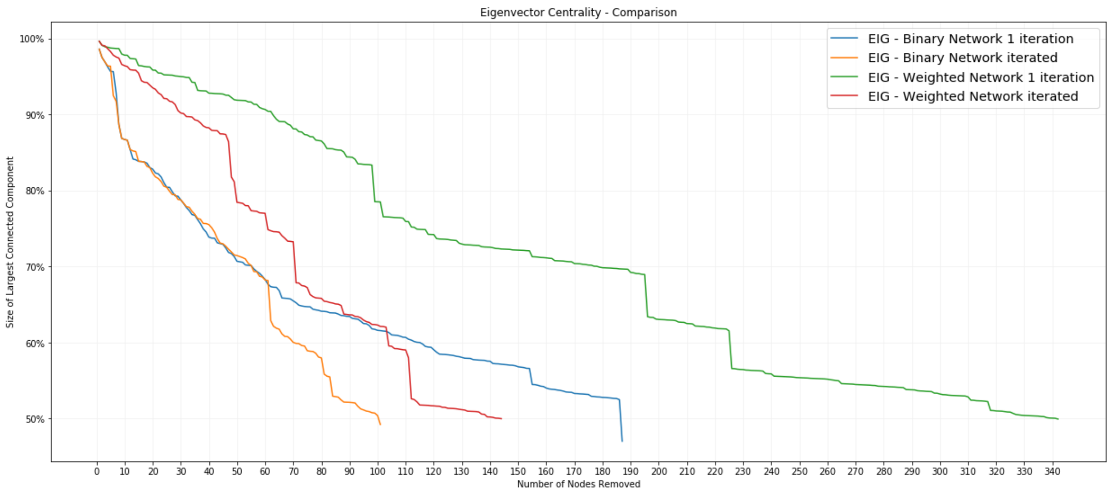



**Figure 3.8:** Eigenvector Centrality: Size of LCC

The figures represent the trend of the network at the removal of the various nodes according to the eigenvector centralities taken into consideration: in figure 3.6 the total number of Bitcoins present in the remaining network, in figure 3.7 the number of isolated nodes accumulated at the removals, the size of the largest connected

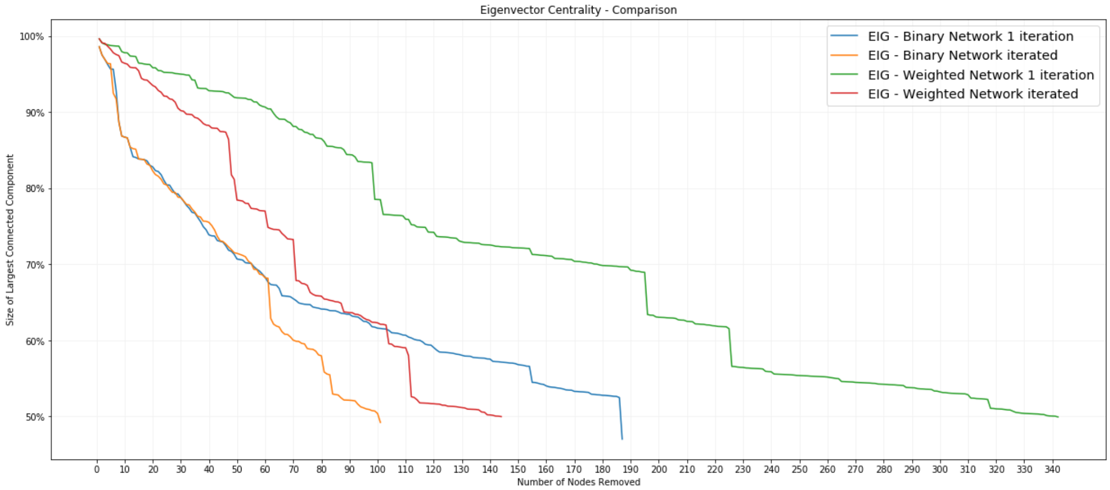



**Figure 3.9:** Eigenvector Centrality: number of Connected Components

component in figure 3.8 and the number of connected components that were created in figure 3.9.

**Betweenness Centrality Binary Network**: for the attack based on the betweenness centrality on the binary network both the measures performed very well. They needed few of 80 nodes to halved the LCC.

Only the betweenness centrality needed more nodes in the Iterated, 76 central nodes respect to the One Iteration, 71 nodes. The betweenness centrality One Iteration in the binary network needed the lowest number of central nodes for achieved the task of halved the LCC.

By removing central nodes in the binary network One Iteration were obtained 34 connected components composed of 2 nodes, 4 composed of 3 nodes and one of 11 nodes with 2,845 isolated nodes. Similar results were obtained in the Iterated network with an additional 4 connected components composed of 2 nodes and 2 of 3 nodes and 2,843 isolated nodes.

The total network capacity decreases from 1040.56 ₿ to 239.33 ₿ in One Iteration and 189.60 ₿ in Iterated with average capacity per channel of 0.028 ₿ and 0.024 ₿, respectively.

The diameter in both cases increased from 7 to 12 and the average short path length also increased from 3.17 to 4.05 in the One Iteration and slightly more in the Iterated at 4.32.

**Betweenness Centrality Weighted Network**: in the weighted network, the two iterative methods needed more removed central nodes than the binary network case to complete the task, with 83 and 97 respectively.

The LCC breaks down into 37 connected components of size 2 nodes, 4 of size 3 nodes, and 1 from 11 nodes while there are 2,832 isolated nodes in the One Iteration.

In the Iterated instead there are 5 connected components made by 2 nodes and 1 by three nodes, more than in the previous test and a total of 2,792 isolated nodes.

The maximum channel capacity is decreased to 0.59 ₿ in both tests, there is also a decrease in the total amount of funds at 130.06 ₿ for the One Iteration and 90.95 ₿ for the Iterated.

The center point dominance resulted unchanged respect to the original network with a value of 0.15 in both cases and there was a small increase in the average length of the shortest paths at 4.01 for One Iteration and slightly higher for Iterated with 4.28.

**Betweenness Centrality Mixed Network**:the analysis on the mixed network has been made with the alpha calculated on the degree centrality and both the One Iteration and the Iterated have been made.

Both tests needed less central nodes to remove to complete the task than the same tests on the weighted network but more nodes than the same tests on the binary network, more precisely 75 for the One Iteration and 79 for the Iterated.

The LCC was divided into 37 components of 2 nodes, 4 of 3 and one of eleven with 2,820 nodes isolated in the One Iteration and 40 components of 2, 6 of 3 and 1 of 11 and 2,811 nodes isolated in the Iterated.

The diameter had grown to 12 in both cases and the central point of dominance was very similar between the two tests with 0.09 and 0.10 respectively. The results obtained were intermediate between the tests performed on the previous networks.

**Load Centrality**: the tests performed on the two networks gave exactly the same results as the respective tests performed on betweenness centrality and the nodes removed were exactly the same, which is why it is no longer considered in the rest of the analysis.

**Betweenness Current Flow Centrality**: for the current flow betweenness only the One Iteration test was done on the weighted network. This is because it would require a lot of computational time to run the other tests as well.

This test needed 72 central nodes to fulfill the task. From the LCC, 32 connected components consisting of 2 nodes, 5 of 3, one of 11, one of 14 and one of 15 and 2802 isolated nodes were detached. The current flow betweenness centrality was the test that split into more connected components of different sizes throughout the analysis.

The total capacity of the remaining network was 336.51 ₿ more than double the weighted network test of normal betweenness centrality this was also due to the larger connected components that formed.

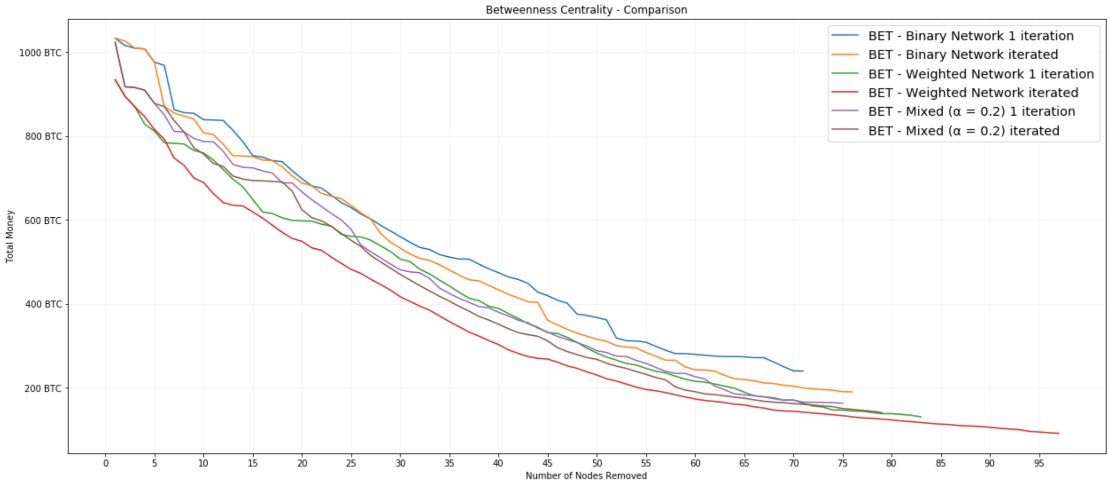



**Figure 3.10:** Betweenness Centrality: Total amount of ₿

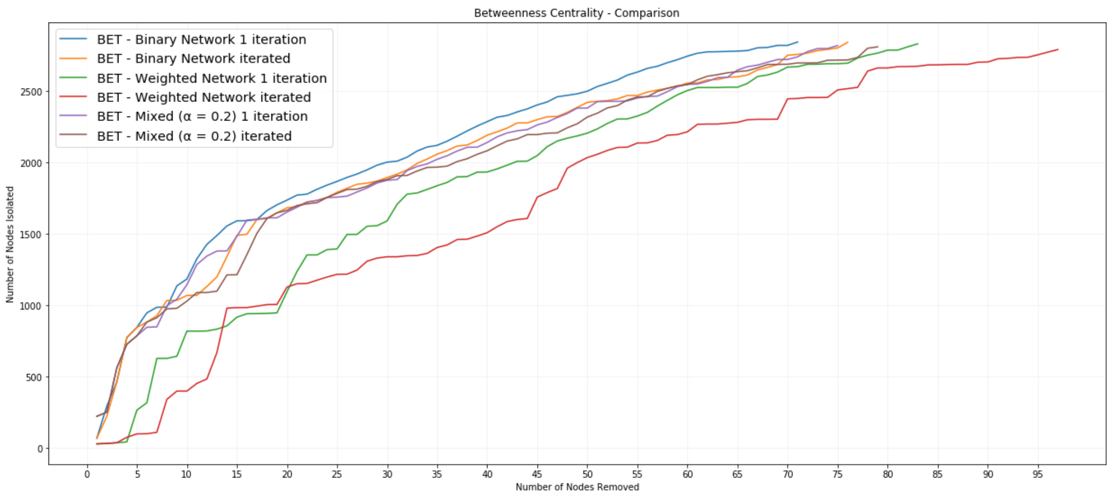



**Figure 3.11:** Betweenness Centrality: Isolated Nodes

The diameter also rises in this test to 12 hops and the average length of the shortest paths to 3.92 while the central point of dominance drops to 0.09.

The figures represent the trend of the network at the removal of the various nodes according to the betweenness centralities taken into consideration: in figure 3.10 the total number of Bitcoins present in the remaining network, in figure 3.11 the number of isolated nodes accumulated at the removals, the size of the largest connected component in figure 3.12 and the number of connected components that were created in figure 3.13.

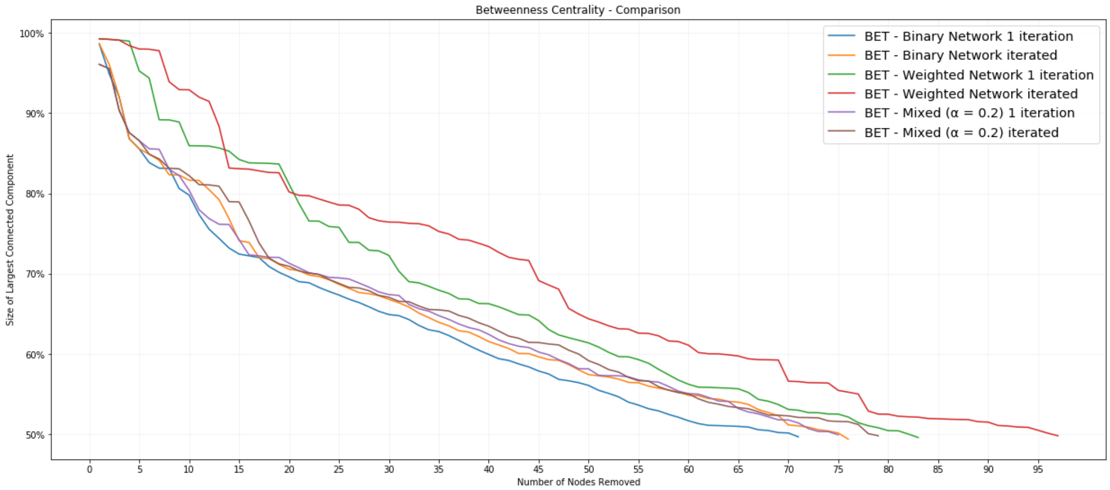



**Figure 3.12:** Betweenness Centrality: Size of LCC

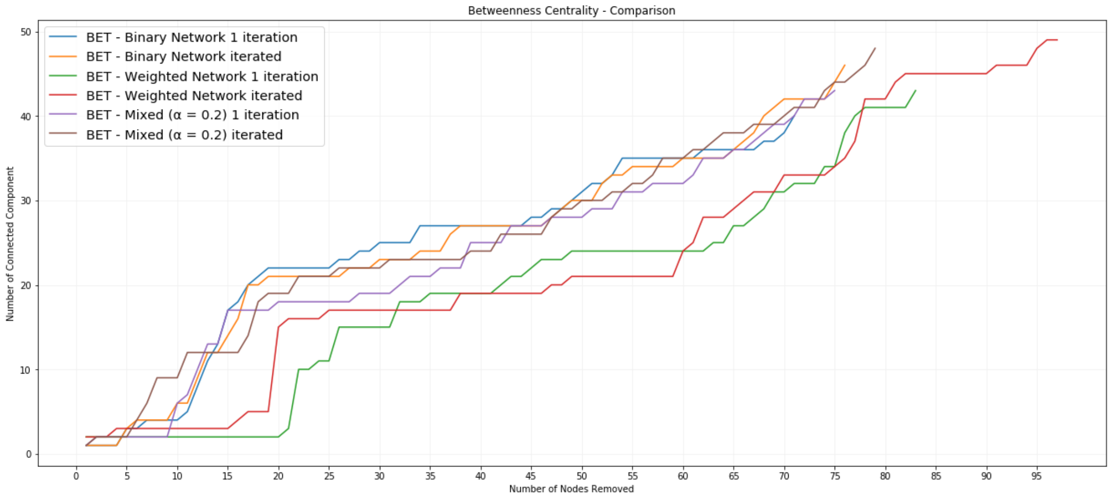



**Figure 3.13:** Betweenness Centrality: number of Connected Components

**Closeness Centrality Binary Network**: The two tests in the binary network in closeness centrality showed similar results for removing the top 50 nodes, with slightly better performance in Iterated. In the end to halve the LCC, the One Iteration needed 296 central node removals with 50 connected components of size 2, 5 of size 3, 2 of size 4, 1 of size 5, and 1 of size 6 with 2,566 isolated nodes. Iterated required 113 central node removals with 45 connected components of size 2, 4 of size 3 and 1 of size 6 and 2,773 isolated nodes.

The Closeness centrality One Iteration needed at least 33 nodes to reduce the LCC by 20% of the original size, three times the number of nodes that degree centrality needed in both tests in the binary network.

There was also a decrease in the total amount of funds, with a total of 75.25 ₿ in One Iteration and 106.11 ₿ in Iterated.

There is an increase in diameter, 13 in both cases, and average shortest path with a slightly higher value in the One Iteration case with 4.96 in contrast to 4.81 in the Iterated. There was an increase in center point dominance with 4.96 in One Iteration and 4.81 in Iterated.

**Closeness Centrality Weighted Network**: the One Iteration needed the removal of 294 central nodes to halve the LCC by breaking it down into 61 connected components of size 2, 9 of size 3, 1 of size 4, and 2.542 isolated nodes. The Iterated on the other hand needed 155 central nodes to be removed forming 49 connected components of size 2, 6 of size 3, and 2,542 isolated nodes.

In the analysis performed in the weighted network the closeness centrality is one of the measures that needs more nodes to reduce the LCC of the 10% with 50 central nodes. The network capacity was reduced below 6% of the total in both cases by 41.68 ₿ and 54.95 ₿ respectively.

The diameter of the net after the One Iteration test had increased to 11 while for the Iterated to 14 hops, the highest result among all tests of all measures. The central point of dominance doubled from the original point of dominance in One Iteration while it remained unchanged in Iterated.

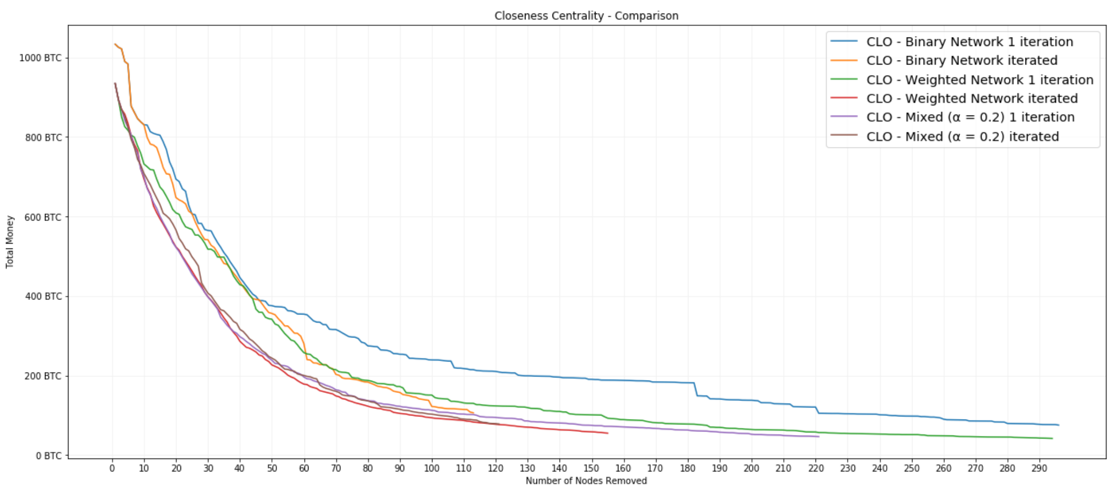



**Figure 3.14:** Closeness Centrality: Total amount of ₿

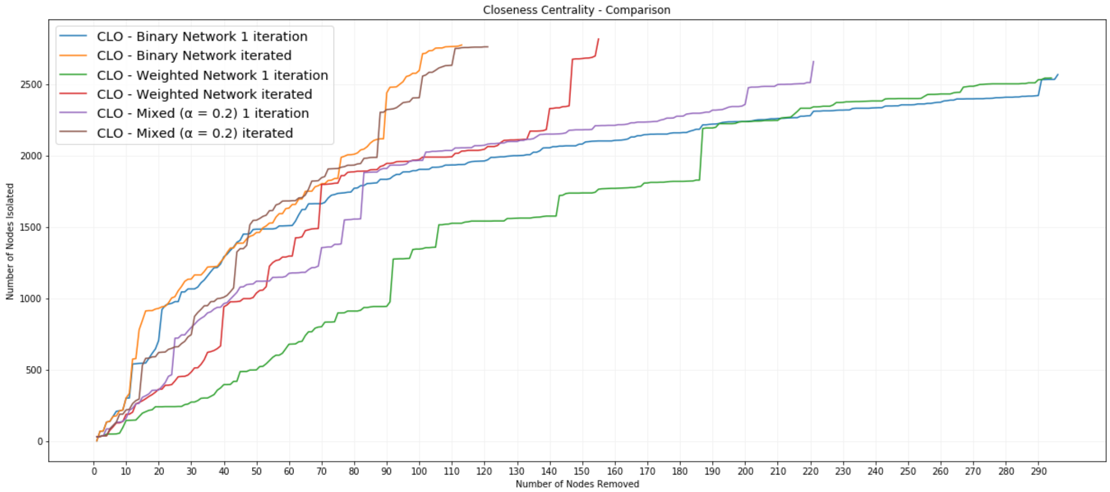



**Figure 3.15:** Closeness Centrality: Isolated Nodes

**Closeness Centrality Mixed Network**: the One iteration in the mixed network performed better in terms of nodes to be removed, it needed 221 central nodes removed, 70 nodes less than the two respective One Iterations in the other 2 networks. Considering the Iterated it needed 121 central nodes removed, 34 nodes less than the Iterated on the weighted network but 8 nodes more than the binary one.

In the One Iteration from the removal of the central nodes 58 connected components of size 2 nodes, 6 of size 3 and 2,657 isolated nodes were formed from the LCC. In the Iterated one, 48 connected components of size 2 and 4 of size 3 and 2,760 isolated nodes were formed.

In both tests it also gave intermediate results regarding the diameter respectively 12 for the One Iteration 11 for the Iterated with average length of the shortest paths respectively 4.63 and 4.72. For the central point of dominance the results are both similar to their counterparts calculated on the weighted network respectively 0.29 and 0.15.

The figures represent the trend of the network at the removal of the various nodes according to the closeness centralities taken into consideration: in figure 3.14 the total number of Bitcoins present in the remaining network, in figure 3.15 the number of isolated nodes accumulated at the removals, the size of the largest connected component in figure 3.16 and the number of connected components that were created in figure 3.17.

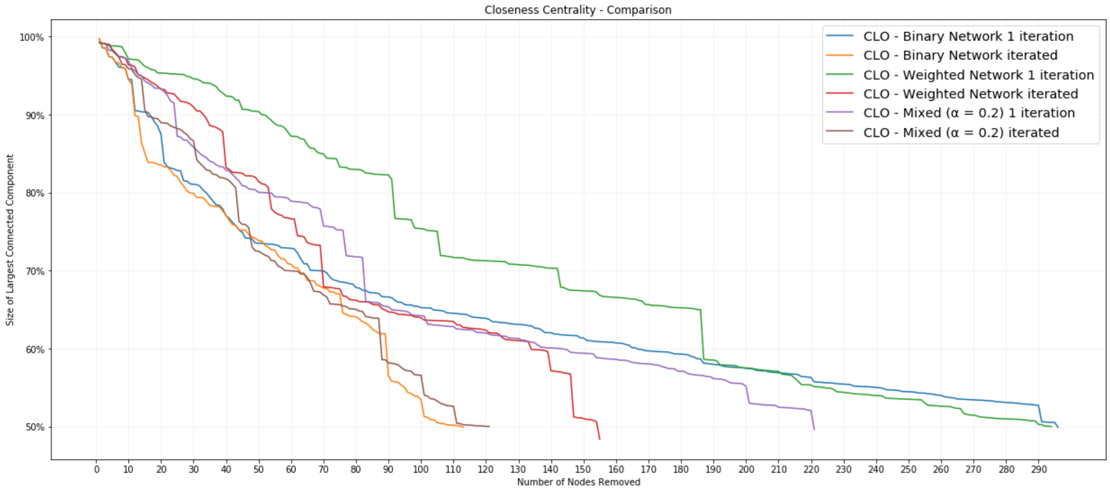



**Figure 3.16:** Closeness Centrality: Size of LCC

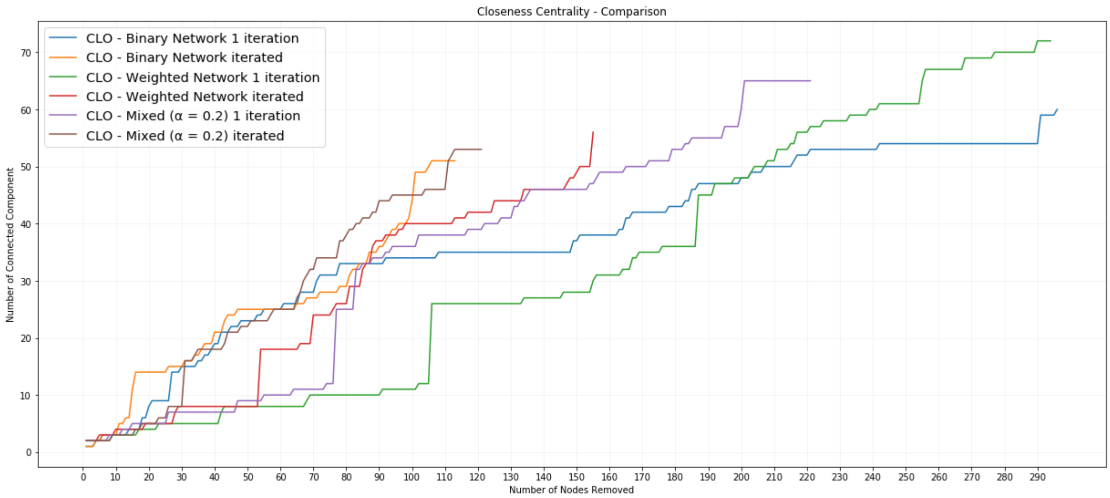



**Figure 3.17:** Closeness Centrality: number of Connected Components

**Vote Rank**: for Vote Rank, nodes would vote for the best node in its neighborhood, the node receiving the most votes was out for subsequent votes and the voting ability of its neighbors decreased, iteratively.

The final list of central nodes for this method with one iteration consisted of 415 nodes, the remaining ones had a negative score according to the algorithm.

With the removal of all nodes in the list Vote Rank was not able to halve the LCC.

The algorithm performed well for the first 50 nodes, reducing the LCC with less nodes than the degree centrality, after 40 best nodes the LCC is 60% of the original size.

On the other hand, after 50 nodes, further node removals in the ranking did not perform as well as the previous ones. In fact, after removing all nodes in the ranking, the LCC was still not halved.

For the **2-Iteration** of Vote Rank once the first list of central nodes was calculated and removed, the second list was calculated. The Vote Rank recalculation required 5 nodes to complete the task of halving the LCC, with a total of 420 central nodes required.

This removal created 30 connected components consisting of 2 nodes, 4 of 3 nodes, and 2.537 isolated nodes. The total funds dropped to 409.37 ₿ and the average per channel to 0.036 ₿.

The diameter at the end was 9, only two jumps more than the original size with an average shortest path length of 3.38 and central point dominance of 0.06.

For Vote Rank with the Iterated method, the nodes that were chosen were exactly the same as the Iterated degree centrality for the binary network, as nodes that had more channels received more votes, and their neighbors restored their initial and reinstated voting capacity for the next vote, if they were not isolated nodes.

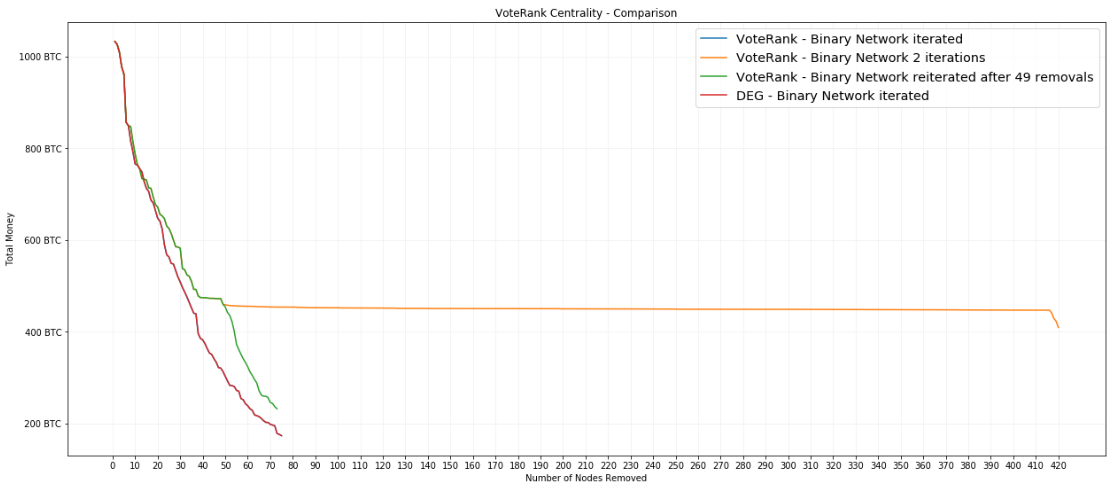



**Figure 3.18:** Vote Rank: Total amount of ₿

Recalculating the Vote Rank after **removing 49** nodes per list, the algorithm needed 73 nodes to complete the task. Removing these nodes created 37 connected composites consisting of 2 nodes, 4 of 3 and one of 11, and 2,833 isolated nodes. Total funds dropped to 232.24 ₿ and average per channel to 0.025 ₿. Network diameter increases to 12 with average short path length of 4.00 and central point of dominance dropped to 0.08.

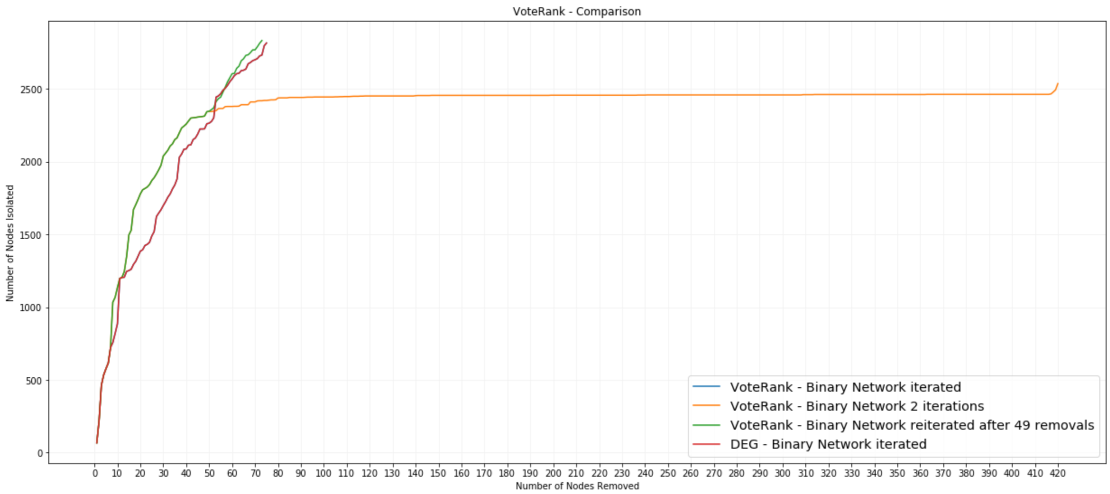



**Figure 3.19:** Vote Rank: Isolated Nodes

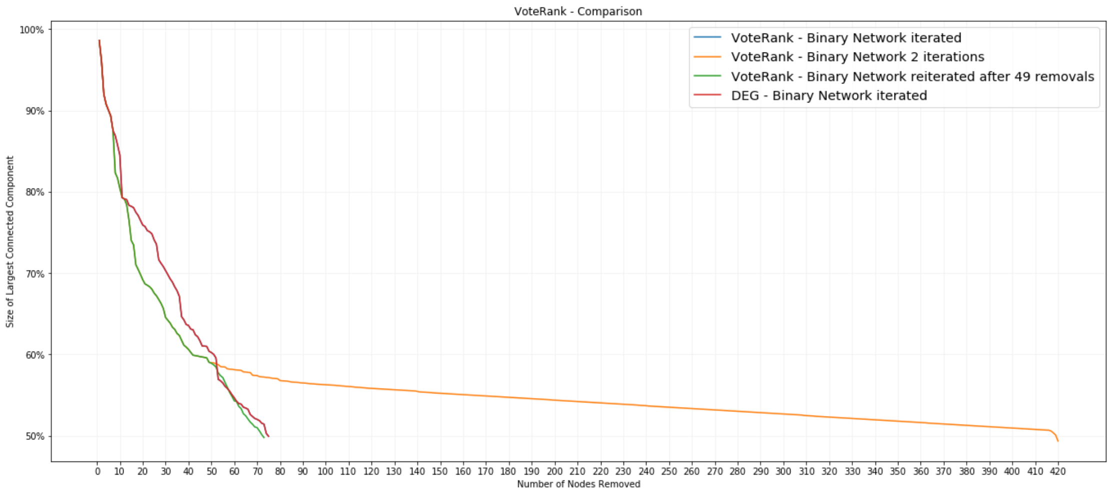



**Figure 3.20:** Vote Rank: Size of LCC

The figures represent the trend of the network at the removal of the various nodes according to iterated methods used with Vote Rank and the degree centrality Iterated in the binary network: in figure 3.18 the total number of Bitcoins present in the remaining network, in figure 3.19 the number of isolated nodes accumulated at the removals, the size of the largest connected component in figure 3.20 and the number of connected components that were created in figure 3.21.

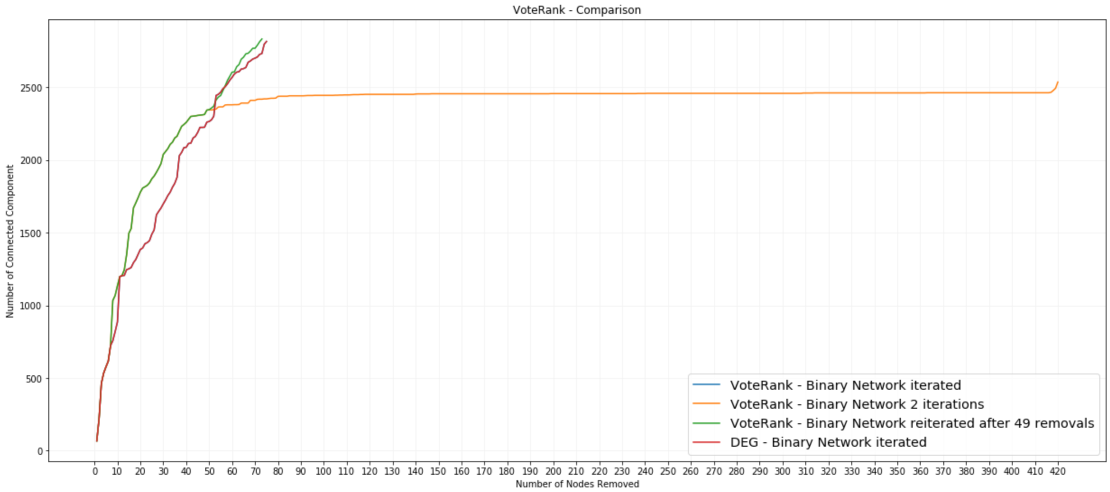



**Figure 3.21:** Vote Rank: number of Connected Components

**Second Order Centrality**: After removal of the best 25 nodes in this case the LCC was reduced by about 20%, and with 45 nodes at 70% of its original size. The task was completed with the removal of the 124$^{\text{th}}$ best node, with 56 connected components of size 2 and 3, 52 and 4 respectively, and 2,778 isolated nodes. The results were similar to other previous cases such as the Iterated cases of degree centrality and betweenness in the weighted network cases, and eigenvector centrality and closeness centrality in the binary network.

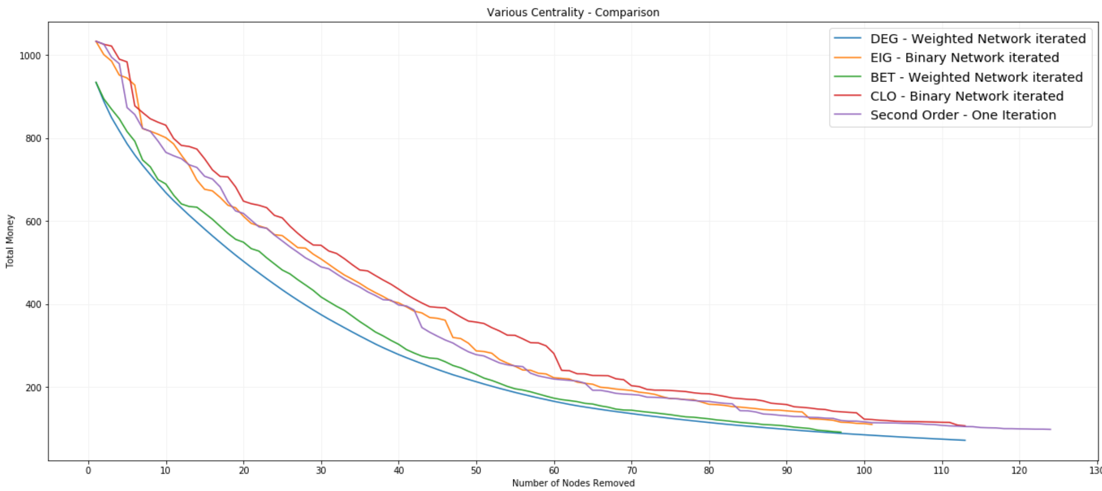



**Figure 3.22:** Second Order Centrality: Comparison total ₿

The total capacity of the network decreased to 97.52 ₿ with an average per

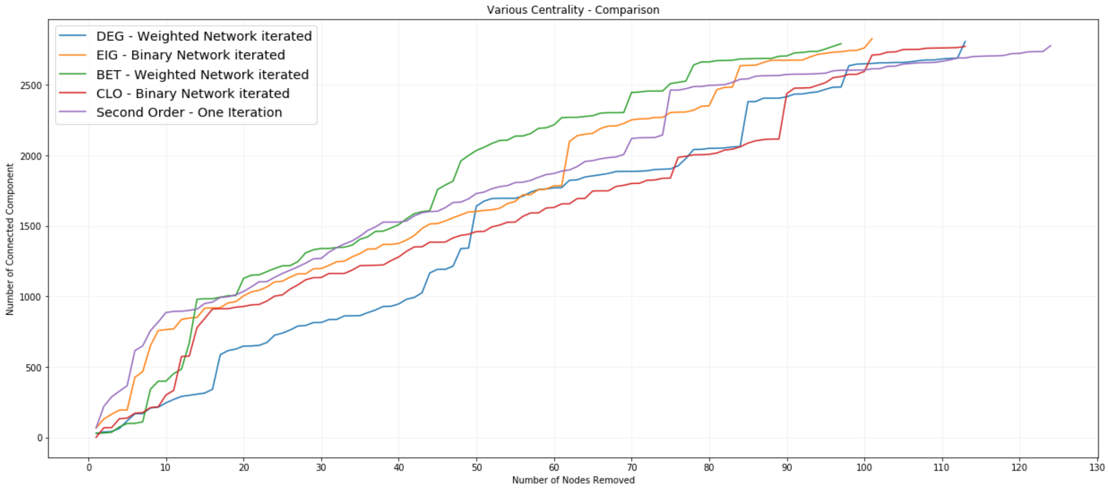



**Figure 3.23:** Second Order Centrality: Comparison isolated nodes

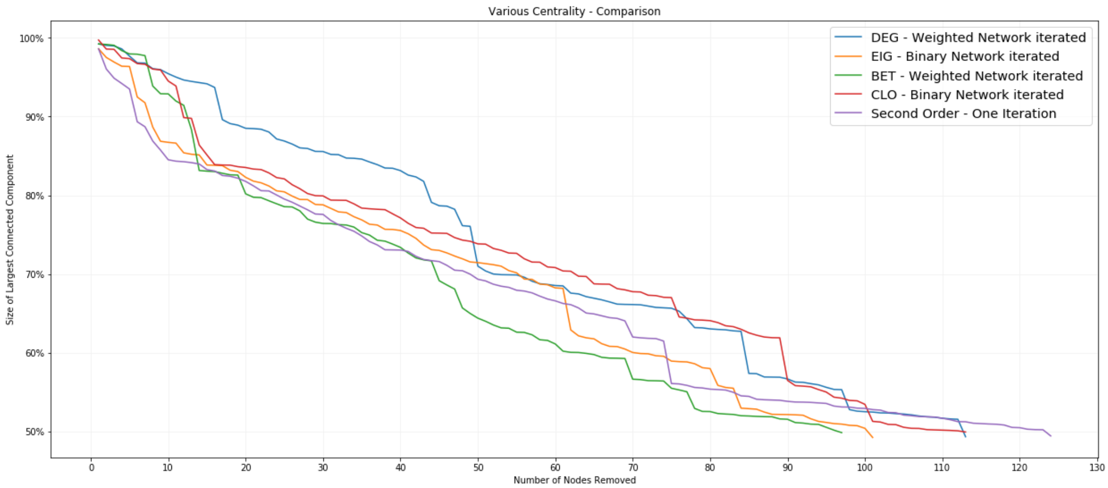



**Figure 3.24:** Second Order Centrality: Comparison LCC size

channel of 0.017 ₿.

The diameter increased to 13 hops, the average shortest path length increased to 4.85, and the center point dominance 0.16.

The figures represent the trend of the network at the removal of central nodes according to the Second Order and most similar trends like eigenvector centrality and closeness centrality Iterated in the binary network, degree centrality and betweenness centrality Iterated in the weighted network.

In figure 3.22 the total number of Bitcoins present in the remaining network, in figure 3.23 the number of isolated nodes accumulated at the removals, the size

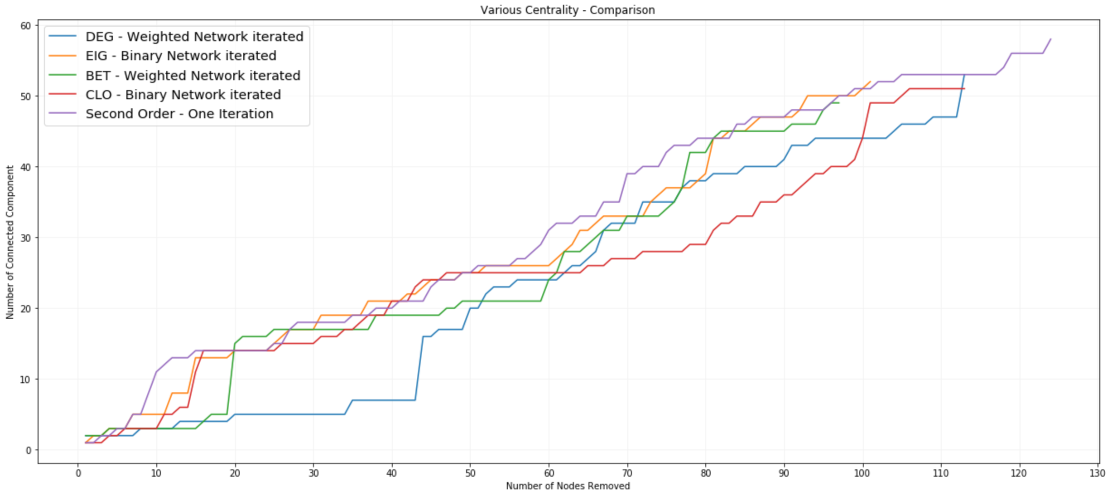



**Figure 3.25:** Second Order Centrality: Comparison connected components

of the largest connected component in figure 3.24 and the number of connected components that were created in figure 3.25.

**Random**: The network proved to be very robust in the analysis performed with random removal. In fact it takes about one third of the nodes in the total network to reduce the LCC to half.

Among the various tests with random removal, the best result was obtained after the removal of 2,107 nodes, the 35% of the total nodes of the network. Only four connected components consisting of 2 nodes and 876 isolated nodes were detached from the LCC. The removal of random nodes caused a linear drop in network nodes.

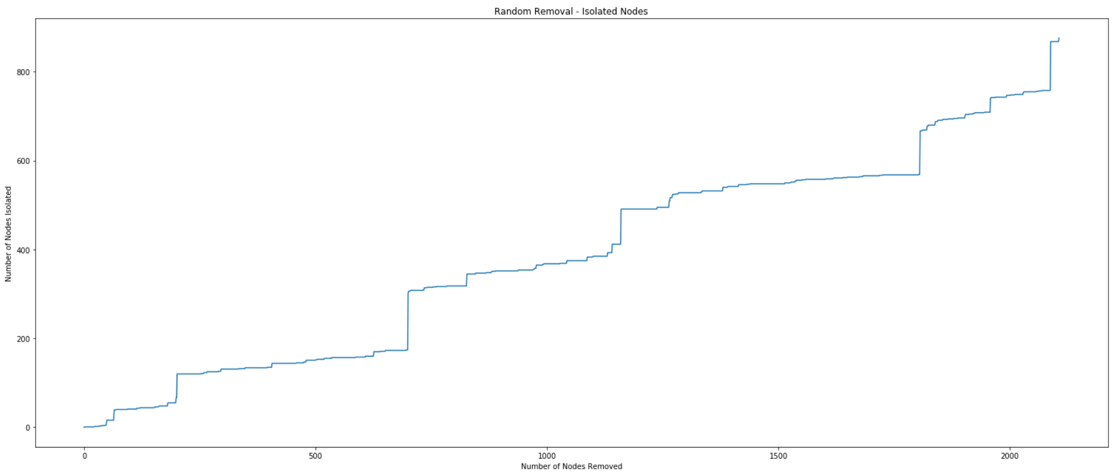



**Figure 3.26:** Random removal isolated nodes

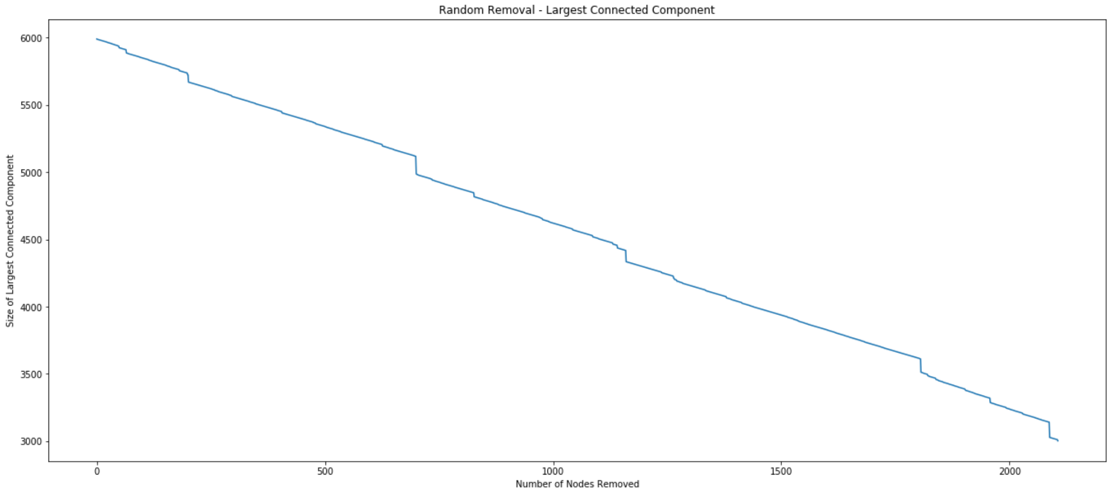



**Figure 3.27:** Random removal LCC size

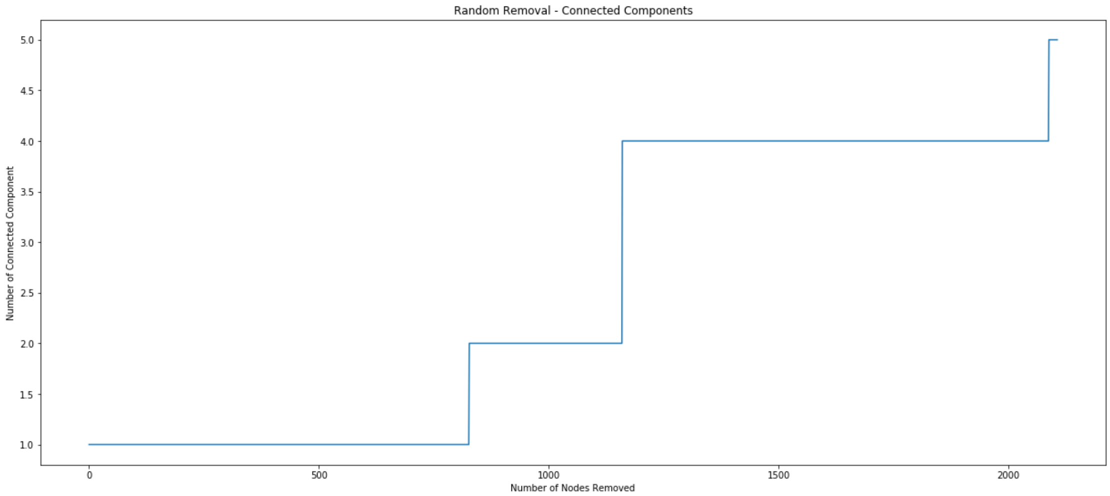



**Figure 3.28:** Random removal number of connected components

The network did not experience any noticeable increase in terms of the number of hops per transaction, and the total network funds dropped to 452.82 ₿ with average per channel of 0.045 ₿.

The diameter increased by only one hop from 7 to 8, while the average shortest path distance remained unchanged at 3.17 and the central point dominance increased to 0.23.

The figures represent the trend of the network at the random removal: in figure 3.26 the number of isolated nodes accumulated at the removals, the size of the largest connected component in figure 3.27 and the number of connected components that were created in figure 3.28.

### 3.5.1 Elite Nodes

At the end of all the analyses, all the centrality scores that were obtained from the various nodes on all the measures considered were collected.

An important study on the network was to find if there was any node that satisfied the condition of being a removed central node on all calculated measures. For simplicity of notation these nodes will be called Elite Nodes below.

| Elite Nodes Removal: Network Stat | |
|---|---|
| Nodes Removed | 41 |
| | |
| Total Nodes | 4,335 (72%) |
| | |
| Connected Components: (LCC) | |
| Size 2 | 19 |
| Isolated | 1,630 (27%) |
| | |
| Channels: | |
| Total Channels | 14,287 (47%) |
| Channels after merging | 13,013 (48%) |
| Merged Channels | 1,274 (9%) |
| | |
| Max Capacity | 5.0 ₿ |
| Min Capacity | $1.1 \times 10^{-5}$ ₿ |
| | |
| Total Capacity | 405.63 ₿ (39%) |
| Average Capacity | 0.031 ₿ |
| | |
| Diameter of LCC | 10 |
| Central Point Dominance | 0.17 |
| Average Shortest Path | 3.76 |

**Table 3.6:** Elite Node Results

This analysis showed that there were a total of 41 nodes that were central for all centrality measures. The node with ID 2 which was the node with the most connections in the network was well ranked in many of the centrality measures taken in the binary network. In contrast, it was central but on the low ladder in the rankings of the centrality measures taken in the weighted network. This denoted that the node had many connections and was also well connected but these connections resulted in not strong ties.

Other nodes of interest were the articulated nodes found earlier. Of these six nodes, only three were part of the Elite Nodes. In particular, the node with ID 177 was well ranked in every measure of centrality, both binary and weighted network. In contrast, the node that best disassembled the network into multiple connected components, the node with ID 4254, was not present among the Elite Nodes.

The table 3.6 shows the situation of the network after the removal of the Elite Nodes. The network presented itself with 72% of the total nodes of the original network, that is 4,335 nodes of which 19 components connected of size 2 nodes, 12 formed by the removal of the Elite Nodes. Of these nodes 52% had only one channel. The node with the highest degree has 415 neighbors, and only 164 nodes have more than 30 channels. The removal of the Elite Nodes resulted in the isolation of 1,630 nodes.

The total number of channels was almost halved, and the total capacity of the network is 405.63 ₿, 39% of the initial funds.

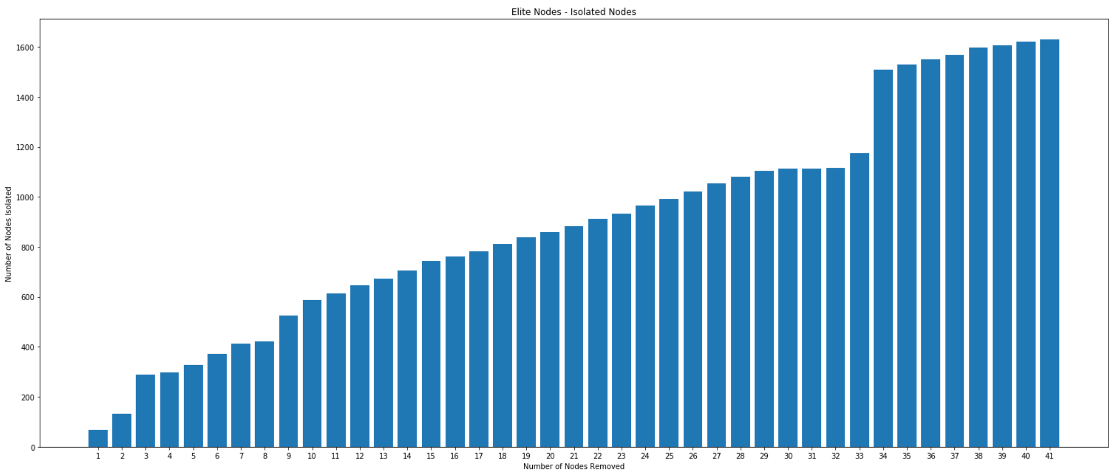



**Figure 3.29:** Elite Nodes isolated nodes

The diameter increased slightly with a maximum distance of 10 hops 3 more than the original network and an average shortest path distance increased to 3.76, the central point dominance is almost unchanged from 0.16 to 0.17.

Elite Nodes removal was shown to be more effective than random node removal. The network diameter and average shortest path lengths obtained from Elite Nodes removal were greater than those obtained from random removal. In addition, the total network bottom decreased more with the removal of Elite Nodes than with the 2,107 random nodes. This showed that some nodes may indeed have an influence

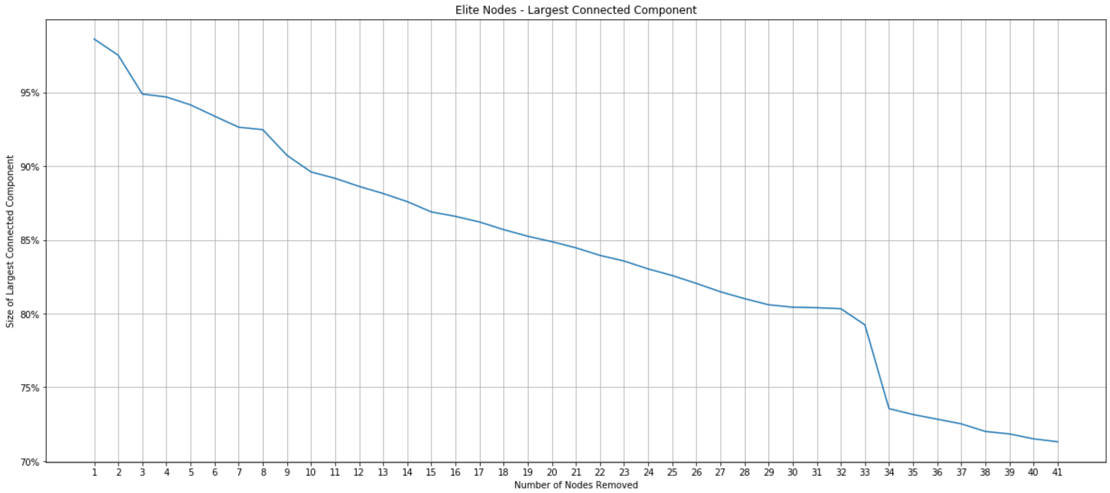



**Figure 3.30:** Elite Nodes LCC size

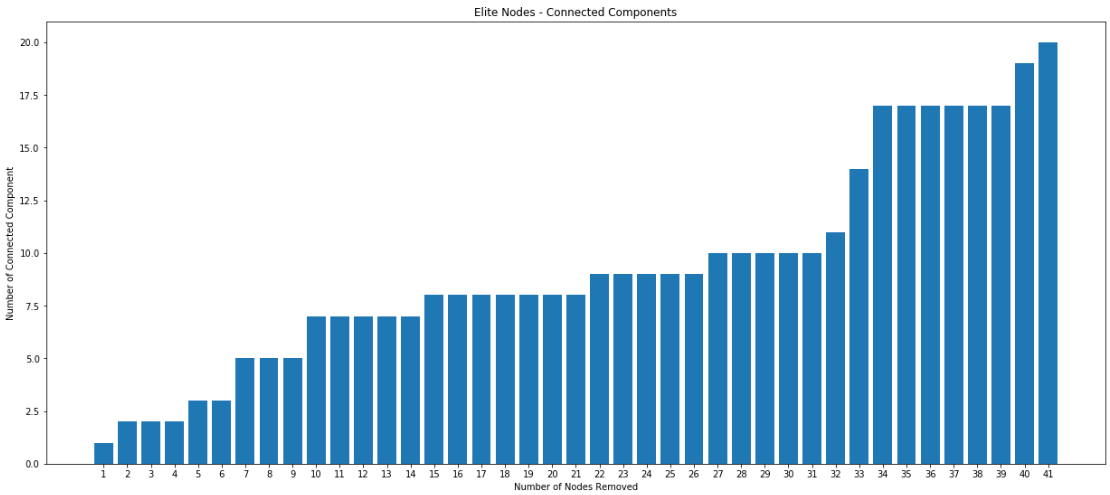



**Figure 3.31:** Elite Nodes number of connected components

on the network.

The figures represent the trend of the network at the removal of Elite Nodes: in figure 3.29 the number of isolated nodes accumulated at the removals, the size of the largest connected component in figure 3.30 and the number of connected components that were created in figure 3.31.

In the next chapter, payment simulations using CLoTH will be performed.

# Chapter 4

# Simulations on Centrality

The following will introduce CLoTH the Lightning Network simulator for simulating payments in a payment network. Section 4.1 will cover CLoTH in general and the functions used within it to simulate payments. Section 4.2 will discuss previous analyses on two networks of interest and results obtained. Finally the section 4.3 will present the analyses performed in this work and the results obtained.

## 4.1 CLoTH in general

CLoTH[8] is a Lightning Network simulator. It simulates payments on the Lightning Network and produces performance measures such as the probability of successful payments, the average time per payment and the average length of payment routes.

The functions implemented in CLoTH to perform payment simulations are based on those used to route HTLCs defined on `lnd`, one of the various Lightning Network implementations.

The simulator takes as input a definition of a payment network and a list of payments to be simulated. The network in the simulations is represented by three files that contain the data structure that defines the various element of the network: the channel file, the edge file, and the node file.

A channel is defined by the ids of the two nodes it connects, the ids of the edges that indicate the two directions of the channel, and the total capacity of the channel. An edge is defined by the channel id to which it belongs, the current balance in this edge, fixed fee for payments and a fee proportional to the value of the payment, the timelock of the HTLC, and the minimum value for a payment to be routed for this direction. The node is defined by its id.

These data structures are easily modified and adaptable to case studies such as removing the various central nodes of centrality measures.

Payments consist of the id of the sender node, id of the receiver node, the amount of the payment and the time in which it occurs in the simulator. Payments in the simulation will be randomly generated.

The following will briefly describe how simulations work on CLoTH. For more details the reader is referred to [8].

The simulator works to discrete events. These events changing the state make to advance the simulation of the execution of the payment. The simulated CLoTH functions are mapped to the HTLC messages according to the type of node that forwards them, thus performing different functions depending on the case explained briefly below.

The **`find_route()`** simulates a sender node that looks for a routing path for a new payment or a previously failed one to retry it. This function also records the payment statistics used to calculate the performance measures of the simulator. The blacklists of nodes and edges are updated by removing them if a timeout has expired.

At the end the modified Dijkstra is used to find the route. If the route is found a `send_payment` is called otherwise the payment fails.

The **`send_payment()`** simulates a sender nodes that checks if the forwarding edge is known or has the necessary balance to forward the payment. If at least one of these conditions is not met the edge is blacklisted and `find_route()` is called.

If both checks are verified the edge balance is scaled by the amount of the payment simulating the creation of an HTLC between the two nodes and a `forward_payment` event is created.

The **`forward_payment()`** simulates payment routing by intermediate nodes. It checks that the next node is cooperative before and after the HTLC is established. If the node is non-cooperative before the HTLC is established, a `receive_fail` event is scheduled, in the second case it waits for the timelock of the established HTLC to release the funds. The edge is blacklisted in both cases.

A check on the presence of the forward edge is done as in `send_payment`. If the check fails the function generates a `forward_fail` event to notify the previous hops of the failed payment up to the previous node to the sender which generates a `receive_fail`.

Another check is done on the edge balance as in `send_payment`. If the outcome is positive and the next node is an intermediary node another `forward_payment` event is created, otherwise a `receive_payment` is generated if the next node is the payment receiver.

The **`receive_payment()`** simulates the receipt of payment by the receiving node. It increments the balance of the opposite edge directed to the previous hop after checking for cooperativity. A `forward_success` event is generated at the end.

The **`forward_success()`** simulates the forwarding of a payment success by an intermediate hop. It checks the cooperativeness of the node with which an HTLC already exists, and if it is positive, the balance is reduced in the edge directed to the opposite node. If the next node is the sender it creates a `receive_success` event otherwise another `forward_success` if it is another intermediate hop.

The **`receive_success()`** simulates the receipt of a payment success by the sender. Payment statistics are recorded and it does not create any subsequent events as the payment is completed.

The **`forward_fail()`** simulates the receipt of a failed payment from an intermediate node. It restores the previous state of the balance before the payment attempt.

Generates a `forward_fail` or a `receive_fail` event depending on whether the next node is an intermediate node or the sender of the payment.

The **`receive_fail()`** simulates the receipt of a failed payment from the sender. Restores the previous state of the budget related to the HTLC between the sender and the intermediate node. Generates a `find_route` event to retry the payment.

After the discrete time events simulation, the simulator carries out one phase of post-processing where it collects the obtained data transforming them in measures of performance useful to fine statistical purposes.

The performance of the network will be given by the probability of a payment to be successful, the probability of failure due to the absence of budget, the probability of failure due to the lack of a path that connects the two nodes sender and receiver, the average time to complete a payment, the average number of attempts needed to payments and the average length of the path of payments.

## 4.2 Previous Simulations

Two analyses were done, one using a snapshot of the Lightning Network and testing various parameters by combining them. This analysis was used to understand what instances of network inoperability were present and what they were due to.

Another analysis was done on a synthetic network created using the parameters entered into the CLoTH simulator. This analysis is useful to understand the impact of the parameters entered into the simulator on the performance of the payment network.

**Lightning Network snapshot**: an initial analysis was performed via CLoTH on a snapshot of the Lightning Network mainnet from June 2018 (1,221 nodes and 5,167 channels with average capacity of 381,350 satoshis) using network composition as fixed parameters and other variable parameters as input to the simulator [8].

These variable parameters were the probability of a node being uncooperative, before the HTLC was established and after, the average payment rate, the number of payments, and the tuner of the payment amount which is defined by the order of magnitude in satoshis. For each of these parameters the tuning has been made on various intervals.

The purpose of the simulation was to discover cases of non-operativeness. A case of non-operativeness indicates that the percentage of success falls under the 50%. The used strategy previewed to use for every variable a not stressing value, which does not negatively affect the simulation such as a percentage of failure equal to zero, and a stressing value that influences on the simulation, for example a percentage of failure different from zero.

The initial simulation was done with all variables with the not stressing value, then the stressing value was used one at a time on different simulations. If a case of not operating occurred for one of these simulations, no further simulations were done with other variables with stressing values. Otherwise, a stressed value on another variable was used.

The results of the various simulations showed that some combinations created cases of not working. Stressing the payment amount from the tuner by giving value 5 rather than the relaxed value 1 decreased the performance of the network bringing the probability of successful payment to 46.13% or below 50%. Moreover the probability of payment failure for no route found increased from 24.16% to 46.11%. This was due to the fact that the channels did not have the necessary capacity to route payments with large amounts.

The stressed value of the payment rate also affected the performance of the network. The relaxed value of 10 was increased to 100 reducing the probability of a successful payment to 43.88%. This was due to an unbalance of channels that were exhausted due to the high rate of payments being submitted.

Another non-operational case was found with a combination of tuner payment amount set to 4 and the probability of a node uncooperative set to 10%. The success rate dropped to 46.1% mostly due to too high payments and lack of routes to route them. This analysis led to the conclusion that one of the main problems on the Lightning Network is the routing of payments with high amounts.

**Synthetic network**: another analysis was done on a synthetic network generated using the network generator included in the CLoTH simulator. These simulations were performed with the purpose of studying the impact of each input parameter individually on network performance.

These input parameters in addition to those tested in the previous simulation were specifically: the number of nodes in the network, the average number of channels per node, channel capacity, and a topology tuner where a value equal to 0 indicated the centralized case where a hub node was connected with all the remaining nodes instead a value that tends to inf indicated a completely decentralized topology with nodes that were all the same size and no hubs were present.

In this simulation, the variables were considered individually and the others set to a default value so as not to affect the calculations.

The default synthetic network was totally decentralized (inf) composed from 100,000 nodes with an average of 5 channels each with average capacity of 100,000 satoshis.

The results obtained in this analysis stated that in a decentralized network three channels per node were not sufficient to have a robustly connected network. The probability of success of payments was 59.61% and the probability of failure for lack of route was 23.34% while for lack of funds was 16.77%.

With 5 channels per node, the success rate increased to 99% and the optimal condition for which there were no failed payments was found with 11 nodes per node.

The probability of node non-cooperation did not give a significant impact on network performance. Only the worst case with an improbably high probability of non-cooperation of 10% caused a probability of failure of 11.83%.

The results regarding network topology showed that payment times were lower with a centralized topology (333.16 ms) than with the fully decentralized case (1,391.92 ms). In addition there was an increase in average hops per payment from 2.90 to 10.34 but it did not affect the payment success rate.

## 4.3 Simulations on centrality with CLoTH

Simulations with CLoTH were performed on centrality analyses on networks where nodes were removed using the One Iteration algorithm. The central nodes removed were those that are central from the beginning of the removal process. This choice was made to focus the analysis on the performance of the Lightning Network in its initial state and in the state immediately following the removal of the central nodes.

In fact, the Iterated analysis after node removal and recalculation of the centrality measure studied a Lightning Network graph in a different state than the previous one in each iteration.

**Simulation Inputs**: in addition to the network specifications defined by the channel files, edges and nodes, other parameters regarding payments had to be given as input to be necessarily defined to the simulator for the generation of random payments between two nodes in the network. These payments were chosen with a low value, on average 100 satoshis.

For the analysis, 50,000 payments were generated to be simulated at a rate of 100 payments per second. This number was chosen based on analyses done in past studies so that it did not have a heavy impact on computation time, but gave relevant results.

For each centrality measure computed in the previous chapter using the One Iteration algorithm, simulations were conducted on the Lightning Network removing the central nodes. In addition to the removal of the total list of central nodes, three intermediate simulations were also conducted for each quarter of the list to show the performance trend of the network upon removal of a portion of central nodes.

The simulator gave network performance measures in various aspects such as payment success rate, failure rate due to lack of route, and lack of budget.

Other performance measures given in the output included average payment time, average route length, and number of attempts required for successful payment.

The results obtained from simulations on the various centrality measures and their intermediate removals will be discussed below for each performance measure.

The various analyses will compare the original network with the removal of the obtained central nodes of the related centrality measures and their intermediate steps. The original network has a payment success rate of 99.02%. The failure rate was mainly due to lack of paths of 0.61% and 0.37% due to lack of budget in the channels. The average route length of payments in simulations was 3.17 hops with an average time for payment of 635.11 ms.

### 4.3.1 Simulation with Degree Centrality

This section will discuss the results obtained from degree centrality on the binary and weighted network simulations.

The results regarding the success, failure and the average length of the payments obtained in the original network, after the removal of all the central nodes and in the intermediates are visible in the figure 4.1 regarding the binary network, while in the figure 4.2 regarding the weighted network. At the end in the table 4.1 are reported the final results with also the average time of payments obtained from the simulations.

**Degree Binary**: in the binary degree centrality after the removal of all central nodes presented a drop in performance of almost 8% having a success rate of 91.91% with a sharp decline after the three-quarter removed. The drop in performance is mainly due to lack of paths of 7.57% and 0.52% due to lack of funds.

In the performance of intermediate simulations, payments still have a good success rate. With the removal of half of the central nodes the success rate drops to 97.99% mainly due to lack of path 1.44% and a small part due to lack of funds 0.56%.

In the binary degree centrality the failure of payments was mainly due to the lack of paths between nodes after the removal of the central ones. There was also an increase in average payment time to 878.35 ms as can be seen in the table 4.1.

Also as illustrated in Figure 4.1 removing three-quarters of total nodes causes more failures due to lack of budget than removing all nodes. This may mean that some of these central nodes had low capacity channels or channels that were often used by a single direction and therefore saturated. After the removal of these nodes in fact there is an increase in failures due to the absence of path and decrease in failures due to the budget.

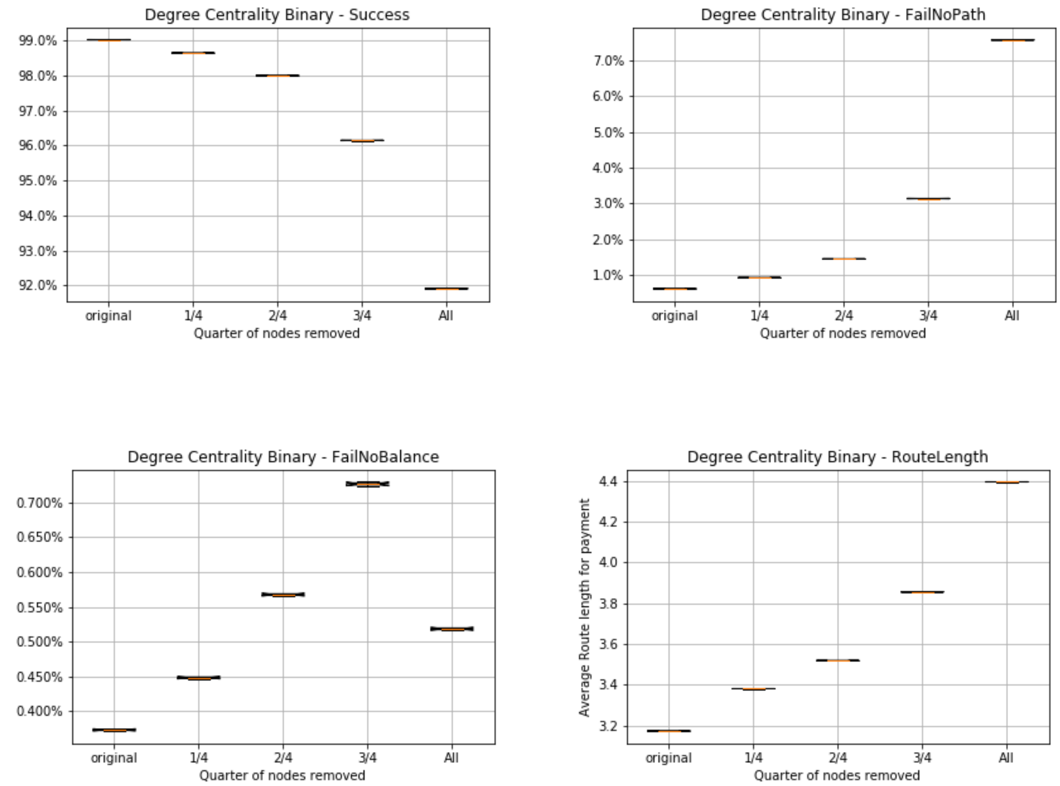



**Figure 4.1:** Success rate and failure in Degree Binary

**Degree Weighted**: in weighted degree centrality the removal of all central nodes resulted in a drop in payment success rate of almost 10% with a 90.85% success rate. Payment failure is primarily due to lack of path between nodes for 8.41% of payments while only 0.74% for no balances. The average length of payment paths increases to 4.62 hops.

Regarding the intermediate results, there was a fairly linear decrease in the payment success rate. With the removal of the first quarter of nodes it decreases to 98.25% increasing to 95.85% with the removal of half and 94.04%. Even in the intermediate cases, the main cause of failure was due to pathways.

Average payment times increased to 942.35 ms after removal of all central nodes, over 300 ms compared to the original network. Intermediate removals show a linear increase in times to node removal.

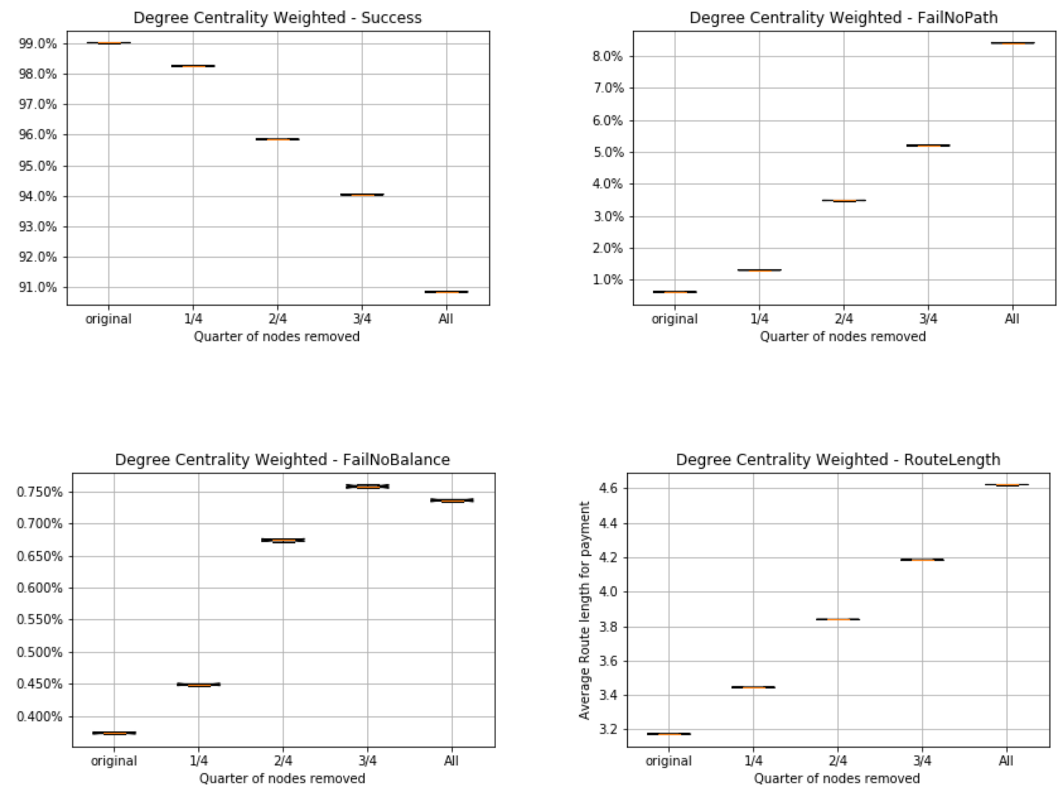



**Figure 4.2:** Success rate and failure in Degree Weighted

Table 4.1 shows the results obtained from the two degree centrality analyses. Removing the weighted degree centrality nodes was found to be more effective in dropping network performance but requires twice as many nodes to be removed. There was an increase in average payment time in both simulations that was slightly higher in the weighting.

| Comparison of simulations | | | |
|---|---|---|---|
| | Original | Degree Binary | Degree Weighted |
| Nodes removed | | 80 | 162 |
| Success | 99.02% | 91.91% | 90.85% |
| Fail no Path | 0.61% | 7.57% | 8.41% |
| Fail no Balance | 0.37% | 0.52% | 0.74% |
| | | | |
| Average Time | 635.11 ms | 878.35 ms | 942.35 ms |
| Average Route Length | 3.17 | 4.39 | 4.62 |

**Table 4.1:** Comparison of performance in original Lightning Network and Degree Centralities

### 4.3.2 Simulation with Eigenvector Centrality CLoTH

This section will discuss the results obtained from eigenvector centrality on the binary and weighted network simulations.

The results regarding the success, failure and the average length of the payments obtained in the original network, after the removal of all the central nodes and in the intermediates are visible in the figure 4.3 regarding the binary network, while in the figure 4.4 regarding the weighted network. At the end in the table 4.2 are reported the final results with also the average time of payments obtained from the simulations.

**Eigenvector Binary**: The success of payments in the eigenvector centrality dropped to 89.11% in the simulation with the removal of all central nodes. Again, the largest percentage was due to paths for 10.21% and a small portion for the balance 0.68% of failures. In addition, the average length of payment paths goes up 2 hops to 5.12. The intermediate cases do not yield results of interest. Removing the first node reduces the probability of success for both failure cases by 0.44%. Only after three-quarters of the central nodes does the success rate drop below 95% and the average path length rises to 4.20 hops. Even in eigenvector centrality, payment failure is mainly due to the lack of paths between nodes after the removal of the central ones. The average time of the payments rises above a second with 1,030 ms on average, as can be see in Table 4.2.

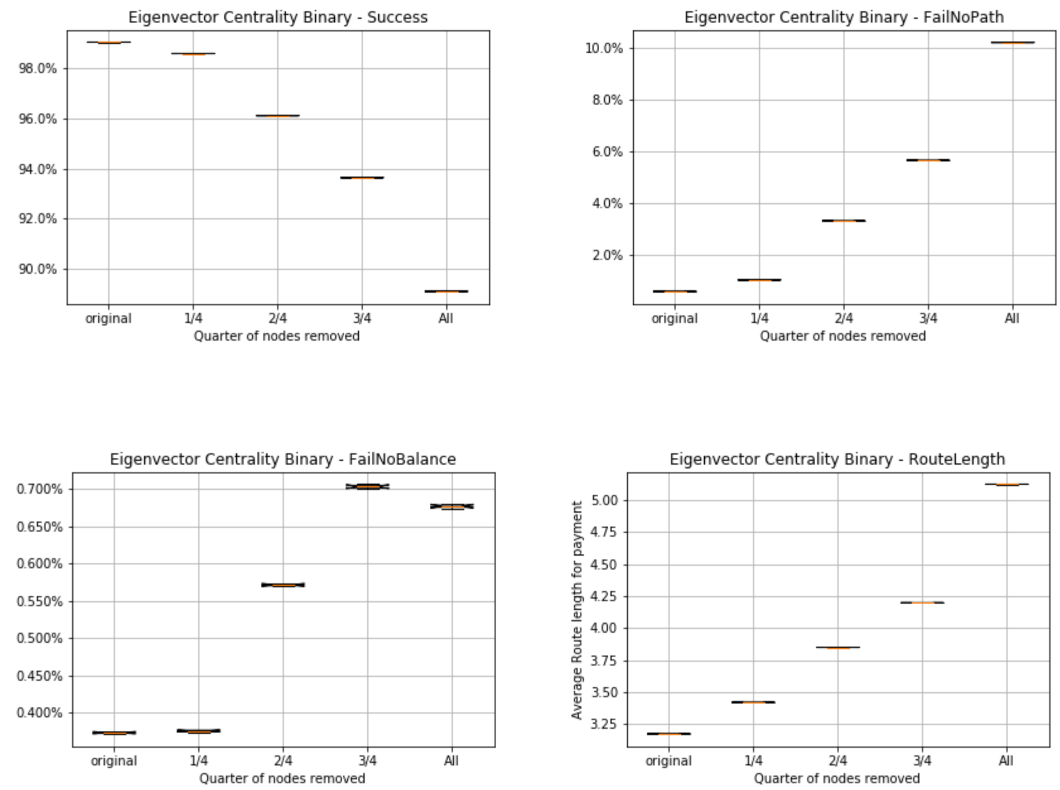



**Figure 4.3:** Success rate and failure in Eigenvector Binary

**Eigenvector Weighted**: In weighted eigenvector centrality the success rate dropped linearly based on intermediate removals. The payment success rate dropped below 94% due primarily to path failure, after the remotion of half nodes.

With the removal of all the central nodes the success rate of payments dropped to 88.64% with 10.51% caused by path failures and 0.85% for balance failure. The average time per payment increases to 912 ms.

This was the centrality measure for which the rate of failures rose more between all the tests, but it was also that one that demanded more central nodes in order to halve the largest connected component, with 342 central nodes.

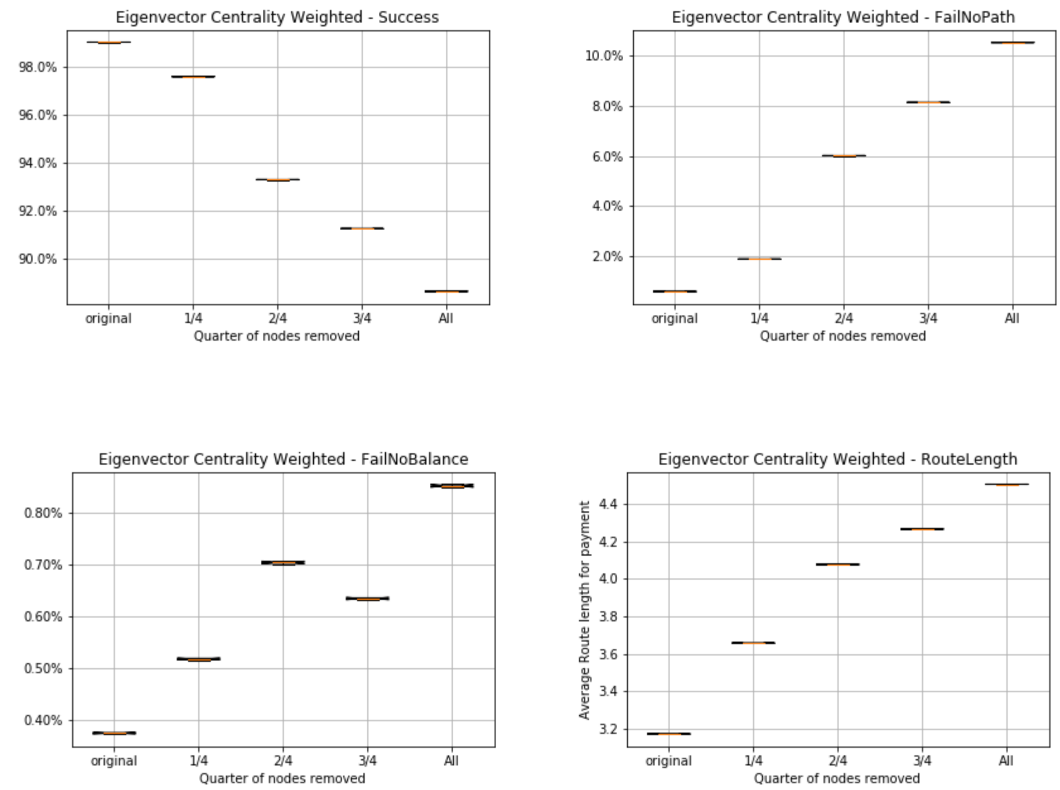



**Figure 4.4:** Success rate and failure in Eigenvector Weighted

Table 4.2 shows the results obtained from the two eigenvector centrality analyses. Removing nodes from the binary eigenvector centrality was more effective in terms of average time and average path length in dropping network performance, requiring about half as many nodes to be removed. The weighted eigenvector, on the other hand, is more effective in dropping the success rate but requires more nodes. With the same amount of nodes removed in the binary case, the weighted eigenvector would reduce the hit rate to about 88.64%.

| Comparison of simulations | | | |
|---|---|---|---|
| | Original | Eigenv. Binary | Eigenv. Weighted |
| Nodes removed | | 187 | 342 |
| Success | 99.02% | 89.11% | 88.64% |
| Fail no Path | 0.61% | 10.21% | 10.51% |
| Fail no Balance | 0.37% | 0.68% | 0.85% |
| Average Time | 635.11 ms | 1,030.83 ms | 912.43 ms |
| Average Route Length | 3.17 | 5.12 | 4.51 |

**Table 4.2:** Comparison of performance in original Lightning Network and Eigenvector Centralities

### 4.3.3 Simulation with Betweenness Centrality

This section will discuss the results obtained in the Betweenness centrality on the binary and weighted network simulations.

The results regarding the success, failure and the average length of the payments obtained in the original network, after the removal of all the central nodes and in the intermediates are visible in the figure 4.5 regarding the binary network, while in the figure 4.6 regarding the weighted network. At the end in the table 4.3 are reported the final results with also the average time of payments obtained from the simulations.

**Betweenness Binary**: in the binary betweenness centrality after the removal of all central nodes presented a drop in performance of almost 7% having a success rate of 92.72%. The drop in performance is mainly due to lack of paths of 6.84% and 0.52% due to lack of funds.

In the performance of intermediate simulations, payments still have a good success rate. With the removal of half of the central nodes the success rate drops to 98.01% mainly due to lack of path 1.34% and a small part due to lack of funds 0.65%. Also as illustrated in Figure 4.5 removing half or three-quarters of total nodes causes more failures due to lack of budget than removing all nodes.

In the binary betweenness centrality the failure of payments was mainly due to the lack of paths between nodes after the removal of the central ones. There was also an increase in average payment time to 814.25 ms as can be seen in the table 4.3.

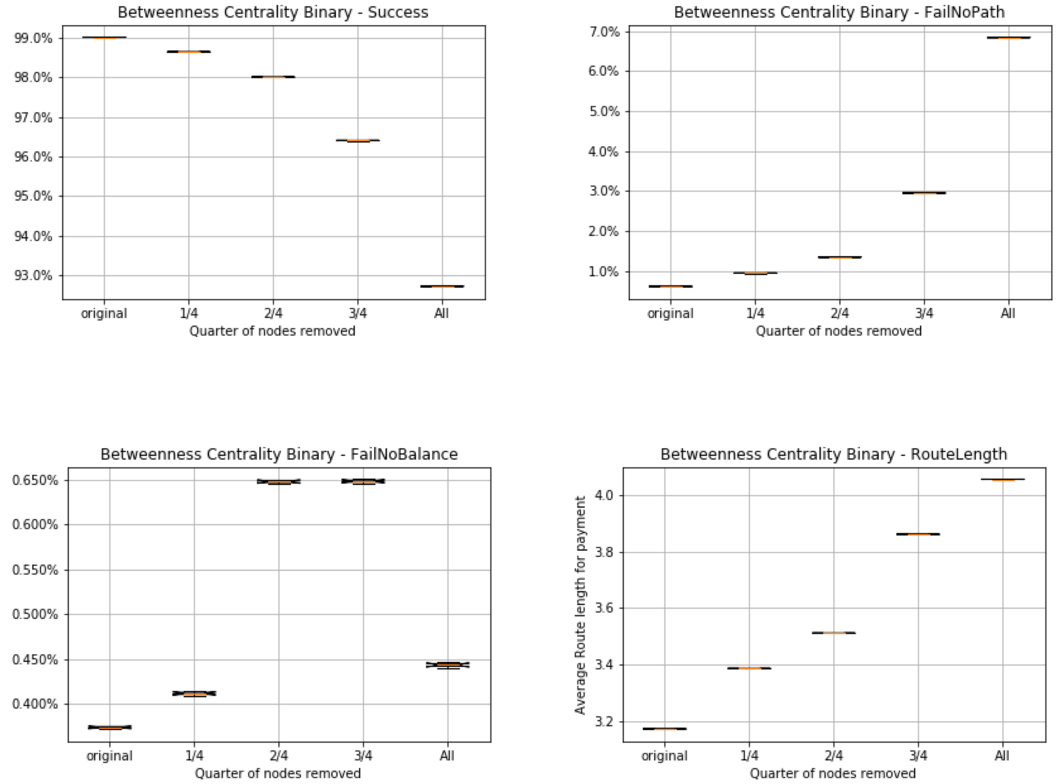



**Figure 4.5:** Success rate and failure in Betweenness Binary

**Betweenness Weighted**: in weighted betweenness centrality the removal of all central nodes resulted in a drop in payment success rate of almost 8% with a 92.24% success rate. Payment failure is primarily due to lack of path between nodes for 7.22% of payments while only 0.54% for no balances. The average length of payment paths increases to 4.62 hops.

Regarding the intermediate results, there was a fairly linear decrease in the payment success rate. With the removal of the first quarter of nodes it decreases to 98.38%. With the removal of half central nodes, there is an decreasing to 97.67% in payment success and a 95.69% of payment success after the removal of three-quarters of central nodes. Even in the intermediate cases, the main cause of failure was due to pathways.

Average payment times increased to 807.11 ms after removal of all central nodes, over 300 ms compared to the original network. Intermediate removals show a linear increase in times to node removal.

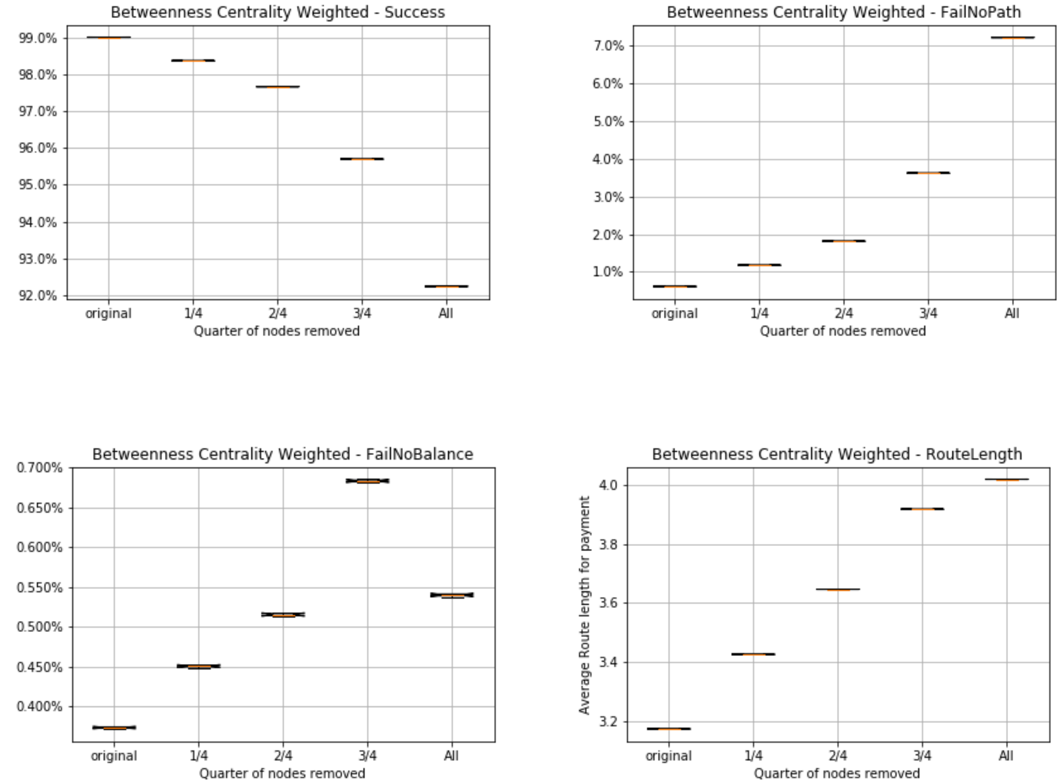



**Figure 4.6:** Success rate and failure in Betweenness Weighted

Table 4.3 shows the results obtained from the two betweenness centrality analyses. These two analysis give similar results in all performances, but the weighted needed more nodes.

### 4.3.4 Simulation with Closeness Centrality

This section will discuss the results obtained from closeness centrality on the binary and weighted network simulations.

The results regarding the success, failure and the average length of the payments obtained in the original network, after the removal of all the central nodes and in the intermediates are visible in the figure 4.7 regarding the binary network, while in the figure 4.8 regarding the weighted network. At the end in the table 4.4 are reported the final results with also the average time of payments obtained from the simulations.

| Comparison of simulations | | | |
|---|---|---|---|
| | Original | Between. Binary | Between. Weighted |
| Nodes removed | | 71 | 83 |
| Success | 99.02% | 92.72% | 92.24% |
| Fail no Path | 0.61% | 6.84% | 7.22% |
| Fail no Balance | 0.37% | 0.44% | 0.54% |
| Average Time | 635.11 ms | 814.25 ms | 807.11 ms |
| Average Route Length | 3.17 | 4.06 | 4.02 |

**Table 4.3:** Comparison of performance in original Lightning Network and Betweenness Centralities

**Closeness Binary**: Simulation on the network after all central nodes were removed according to closeness centrality gave a discrete performance drop. The success rate of the payments fell to 89,96% with 9,43% of failures due to the path while 0,61% to the balance. With half of the central nodes removed, the success rate dropped below 95% and the average payment route length was 3.95 hops with an average time per payment of 797 ms. Removing all nodes further increased the payment time to 1 second and the average route length to 4.98 hops.

The removal of three-quarters of nodes the rate of failed payments decreased mainly for no-balance compared to the simulation with half nodes removed, from 0.72% to 0.64%. This trend resumes also with the removal of all the central nodes going down to 0,61%.

**Closeness Weighted**: for the simulations carried out on the network after the removal of the central nodes according to the weighted closeness centrality there was a decrease in the success rate of payments under 90% with 88.74%. The 10.45% of the failures was due to the lack of path between the nodes and the remaining 0.81% instead for no balance.

Table 4.4 shows the results obtained from the two closeness centrality simulations. The removal performed in the weighted closeness centrality is found to be more effective in dropping network performance. The simulation on the binary network, on the other hand, made the average time spent on transactions and the average path length increase.

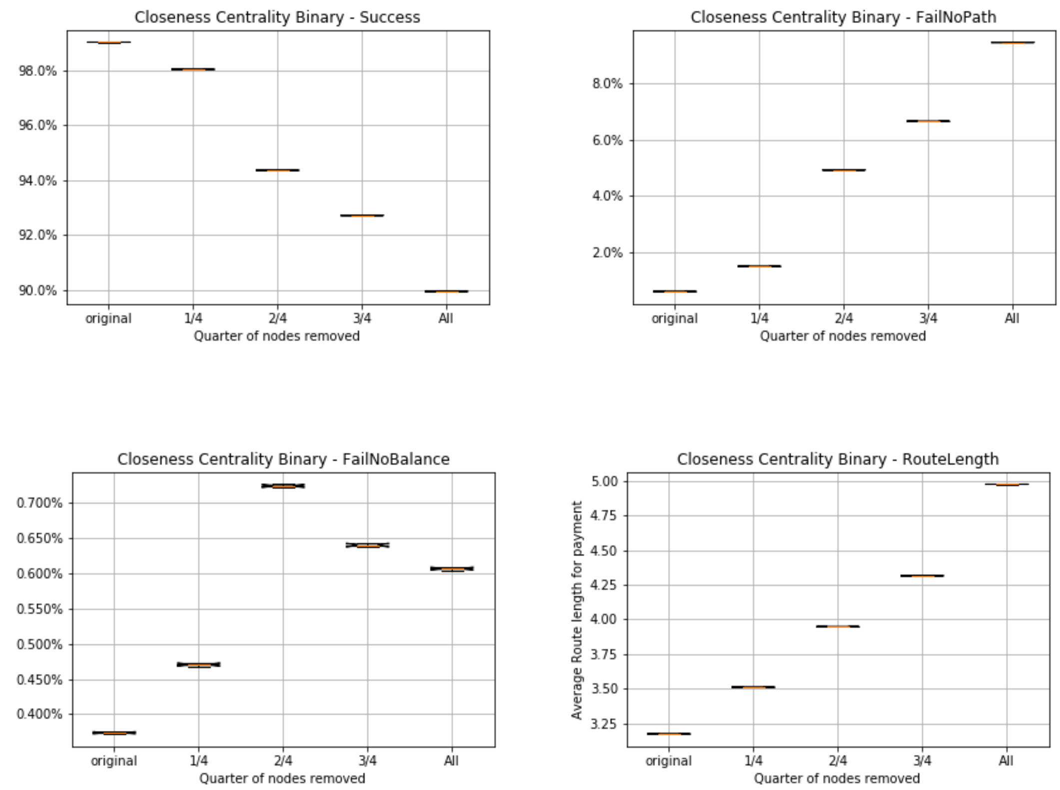



**Figure 4.7:** Success rate and failure in Closeness Binary

| Comparison of simulations | | | |
|---|---|---|---|
| | Original | Closeness Binary | Closeness Weighted |
| Nodes removed | | 296 | 294 |
| Success | 99.02% | 89.96% | 88.74% |
| Fail no Path | 0.61% | 9.43% | 10.45% |
| Fail no Balance | 0.37% | 0.61% | 0.81% |
| | | | |
| Average Time | 635.11 ms | 1004.13 ms | 894.72 ms |
| Average Route Length | 3.17 | 4.98 | 4.43 |

**Table 4.4:** Comparison of performance in original Lightning Network and Closeness Centralities

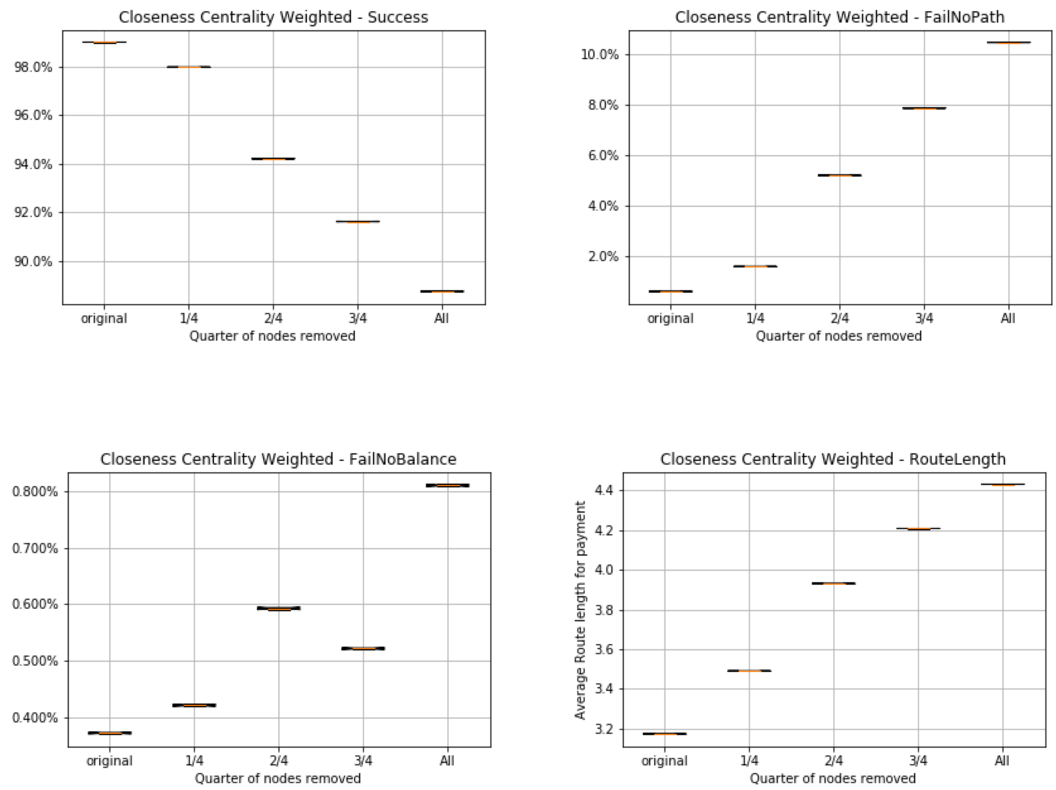



**Figure 4.8:** Success rate and failure in Closeness Weighted

### 4.3.5 Second Order and Current Flow Betweenness

This section will discuss the results obtained from second order centrality and the current-flow betweenness centrality simulations.

The results regarding the success, failure and the average length of the payments obtained in the original network, after the removal of all the nodes and in the intermediates are visible in the figure 4.9 regarding the network with random removal, while in the figure 4.10 regarding the weighted network.

**Second Order**: for the simulation after the removal of all 124 of central nodes according to the second order centrality network the success rate of the payments fell to 90,78% with 8,63% of failures due to the path while 0,58% to the balance.

With half of the central nodes removed, the success rate dropped to 97,68% and the average payment route length was 3.68 hops with an average time per payment of 738.26 ms. Removing all nodes further increased the payment time to 968.55 ms and the average route length to 4.83 hops.

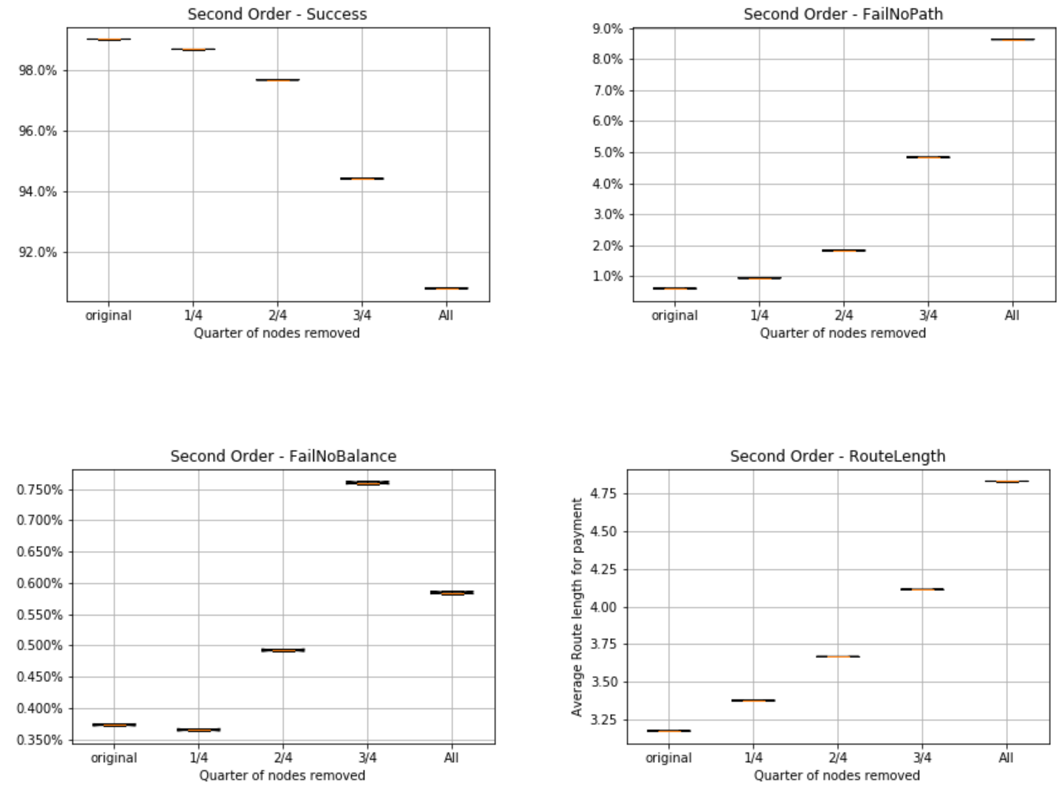



**Figure 4.9:** Success rate and failure in SecondOrder

**Current Flow Betweenness**: the simulation with the removal of central nodes according to the current flow centrality resulted in a drop in payment success rate of with a 91.00% success rate. Payment failure is primarily due to lack of path between nodes for 8.50% of payments while only 0.50% for no balances. The average length of payment paths increases to 3.91 hops.

Regarding the intermediate results simulations, the success rate dropped to 98.07% with the remotion of half centrality nodes, the majority of failures was due to no path with 1.39% and 0.54% for no balance, the network still had good performance. Removing three quarters of the total central nodes in simulations, the success rate was 96.40% and the failures due mainly to no path for 2.80% and the remaining 0.80% for no balance.

Average payment times increased to 783.82 ms after removal of all central nodes, 148 ms compared to the original network. Intermediate removals show a linear increase in times to node removal.

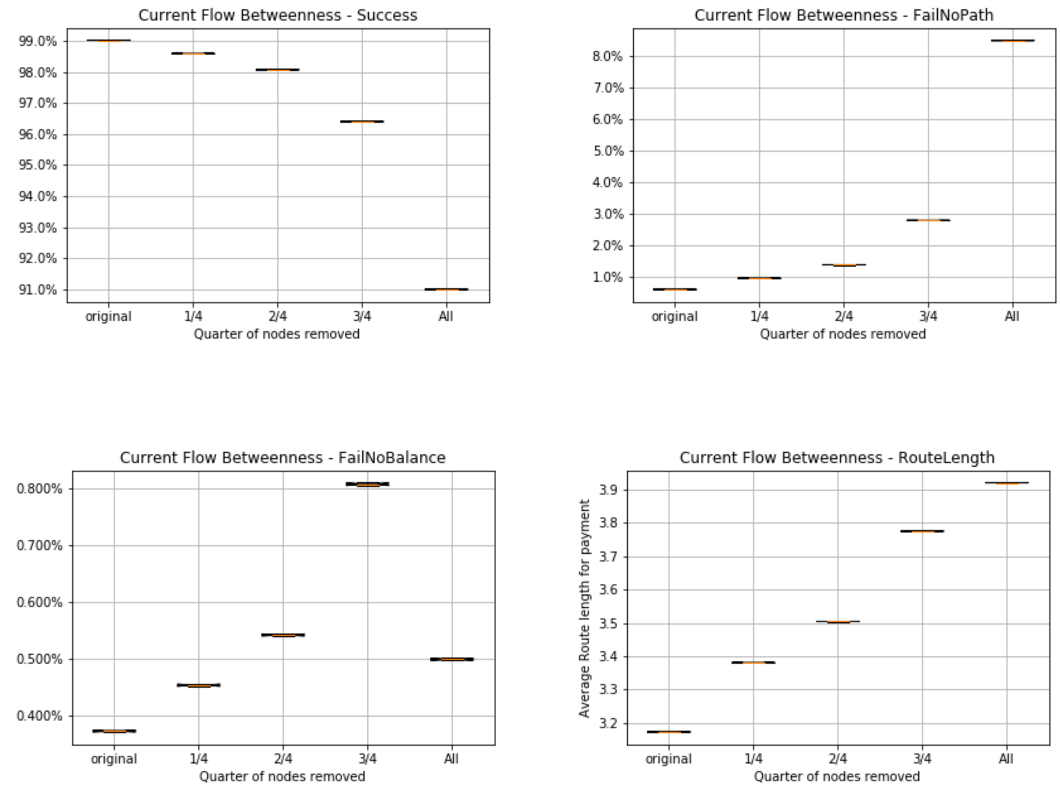



**Figure 4.10:** Success rate and failure in Current Flow

### 4.3.6 Simulation with Random removal and Elite Nodes

This section will discuss the results obtained from the random removal and the removal of the Elite Nodes simulations.

The results regarding the success, failure and the average length of the payments obtained in the original network, after the removal of all the nodes and in the intermediates are visible in the figure 4.11 regarding the network with random removal, while in the figure 4.12 regarding the weighted network. At the end in the table 4.5 are reported the final results with also the average time of payments obtained from the simulations.

**Random removal**: after the removal of all the nodes in the list of removal network performance didn't drop by much. In fact, after removing all 2,107 nodes from the list, the success rate dropped by only 1% compared to the original network, or 98.02%. 1.51% of the failures were due to the path not found and the remaining 0.46% were due to funds. There was also no increase in the average route length as showed in table 4.5.

**Elite Nodes**: the simulation after removing all 41 Elite Nodes in the network gave a discrete decrease in network performance. The success rate dropped to 96.75% with the majority of failures due to no path with 2.56% and the remaining 0.69% due to balance. Removing three-quarters of the total Elite nodes the network still had good performance as the success rate was 98.09% and the failures due mainly to no path for 1.35% and the remaining 0.56% for no balance.

Average payment times increased to 756.76 ms after removal of all central nodes, over 120 ms compared to the original network. Intermediate removals show a linear increase in times to node removal.

Table 4.5 shows the results obtained from the random and the Elite Nodes removal. From the table it can be seen that removal of Elite Nodes is much more effective than random removal of nodes. With only 41 nodes or 0.68% of the nodes in the network reduced the success rate more than the random removal of 2,107 nodes, 96.75% and 98.02% respectively. Among the random nodes removed there were also 12 of these Elite Nodes but too few to affect the performance of the network with their removal.

The average path length was greater after Elite node removal than the average length found in the simulation run after random removal, 3.77 hops versus 3.18 hops.

The results obtained on the various measures of centrality showed that the network had a decrease in performance in the absence of central nodes. These nodes mainly cause with their removal a lack of paths between nodes as seen from the results obtained.

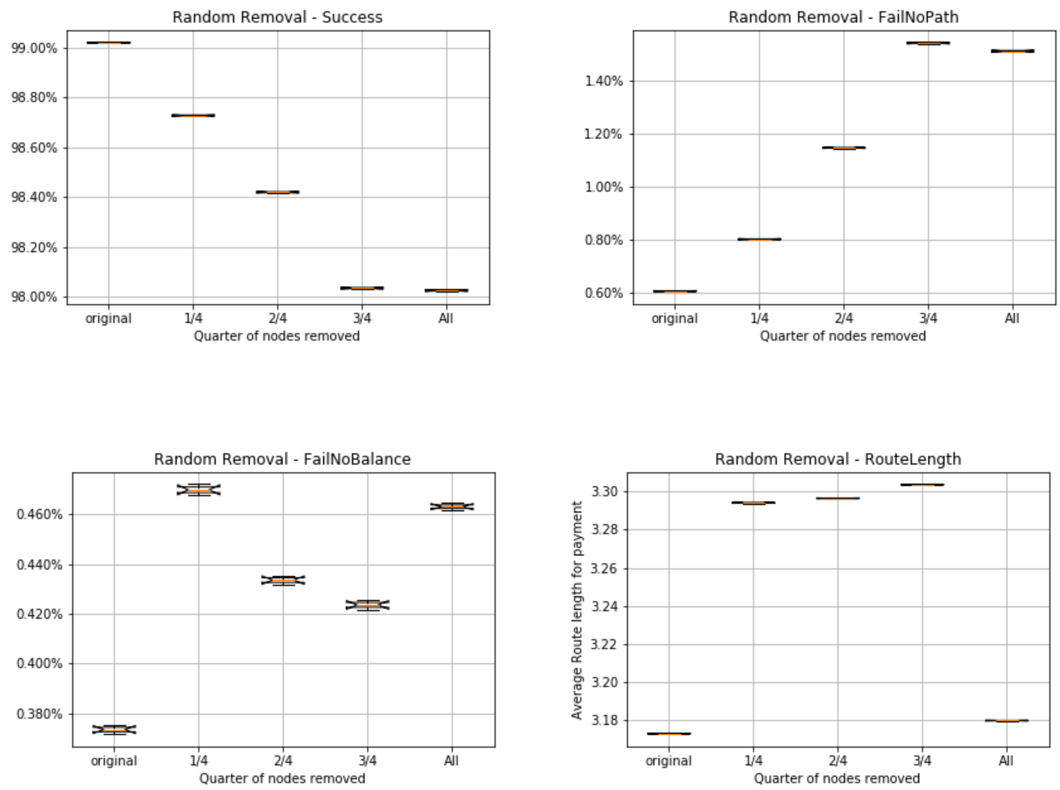



**Figure 4.11:** Success rate and failure in random removal

Despite this, the remaining network possessed enough links to ensure transactions between the remaining nodes with a good success rate. In fact, the lowest value of the success rate obtained in the weighted eigenvector centrality was 88.64%.

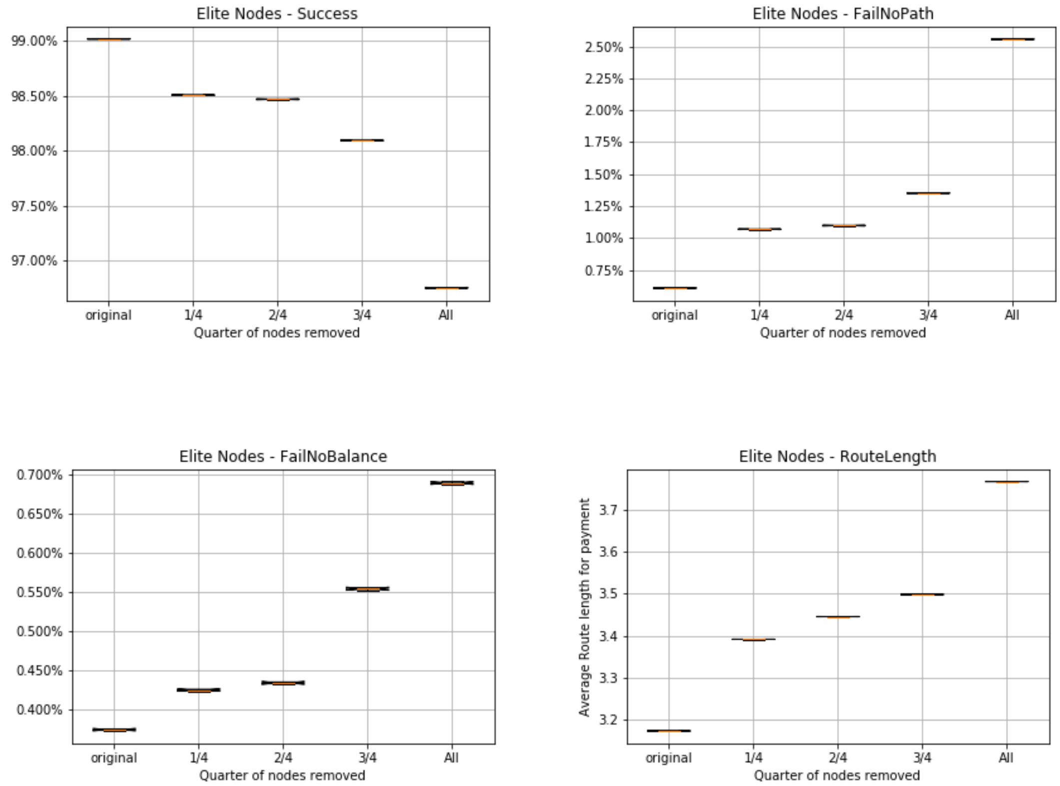



**Figure 4.12:** Success rate and failure in Elite Nodes

| Comparison of simulations | | | |
|---|---|---|---|
| | Original | Random | Elite Nodes |
| Nodes removed | | 2,107 | 41 |
| Success | 99.02% | 98.02% | 96.75% |
| Fail no Path | 0.61% | 1.51% | 2.56% |
| Fail no Balance | 0.37% | 0.46% | 0.69% |
| Average Time | 635.11 ms | 643.87 ms | 756.76 ms |
| Attempts | 1.015 | 1.045 | 1.029 |
| Average Route Length | 3.17 | 3.18 | 3.77 |

**Table 4.5:** Comparison of performance in original Lightning Network, after random removal and Elite Nodes

# Chapter 5

# Conclusions

Lightning Network is the most popular payment channel network on Bitcoin designed to solve the scalability problem. Thanks to its smart contracts called HTLC, it allows unbounded off-chain payments in almost instant time through trustless intermediaries, with small fees to route them.

Since channel creation takes funds from the blockchain nodes tend to create few channels by connecting to central nodes in the network rather than creating ad hoc ones.

Central nodes can compromise the decentralization of the network and if they are not cooperative they can increase the rate of failed payments or the average distances to cover for payment and consequently an increase of the fees.

This study focused on finding these various nodes based on different centrality measures and seeing the behavior and topology of the Lightning Network after the removal of the central nodes.

For the study, several tests were performed for the various centrality measures. The binary network and the weighted network were considered by changing the weight value for the edges (considering or not considering the weight parameter).

Network analyses were conducted by removing nodes based on their centrality by performing two removal strategies.

In the first, the removed nodes were sequentially chosen from the list after a single iteration of the centrality measure. In the second, the centrality measure was recalculated after each removal and then the node removed. The number of detached connected components, isolated nodes, and the size of the largest connected component were reported for each node removed. This analysis stopped when the largest connected component had half of the initial nodes in the network.

Topological analyses were followed by simulations using CLoTH. CLoTH is a Lightning Network simulator. It simulates payments on the Lightning Network and produces performance measures, such as probability of payment success and average payment time. For each centrality measure, simulations were conducted on the Lightning Network removing the central nodes. In addition of the remotion of the total list of central nodes three intermediate simulations were also carried out for each quarter of the list.

Topological results showed that few central nodes are enough for halve the largest connected component. Most of these nodes are isolated, confirming that nodes have few connections with central nodes. Payments needed on average more hops to reach the destination after the removal of the central nodes. There was also an increase in network diameter, the maximum distance between two nodes.

There was also a significant reduction in the number of channels, from 30,457 in the original network to a minimum of 5,095 during the analysis with an iteration of closeness centrality in the binary network. There was also a notable reduction in total capacity which drops to a minimum of 40 ₿ in the eigenvector centrality iterated on the weighted network.

The simulations on CLoTH confirmed the results obtained from the topological analysis performed previously such as the average increase in the number of hops per payment, adding details regarding the increase in the average time for payments and more payment failures.

On average, removing the various central nodes for each centrality measure reduced payment success by 9%, from 99% in the original network to an average of 91%. The most significative reduction was obtained by eigenvector centrality on the weighted network with 89% successful payments.

Average payment times had an increase of about 268 ms from 635 ms in the original network to 903 ms on average across measures. Payments have to take one extra hop to reach their destination.

For the test with random removal, the network was very robust experiencing no actual change on arrival times from 635 ms to 644 ms and only 1% of payments fail with a 98% of success rate. The test with the removal of Elite Nodes gave the following results: the success rate decreased to 97% and the average payment time increased to 757 ms with an increase of 122 ms compared to the original network.

The analysis and the simulations showed that in the network there were indeed central nodes that with their non-cooperation to the network can create inconveniences to it. In fact, the removal of 41 (0,7% of the total nodes) central nodes for all the measures considered was much more effective than a random removal of 2.107 (35% of the total nodes) nodes for the simulations with CLoTH, giving higher results in all fields.

Despite these results, the remaining network appeared to be well connected in as it did not have too low a success rate. In fact the vast majority of nodes that remained isolated from the network belonged to small users connected only to one of these central nodes.

Future work will investigate different centrality measures such as group measures of nodes. In this study the nodes were treated individually, but it is also interesting to understand if in the network there are nodes even distant that taken together can divide the largest connected component in several chunks of considerable size as opposed to small connected components formed by a few nodes.

In addition, other future work of interest will be simulations using CLoTH studying the intermediate stages of the network during the removal Iterated and study whether the removal of these nodes causes more or less damage to the network than the study carried out on this thesis work.

# Bibliography

[1] Satoshi Nakamoto. «Bitcoin: A peer-to-peer electronic cash system Bitcoin: A Peer-to-Peer Electronic Cash System». In: *Bitcoin. org.* (2009). URL: `https://bitcoin.org/en/bitcoin-paper` (cit. on pp. 1, 5).

[2] Joseph Poon and Thaddeus Dryja. *The bitcoin lightning network: Scalable off-chain instant payments.* 2016 (cit. on pp. 1, 9).

[3] Stefano Martinazzi. «The evolution of Lightning Network's Topology during its first year and the influence over its core values». In: *arXiv preprint arXiv:1902.07307* (2019) (cit. on pp. 1, 21).

[4] Stefano Martinazzi and Andrea Flori. «The evolving topology of the Lightning Network: Centralization, efficiency, robustness, synchronization, and anonymity». In: *PloS one* 15.1 (2020), e0225966 (cit. on pp. 1, 21).

[5] Elias Rohrer, Julian Malliaris, and Florian Tschorsch. «Discharged payment channels: quantifying the lightning network's resilience to topology-based attacks». In: *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW).* IEEE. 2019, pp. 347–356 (cit. on pp. 1, 21).

[6] Jian-Hong Lin, Kevin Primicerio, Tiziano Squartini, Christian Decker, and Claudio J Tessone. «Lightning Network: a second path towards centralisation of the Bitcoin economy». In: *New Journal of Physics* 22.8 (2020), p. 083022 (cit. on pp. 1, 21).

[7] István András Seres, László Gulyás, Dániel A Nagy, and Péter Burcsi. «Topological analysis of bitcoin's lightning network». In: *Mathematical Research for Blockchain Economy.* Springer, 2020, pp. 1–12 (cit. on pp. 1, 22).

[8] Marco Conoscenti. «Capabilities and Limitations of Payment Channel Networks for Blockchain Scalability». PhD thesis. Tesi di Dottorato. Politecnico di Torino, 2019 (cit. on pp. 2, 55–57).

[9] Andreas M Antonopoulos. *Mastering Bitcoin: unlocking digital cryptocurrencies.* " O'Reilly Media, Inc.", 2014 (cit. on p. 5).

[10] Chris Clark. «Bitcoin Internals: A Technical Guide to Bitcoin». In: *Amazon Digital Services* (2013) (cit. on p. 5).

[11] URL: https://en.bitcoin.it/wiki/ (cit. on p. 5).

[12] Kenny L. *The Blockchain Scalability Problem & the Race for Visa-Like Transaction Speed.* July 2019. URL: https://towardsdatascience.com/the-blockchain-scalability-problem-the-race-for-visa-like-transaction-speed-5cce48f9d44 (cit. on pp. 8, 9).

[13] *Software for Complex Networks.* Aug. 2020. URL: https://networkx.org/documentation/stable/ (cit. on p. 12).

[14] Ernesto Estrada, Desmond J. Higham, and Naomichi Hatano. «Communicability betweenness in complex networks». In: *Physica A: Statistical Mechanics and its Applications* 388.5 (2009), pp. 764–774. DOI: 10.1016/j.physa.2008.11.011 (cit. on pp. 12, 16).

[15] Lembris Laanyuni Njotto. *Centrality Measures Based on Matrix Functions.* Sept. 2018. URL: https://www.scirp.org/journal/paperinformation.aspx?paperid=87508 (cit. on p. 13).

[16] Ernesto Estrada. *The structure of complex networks: theory and applications.* Oxford University Press, 2012 (cit. on p. 13).

[17] Werner Dubitzky, Olaf Wolkenhauer, Hiroki Yokota, and Kwang-Hyun Cho. *Encyclopedia of systems biology.* Springer Publishing Company, Incorporated, 2013 (cit. on p. 13).

[18] *Degree Centrality.* URL: https://www.sciencedirect.com/topics/computer-science/degree-centrality (cit. on p. 13).

[19] *Eigenvector Centrality (Centrality Measure).* Feb. 2018. URL: https://www.geeksforgeeks.org/eigenvector-centrality-centrality-measure/ (cit. on p. 13).

[20] Natarajan Meghanathan. «Correlations between centrality measures for mobile ad hoc networks». In: *International Journal of Wireless Networks and Broadband Technologies (IJWNBT)* 4.2 (2015), pp. 15–27 (cit. on p. 13).

[21] Phillip Bonacich. «Power and centrality: A family of measures». In: *American journal of sociology* 92.5 (1987), pp. 1170–1182 (cit. on p. 14).

[22] URL: https://www.sci.unich.it/~francesc/teaching/network/ (cit. on p. 14).

[23] Ulrik Brandes and Daniel Fleischer. «Centrality Measures Based on Current Flow». In: *STACS 2005 Lecture Notes in Computer Science* (2005), pp. 533–544. DOI: 10.1007/978-3-540-31856-9_44 (cit. on p. 15).

[24] URL: https://neo4j.com/docs/graph-algorithms/current/labs-algorithms/harmonic-centrality/ (cit. on p. 15).

[25] *Betweenness Centrality (Centrality Measure).* Feb. 2018. URL: https://www.geeksforgeeks.org/betweenness-centrality-centrality-measure/ (cit. on p. 16).

[26] M. G. Everett and S. P. Borgatti. «The centrality of groups and classes». In: *The Journal of Mathematical Sociology* 23.3 (1999), pp. 181–201. DOI: 10.1080/0022250x.1999.9990219 (cit. on p. 16).

[27] Ulrik Brandes. «On variants of shortest-path betweenness centrality and their generic computation». In: *Social Networks* 30.2 (2008), pp. 136–145 (cit. on p. 16).

[28] Ernesto Estrada and Juan A Rodriguez-Velazquez. «Subgraph centrality in complex networks». In: *Physical Review E* 71.5 (2005), p. 056103 (cit. on p. 17).

[29] Lars Backstrom and Jon Kleinberg. «Romantic partnerships and the dispersion of social ties: a network analysis of relationship status on facebook». In: *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing.* 2014, pp. 831–841 (cit. on p. 17).

[30] Enys Mones, Lilla Vicsek, and Tamás Vicsek. «Hierarchy measure for complex networks». In: *PloS one* 7.3 (2012), e33799 (cit. on p. 17).

[31] Mahendra Piraveenan, Mikhail Prokopenko, and Liaquat Hossain. «Percolation centrality: Quantifying graph-theoretic impact of nodes during percolation in networks». In: *PloS one* 8.1 (2013), e53095 (cit. on p. 17).

[32] Anne-Marie Kermarrec, Erwan Le Merrer, Bruno Sericola, and Gilles Trédan. «Second order centrality: Distributed assessment of nodes criticity in complex networks». In: *Computer Communications* 34.5 (2011), pp. 619–628 (cit. on p. 18).

[33] Jian-Xiong Zhang, Duan-Bing Chen, Qiang Dong, and Zhi-Dan Zhao. «Identifying a set of influential spreaders in complex networks». In: *Scientific reports* 6 (2016), p. 27823 (cit. on p. 18).

[34] Tore Opsahl, Filip Agneessens, and John Skvoretz. «Node centrality in weighted networks: Generalizing degree and shortest paths». In: *Social networks* 32.3 (2010), pp. 245–251 (cit. on pp. 19, 20).