A Measurement Study of Bitcoin Lightning Network

by

Yuwei Guo

B.Eng, Beihang University, 2017

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE COLLEGE OF GRADUATE STUDIES
(Electrical Engineering)

The University of British Columbia
(Okanagan)

December 2019

The following individuals certify that they have read, and recommend to the College of Graduate Studies for acceptance, the thesis entitled:

A Measurement Study of Bitcoin Lightning Network

submitted by Yuwei Guo in partial fulfillment of the requirements of the degree of Master of Applied Science.

Chen Feng, School of Engineering
Supervisor
Yang Cao, School of Engineering
Supervisory Committee Member
Narayan Apurva, School of Engineering
Supervisory Committee Member
Eric Li, Faculty of Management
University Examiner

Abstract

As a promising method to enable fast and scalable Bitcoin transactions, Bitcoin Lightning Network (LN) has experienced rapid development since the end of 2017. LN utilizes the so-called "payment channels" to provide fast off-chain transactions, thereby offloading on-chain burden and enabling instant payments. With many new protocols proposed to improve the performance of LN, little is known about the current state of the network such as its topology, channel characteristics and application performance. This thesis conducts a measurement study on the performance of LN and provides some guide on improvement.

Lay Summary

In this thesis, a systematic measurement on LN was conducted based on the data collected over a period of fifteen months. This measurement studied the payment success rate and how the network performs under attack. Payment channels were also analyzed regarding their functions. This work provides an in-depth understanding of network mechanisms and helps to explore future implications of LN.

Preface

The work outlined in this thesis was conducted in the School of Engineering at the University of British Columbia, Okanagan Campus, under the supervision of Dr. Chen Feng. The main content in this thesis is based on our conference paper accepted by the 2nd IEEE International Conference on Blockchain (Blockchain-2019) with an acceptance rate of 15.9%.

Table of Contents

List of Tables

List of Figures

List of Symbols

CC connected component

DAG directed acyclic graph

LN Lightning Network

LCC largest connected component

LQC largest qualified component

POS Proof of Stake

POW Proof of Work

PR PageRank algorithm

QC qualified component

Acknowledgments

It was just like yesterday when I got my bachelor's degree from Beihang University and came to Canada. But now, my graduate study is coming to an end. Looking back at the past two year's life, there were times with depression, confusion and anxiety, but what's truly important is about self reflection, setting the goals and learning how to deal with obstacles independently. During this process, I received a lot of help and there are many people I must give thanks to.

First, I want to express my gratitude to my supervisor, Dr. Chen Feng. I'm really impressed by his insightful thinking on many research problems. Without his help, I could never finish my research and graduate studies.

Second, I want to thank my parents. They are always by my side when I make decisions. Their selfless love and support helps me go further.

In the end, I want to thank my friends: Chunpu Wang, Renming Qi, Yonghui Lv, Jianyu Niu, Fangyu Gai, Junyuan Leng, Jinfeng Tong, Pengxia Wu and Xueting Yang. Thanks for all your company and I'll never forget the happy times we spent together.

Chapter 1

Introduction

1.1 Motivation

Bitcoin is the first decentralized cryptocurrency [17] created in 2009 by pseudonymous developer Satoshi Nakamoto. Since the release of Bitcoin, the cryptocurrency area has entered an era of prosperity. According to statistics[1], there are 30 thousand cryptocurrencies and the total market capitalization is 220 billion as of Oct 15, 2019. We list some of the most famous cryptocurrencies at that time in Table 1.1.

Table 1.1: List of Some Famous Cryptocurrencies

Name	Released	Symbol	Market Cap (billion)
Bitcoin	2009	BTC	$145.00
Ethereum	2015	ETH	$19.12
XRP	2012	XRP	$12.85

The decentralized control of each cryptocurrency works through distributed ledger technology, typically a blockchain, that serves as a public financial transaction database. The simplified process is shown in Figure 1.1. Alice sends a payment to Bob and submits the transaction to the blockchain, once it is accepted

[1] CoinMarketCap

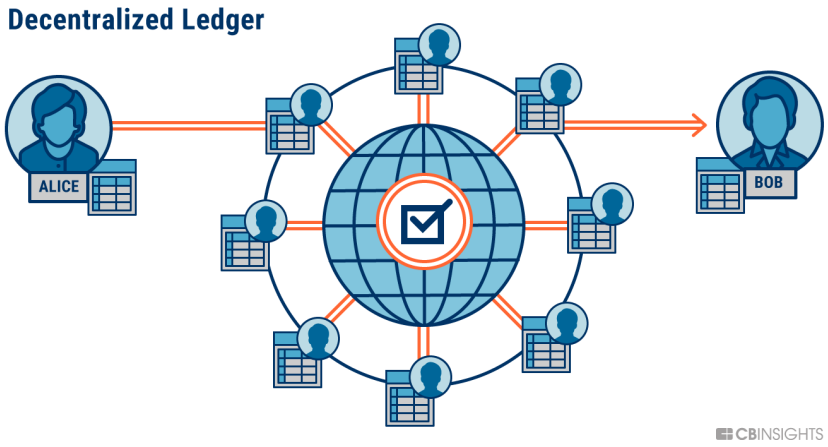


Figure 1.1: Illustration of blockchain

Source: CB Insights

by the network, all nodes will have a copy and this record will become immutable.

Though with such great development, the current blockchain technology still faces a pressing challenge - achieving high throughput. Why is this a problem? Let's first take a look at the basic principles of blockchain (more detailed explanations can be found in Section 2.1). Roughly speaking, the Bitcoin blockchain is comprised of a chain of blocks which store transactions. To ensure security, the growth rate of the chain is limited using Proof of Work (POW) consensus algorithm. Typically it takes 10 minutes on average to generate a new 1MB block. This nature of blockchain leads to scalability issue. From Table 1.2 we can see that Bitcoin can only write 7 transactions per second (TPS). In the meantime, Visa can handle 2,000 TPS on average.

Scaling has long been the focus of developers and many methods are proposed to improve the throughput. Some focused on the blockchain itself (i.e., "Layer 1"), while others intended to build a "Layer 2" system on top of the original blockchain. There exist several exciting "Layer 1" technologies like Algorand and Prism [1, 9]. Algorand adopts Proof of Stake (POS) consensus algorithm and achieves throughput of over 1,000 TPS. Prism uses multiple parallel chains instead of just one and can push the throughput to physical limits.

Table 1.2: Throughput of some payment technologies

Name	Visa	Bitcoin	Ethereum	Layer 2 of blockchain
Throughput (TPS)	2,000	7	15	optimal (according to bandwidth)

In spite of the performance, most "Layer 1" technologies are still at the R&D stage and the community haven't built enough experience. Shifting attention to "Layer 2", we note that one of the most promising solutions is an off-chain system called the Bitcoin Lightning Network (LN) [25] which can also achieve optimal throughput. Furthermore, since implemented in late year 2017, there are now about 3,000 active nodes and 840 BTC (6.7 million USD) in the network.

The cornerstone of LN is the payment channel. A payment channel allows two users to send multiple payments to each other and only touches the blockchain twice: at channel opening and closing time. Local payments inside the channel are settled instantly with no fees, and guaranteed that the rightful states can be broadcasted to the global blokchain at any time. Separate channels form a network by using Hashed TimeLock Contracts (HTLCs), which ensure that payments can be routed through intermediaries. In Section 2.2, we describe LN in more details.

LN has attracted a great attention from both academia and industry. Researchers have proposed protocols to improve routing efficiency [29], network liquidity [16] and rebalance channels [11]. The Bitcoin community recently launched an experiment called "lightning torch" to show the value and expand the influence of LN. There are also some individual projects. For example, a private project named LNBIG[2] has set up tens of nodes in LN and injected more than 300 BTC (the unit of Bitcoin). Moreover, the LNBIG adjusted its channel settings to help rebalance.

In spite of its great success in offloading on-chain burdens, the current LN itself still remains to be further investigated. First, little work has been done to study the topology of LN, therefore many significant questions remain unanswered. For example, how is Bitcoin distributed in this network? How efficiently can we make

[2]The website of the project is https://lnbig.com

a transaction through routing between any two users without a direct channel? Are there any important nodes in the network and to what extent does the network rely on them? The answers to the above questions can not only help us better understand the characteristics of LN, but also provide insights for future protocol design. Second, payment channels in LN are special: unlike connections in the Internet, establishing channels in LN will be charged. However, it's impractical to maintain a channel as long as possible. Because it decreases liquidity by locking deposits inside the channel. Exploring the opening and closing of channels helps provide better user experience. Third, most of the existing protocols lack simulations on a real-world offchain network topology. The need of an easy-access platform for simulation is urgent.

In this thesis, we answer the above questions by conducting a systematic measurement study on LN. Our contributions are as follows:

- We study the topology of the LN and provide an in-depth understanding. In particular, we apply graph theory to evaluate the network performance and then define metrics for demonstrating features of the payment channel.
- Our analysis reveals several issues of LN. Specifically, we show that the current LN does not perform well in routing and prove that it is vulnerable to attacks.
- The issues we observe and the metrics we introduce shed light on user guidance and future protocol design.
- We provide an easy-to-use LN topology for protocol simulations[3]. Our topology can not only better reflect the real situation, but also help to compare the performance of different algorithms.

1.2 Related Work

Since J. Poon and T. Dryja [25] proposed LN based on duplex micro-payment channels, there have been many works focusing on designing new protocols and improving network performance.

[3]For the purpose of reproducibility, data and codes of our work are available at https://github.com/measureln/measurement_study_on_ln.

In the LN, depleted payment channels only have balance on one side and need to be closed. Khalil et al. [11] introduces REVIVE protocol to maintain network balance equilibrium without closing and reopening depleted channels. SpeedyMurmus protocol [29] allows efficient routing for completely decentralized path-based transactions. In the LN payment routing process, intermediaries might collude to recover identities of the payment owners. Green et al. [10] construct anonymous payment channels to prevent privacy leaks of user identity. Though LN reduces many on-chain transactions, the channel deposits are locked as collateral and lacks liquidity. Sprites protocol [16] aims to reduce total collateral costs.

Though many new protocols are brought up, there is almost no paper focusing on the performance measurement of the current LN. Flash [31] utilizes payment characteristics to improve routing performance. However, it focuses on each node's local view and the measurement of the overall network is not covered. Another work [30] displays various attacking strategies and quantifies the network resilience. However, both work only use discrete snapshot graphs. By contrast, our work is among the first to reveal the evolution of network performance and the unique characteristics of payment channels.

There exists some analysis about the Ethereum network and Bitcoin network. The work of Chen, Ting et al. [5] characterizes Ethereum through graph analysis and provides many insights. Lischke et al. [14] mainly investigates the business distribution and business model of the Bitcoin Network based on IP addresses of nodes and business tags of transactions. AddressProbe technique [15] reconstructs Bitcoin network topologies through broadcasted messages and identify some influential nodes. Ron et al. [28] examine the structure of Bitcoin network and conclude that most of the bitcoins are in dormant status. They also find that large transactions are extremely rare in the network and most of the transactions are small-value transactions. Timing analysis [18] discusses the delay of the network and nodes latency.

Some work focuses on the anonymity of the Bitcoin network specifically. Ober et al. [23] point out the anonymity of the Bitcoin network increases overtime. Reid et al. [26] indicate that part of the anonymity of known nodes can be discovered using proper analysis tools with available information online. Koshy et al. [12] demonstrate the possibility to link the Bitcoin addresses with IPs only based on

transaction relay traffic. Biryukov et al. [2] deanonymize the Bitcoin network by identifying nodes the target nodes connect to. Dandelion designed by [15] can avoid privacy disclosure by spreading message through a random line and hop then spreading using diffusion to the whole network.

1.3 Organization of the Thesis

This thesis contains six chapters and the rest of the chapters are structured as follows.

In Chapter 2, the background knowledge of blockchain and Lightning Network is provided.

In Chapter 3, data collecting and processing methods are introduced. The way of constructing network graphs is also presented.

In Chapter 4, two important properties of the payment network are measured based on its topology. In Section 4.1, the routing efficiency is studied. In Section 4.2, the network resilience when some important nodes are under attack is evaluated.

In Chapter 5, the behavior of payment channels are investigated.

In Chapter 6, the entire thesis is concluded.

Chapter 2

Background

The purpose of this chapter is to provide the reader with a general understanding of blockchain technology and the Lightning Network (LN). We start by introducing Bitcoin and principles of blockchain. Then, we talk about some current issues of blockchain and a few proposed solutions, especially for the scalability issue. Finally, we focus on one promising solution - LN, which is the study object of this thesis.

2.1 Bitcoin Blockchain

2.1.1 Bitcoin Blockchain Key Principles

Bitcoin is the first electronic cash system where mutually distrusting peers can trade, without relying on a trusted third party, such as a bank. The fundamental infrastructure of Bitcoin is blockchain, a public distributed ledger maintained by the community to record and verify all the transactions.

Paying with Bitcoin on blockchain has attracted more and more attention as it has several huge advantages. First, the system is purely peer to peer so no banks or financial intermediaries can interrupt user transactions or freeze Bitcoin addresses. Second, it provides privacy as Bitcoin addresses are anonymous and not associated with personal identities. Third, standard wire transfers and foreign purchase typically involve fees and exchange costs, since Bitcoin transactions don't go through

intermediary institutions or needs to be exchanged, the fee is relatively small. This is particularly helpful for international trade.

However, constructing such a distributed ledger is not trivial. First, it has to be tamper-proof so nobody can modify transaction histories. Second, it needs to be agreed by the community so peers can reach consensus on the states. Third, it must tolerate some evil nodes that are not following the system protocol as long as the majority are honest. Next we explain how these features are ensured from the structure of blockchain and the way it works.

As shown in Figure 2.1, blockchain is represented by a chain of blocks. blocks can be composed of the block header and the block body which includes a list of transactions. The block header contains various fields like a version number used to track software or protocol upgrades, a merkle root as the hash root of the block's transactions which makes it easier to verify the transactions, a timestamp, a nonce and difficulty target used for consensus algorithm and a reference to the hash of the previous block, serving as a backward pointer (more details can be found blow). Thus, if the data in any given block is altered, so does the hash of this given block, then so does the next block as it stores this hash and so on. Therefore, no data can be modified without changing all subsequent blocks.

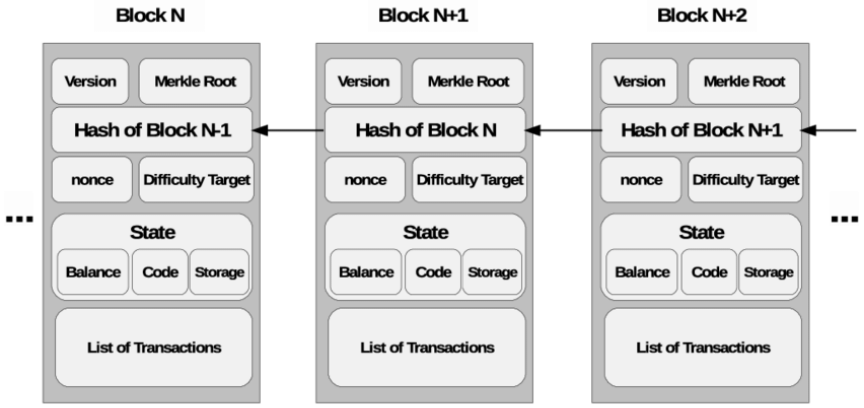


Figure 2.1: Blockchain structure

Source: ResearchGate

With this chain-like structure, the next issue that needs to be addressed is by whom and when new blocks are added to the tail of the blockchain. Here we first explain the hashes. A hash is a string of random numbers generated by putting a given set of data into a deterministic hash function (in Bitcoin is SHA-256). The hash function is one-way as we can not recover the original data from the hash results.

In Bitcoin blockchain, a block is "mined" - added to the blockchain if its hash is smaller than some target (the level of difficulty). Once a node first makes a block with a valid hash, it broadcasts to the network and others will start mining on top of this block. How can nodes alter their block hashes? Well, they include an integer called "nonce" in their blocks and try different nonce values until the block hash is below the target. Due to the random nature of hash, the smaller the target, the more computation nodes need to do. In Bitcoin blockchain, the target is adjusted so that on average a block is generated every ten minutes, note the block proposer can be anyone and is non-predictable. Since the mining process is computationally expensive and requires a lot of work, the consensus algorithm is called Proof of Work (PoW).

With small chances, two miners can find a block at nearly the same time, which will cause the blockchain to diverge into two potential paths forward (the so-called "fork"). The fork is resolved when subsequent blocks are added and one of the chains becomes longer than the alternatives. The network then abandons the blocks that are not in the longest chain (i.e., longest chain rule). This also explains why the block interval is ten minutes instead of seconds. Due to network latency, the smaller the block interval, the more likely duplicate blocks are generated before miners receive the latest block.

In summary, mining is a competitive process. It's extremely difficult to alter an existing block in the chain, since such alteration would require huge computation power to re-mine all subsequent blocks.

2.1.2 Challenges of Bitcoin Blockchain

Though blockchain has achieved great success, it has some issues and limitations.

(a) **Energy consumption:** In PoW, the mining process costs vast amounts of

electricity to compute hash puzzles. It is estimated that the Bitcoin's annual electricity consumption is 70 TWh, which can power 6.7 million U.S. households according to *Bitcoin Energy Consumption Index*[1].

(b) **Latency issue:** Network latency is the amount of time it takes from the creation of a transaction until the first confirmation of it being included in a block. Moreover, to avoid the situation where the block is abandoned due to fork, it is recommended to wait for some longer time. Usually 6 confirmation blocks will significantly reduce the probability that the branch is outcompeted by a conflicting branch. However, this also greatly increased the network latency as users have to wait around 1 hour to make sure their transaction is valid.

(c) **Transaction costs:** In Bitcoin blockchain, miners spend a lot of efforts (e.g., computing power and energy) to mine blocks for a financial reward: with every block added to the blockchain comes a bounty called a block reward, as well as all fees sent with the transactions that were confirmed and included in the block. As a result, the current Bitcoin blockchain network does not favor micropayments because the fee might even exceed the payment itself.

(d) **Scalability issue:** From the above introduction, we can see that the Bitcoin blockchain network can only support very limited number of transactions. While this was enough at the very beginning, the system has been congested for a few years now. For further development of Bitcoin blockchain network, the scalability performance must be improved.

Of all the issues described above, throughput matters the most to a payment network. Current scaling proposals can be roughly divided into three categories: (i) Replacing the underlying consensus algorithms (i.e., PoW) to reduce the block generation interval. A popular variant is Proof of Stake (PoS), which however, raises some other issues like censorship. (ii) Changing the single chain-like structure to parallel chain or directed acyclic graph (DAG) to support more transactions. But these graph structures need to address conflicting transactions as transactions

[1] https://digiconomist.net/bitcoin-energy-consumption

are not ordered linearly. (iii) Developing "Layer 2" solutions which removes most transaction from the blockchain to off-chain.

In this thesis, we focus on one particular "Layer 2" solution - Lightning Network (LN). The idea behind is quite intuitive: we don't really need to keep a record of every single transaction on blockchain. Instead, we only need to publish the final states after two parties have completed multiple transactions while the intermediate transactions are recorded locally (in the so-called channel). By doing so, the advantage is big. First, it's simple and secure. We only need to handle the off-chain transactions, the security is still ensured by the underlying PoW. Second, transactions between peers in their channels are settled instantly and the fees are negligible since they don't enter blockchain. What's more important is that network throughput can be greatly improved with the help of channels.

It is natural to ask, how are channels established and how are they used in LN? We will answer these questions in Section 2.2.

2.2 Lightning Network

2.2.1 Definitions

Below are a few important definitions in Lightning Network (LN) that will be used in this thesis.

- **Channel:** a communication channel that allows two parties to make any secure payments between each other in exchange for making only a few transactions on the blockchain.
- **Contract:** an agreement between two or more entities to use Bitcoin transactions in a certain way, usually a way that allows Bitcoin's automated consensus to enforce some or all terms in the contract. Often called a smart contract.
- **Pre-image/R:** data input into a hash function, which produces a hash of the pre-image. Inputting the same pre-image into the same hash function will always produce the same hash; Lightning uses this feature to create hash locks.

- **Hash Lock:** an encumbrance to a transaction output that requires the pre-image used to generate a particular hash be provided in order to spend the output. In Lightning, this is used to allow payments to be routable without needing to trust the intermediaries.
- **HTLC:** (Hashed TimeLocked Contract) a contract such as that used in a Lightning Channel where both a hash lock and a time lock are used, the hash lock being used to allow Alice to route payments to Bob even through a Carol that neither of them trust, and the time lock being used to prevent Carol from stealing back any payments he made to Alice within the channel (provided Alice enforces the contract).
- **Intermediary:** When Bob has one channel open with Alice and another channel open with Charlie, Bob can serve as an intermediary for transferring payments between Alice and Charlie. With Lightning payments being secured with a hash lock, Bob can't steal the payment from Alice to Charlie when it travels through Bob's node. Lightning payments can securely travel through a theoretically unlimited number of intermediaries.
- **Multisig:** a transaction output that requires signatures from at least one of a set of two or more different private keys. Used in Lightning to give both Alice and Bob control over their individual funds within a channel by requiring both of them sign commitment transactions.
- **Unilateral:** any action performed by only one of the participants in a channel without requesting or needing permission from the other participant. Lightning allows channels to be closed unilaterally (so Alice can close the channel by herself if Bob becomes unresponsive) and attempted fraud can be penalized unilaterally (so Alice can take any bitcoins Carol tried to steal when he broadcast an old commitment transaction).

2.2.2 Payments in Lightning Network

The process of creating a payment channel in LN is shown in Figure 2.2. Alice opens a channel with Bob by sending a deposit of 1 BTC to the 2-of-2 multi-signature address on the blockchain. The fund at the multi-signature address can

only be used with both parties' signature. Besides, the deposit is locked for some designated time (i.e., channel life) so in the meanwhile the two users can send multiple payments to each other. For example, in Figure 2.2, Alice first sends 0.3 BTC to Bob, Bob then sends 0.1 BTC back to Alice, and Alice again sends 0.4 BTC to Bob. These balance updates are settled instantly as they do not need to enter the blockchain. Moreover, each state is signed by both parties and updated in the offline channel in the form of contracts. The contract also includes a timestamp to ensure the local transaction order.

Finally, Alice and Bob can close this channel collaboratively by publishing the latest state (i.e., Alice 0.4 BTC, Bob 0.6 BTC) on the blockchain and claim their money from the multi-signature address. If one party, say Alice is malicious and tries to publish a previous state (e.g., Alice 0.8 BTC, Bob 0.2 BTC), Bob can just send the latest state to the blockchain as a proof and take away all Alice's money in the channel as a punishment. The channel can also be closed unilaterally by any party, just by submitting the latest state at the closing time to the blockchain.

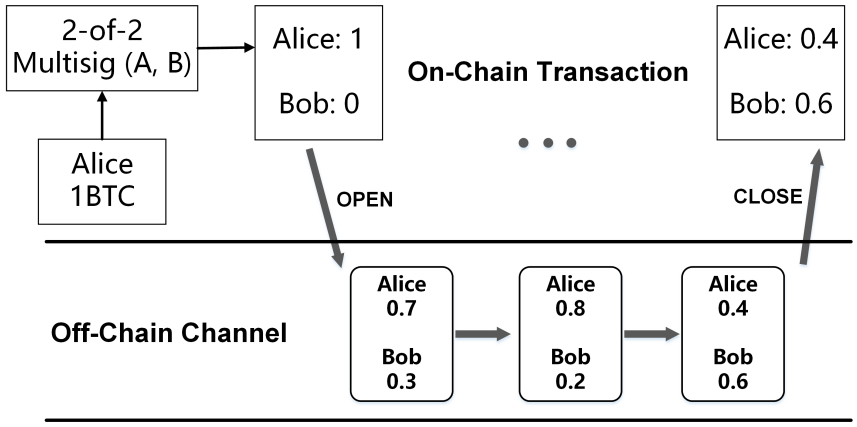


Figure 2.2: Alice-Bob payment channel

The bi-directional payment channel can only ensure secure payments between its two peers. If two parties don't have a direct channel, they can still make transactions via payment routing using HTLC. An HTLC claims that the receiver can only pull money from the sender if he can produce an unknown data R from a known hash H within some lock time. Figure 2.3 shows an example. Alice wants to pay Bob 1 BTC without opening a new channel with him. If Alice just sends money to

Carol, there is no enforcement for Carol to forward the money to Bob. With HTLC, Bob first generates a pair of H and secret R, and shows Alice H. Then Alice can make an HTLC using the H and send it to Carol, claiming that Carol can pull 1 BTC from Alice if she can show Alice the corresponding R in 3 days. Carol then makes another HTLC through H, claiming that Bob can pull 1 BTC from Carol if he can provide R in 1 day. Thus, Bob can get the money from Carol by exposing secret R in time and Carol can get the money from Alice in the same way. The HTLC expires if the receiver fails to provide R in the pre-defined time. This decreasing lock time from 3 days to 1 day ensures that Bob cannot steal money from Carol by showing R and pulling money from Carol after Carol's HTLC with Alice expires.

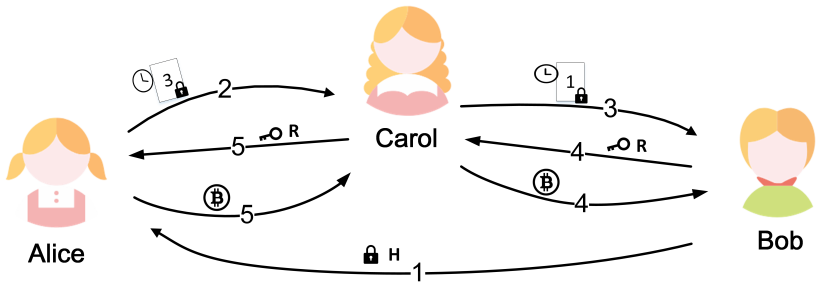


Figure 2.3: HTLC in payment routing

Chapter 3

Data Collection and Abstraction

The first step of analyzing LN is to collect node and channel data. In this chapter, we construct a snapshot graph G to represent LN for further analysis including network structure and graph metrics. We also study the network performance through G and the communication performance through pair channel characteristics. Figure 3.1 is an overview of our method.

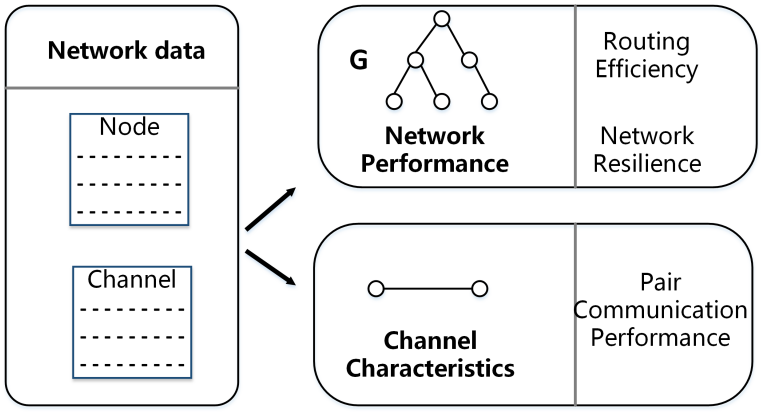


Figure 3.1: An overview of our work

3.1 Data Collection of LN

The network data is composed of information of nodes and channels. Node information includes node public key, node alias, join time and node capacity, which are explained as follows.

- ***Node public key:*** a unique 66-byte long string composed of number and letter which represents the node identity. It is generated from user's private key and is needed to open or close the channel on LN.
- ***Node alias:*** a user-defined name of the node for ease of use.
- ***Node join time***: the time a user joins LN by opening the first channel.
- ***Node capacity***: the sum of capacities of all channels on that node, regardless of the channel was established from the node or to it.

Channel information includes channel id, channel nodes, channel open time and close time, channel open fee and close fee, channel capacity. Detailed information are as follows.

- ***Channel id:*** a set of numbers of length 16 which represents channel identity.
- ***Channel nodes:*** the two owners of the channel.
- ***Channel open/close time:*** the time the open/close channel transaction is published on the blockchain.
- ***Channel open/close fee:*** the fee paid for processing the open/close channel transaction on the blockchain.
- ***Channel capacity:*** the deposit of the channel at opening time.

Note that current channel balance is only visible to owners of the channel for privacy reasons.

The current LN is a P2P network that uses a gossip protocol to discover and disseminate information of nodes and channels. Messages include node announcement message, channel announcement message and channel update message. We explain them below.

- ***Node announcement message:*** it broadcasts or updates node information like public key, port and addresses so that other nodes can connect to it.
- ***Channel announcement and update message:*** it contains information about the channel creation and maintenance so that a node can find routes to its desired destination.

Unlike Bitcoin network where every node has a copy of the full transaction record, there is no such record in LN. Hence, it is hardly possible to monitor the global network. Instead, we can use passive application-level monitoring to retrieve network information. That is, we can set up multiple nodes in LN and connect them to other nodes. Our nodes can participate in message exchange and log messages received from their peers. Typically the more nodes we are connected to, the more completed views we have.

Except for setting up our own listening nodes, we can crawl the network data shared by existing nodes. There are several such websites like https://hashxp.org/lightning/, https://1ml.com and https://graph.lndexplorer.com/. They serve as LN search engines and publish their gathered view of nodes and channels from multiple nodes. In this project, we crawled and processed the data until April 1, 2019 from https://hashxp.org/lightning/ as it has the most comprehensive information. One of the listening node it uses is "rompert.com", which is one of the top 5 nodes having most channels.

3.2 Graph Construction

To investigate the structure and performance of LN, we build an *undirected* snapshot graph G. We ignore channel direction in our graph construction because direction is often unknown and changes all the time. Specifically, a channel is from A to B if user A has a balance. A channel can be bidirectional if both users have their balance. However, user balance state is updated via HTLC off the blockchain and is not broadcasted. We can only obtain the initial balance state when channel was funded and the latest balance assignment when channel was closed. Therefore, at any other time between the open time and close time, the balance assignment in the channel could be arbitrary.

As another reason, our work mainly focuses on the connectivity of the network rather than the specific payment direction. In cases where channel direction cannot be neglected like channel routing, we can assume all channel deposits are assigned to the user who needs to make payments to the other side, which achieves the maximal payment flow in that direction. Under this assumption, we can consider the network as *undirected* again. Below is the construction of our weighted undirected graph G at any specific time.

Definition: $G = (V, E, \omega)$, where V is a set of nodes, E is a set of edges, ω is weight function and $E = \{(v_i, v_j) | v_i, v_j \in V\}$. An edge represents there are one or more active channels between node v_i and v_j at a given time. Here $\omega : E \to \mathbb{R}_+$ associates each edge with a weight, which is the biggest payment amount between node v_i and v_j at that time.

To construct G, we filter existing nodes and active channels at a given time in the following manner. For each channel, if there is no edge between the two nodes, we add one and set the edge's weight to be the channel capacity. If an edge already exists, we increase its weight by the channel capacity. Graph G makes it convenient for us to study network connectivity or node reachability as it abstracts the "is-connected" relationship between nodes.

3.3 Network Structure

We study some descriptive statistics of the network graph G at three specific time instances: June 20, 2018 at 12:00, December 5, 2018 at 12:00 and April 1, 2019 at 12:00. More details can be found in Table 3.1. We can see around 30% of the nodes in G are isolated. Probably those nodes just joined LN out of curiosity and did not open new channels after their first one expired. We remove those isolated nodes as they do not communicate in the network and consider the resulting graph denoted by G'.

Figure 3.2a and Figure 3.2b demonstrate the degree distribution of G' on April 1, 2019. Figure 3.2a shows that it follows the power law, indicating there are many small-degree nodes and a few large-degree nodes [20]. The fitting line is $y \propto x^{-\alpha}$. The larger the α, the less variable of a node's degree. In G' node degree represents the number of direct neighbors. Note that at this time the node with the most

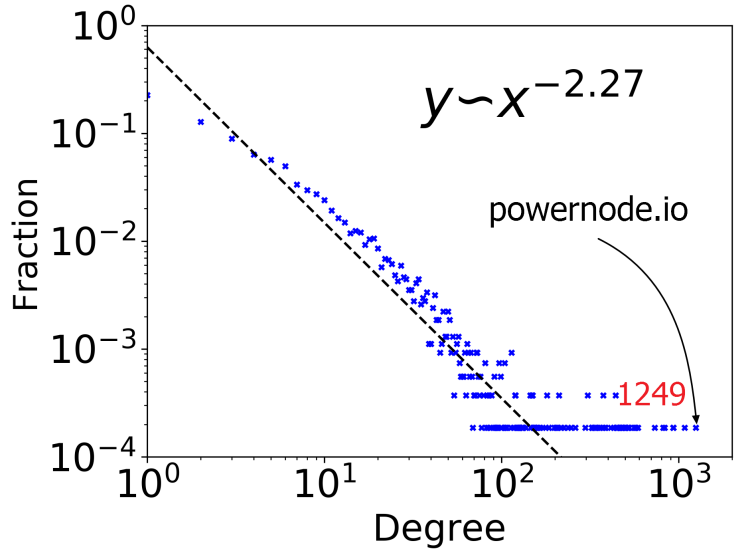
Table 3.1: Statistic of G and G'_{LCC}

Time	G				G'_{LCC}							
	nodes	channels	capacity	isolates	nodes	% of G	channels	% of G	capacity	% of G	cluster	assortativity
2018-06-20	2,604	7,686	25.41	739	1,845	70.85	7,676	99.87	25.39	99.92	0.20	-0.30
2018-12-05	4,411	15,913	462.70	1,587	2,804	63.57	15,903	99.94	462.68	99.99	0.23	-0.25
2019-04-01	7,796	41,705	1,102.66	2,409	5,345	68.56	41,679	99.94	1,102.22	99.96	0.28	-0.31

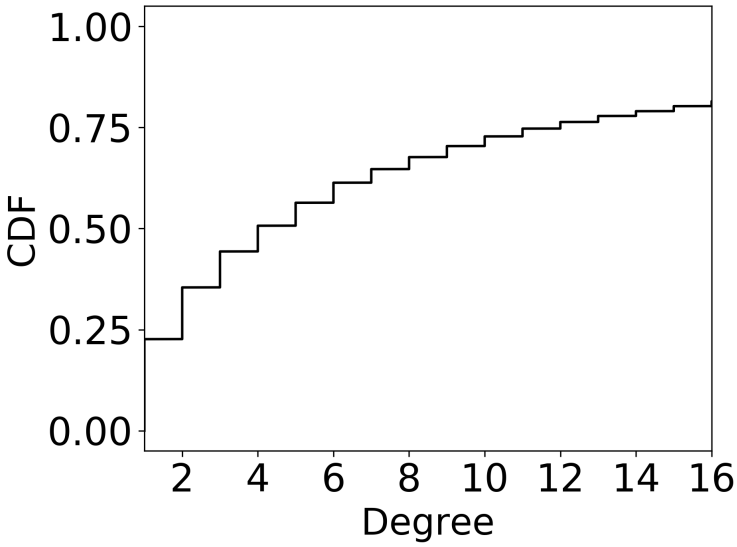
neighbors is "powernode.io", which has 1,249 direct neighbors. More specifically, Figure 3.2b displays the empirical cumulative distribution of node degrees. The result shows that 23% nodes have only one edge and 73% nodes have less than 10 edges. This means that most nodes in LN can only send direct payments to very few neighbors, which is consistent with Figure 3.2a.

In order to obtain more insights, we consider G''s largest connected component (LCC) denoted by G'_{LCC}. Recall that connected component (CC) is a subgraph where any two nodes u, v are connected to each other by certain paths [27]. Note that nodes in different CC's cannot reach each other. From Table 3.1 we can see that G'_{LCC} contains about 99% non-isolated nodes, 99.9% edges and network capacity of G'. This indicates that G'_{LCC} is a giant component that fills most of the network, while the rest of the network is divided into a large number of small CC's. A possible reason is that a majority of nodes in LN want to be able to communicate to each other, therefore they need to be part of LCC. There are very few nodes outside G'_{LCC} in G', possibly these nodes just use LN for making direct point-to-point payments to a few specific neighbors. Hence, we focus on G'_{LCC} in terms of graph metrics and also in Chapter 4 when analyzing network connectivity performance.

Table 3.1 provides some metrics of G'_{LCC}. The clustering coefficient of G'_{LCC} is large (i.e, 0.20, 0.23, 0.28), revealing that if two users A and B both have channels with C, A and B are likely to have a channel with each other. The three clustering coefficients on different days are very close, indicating that the tendency barely changes with time. The assortativity coefficient [19] reveals preference for nodes to connect to similar nodes. Here it is negative, indicating a large degree user tends to open channel with a small degree user rather than a large degree one. A possible reason is that famous users (large degree nodes) tend to serve small users (small degree nodes) as intermediaries.



(a) Power law fitting



(b) Cumulative distribution of degrees

Figure 3.2: Degree distribution

Chapter 4

Network Performance Analysis

In this chapter we evaluate the performance of LN based on graph G. We first study network efficiency in terms of routing success rate, then network resilience under attack, both on April 1, 2019 at 12:00. In addition, we study the performance evolution over time.

4.1 Network Routing Performance

Routing is the process of selecting a path between two nodes in the network. Since 99.9% of the edges and deposits are contained in G'_{LCC} as shown in Section 3.3, nodes and edges outside G'_{LCC} barely participate in routing. So we focus on G'_{LCC} in this section.

Routing is very important in LN. Given the analysis of degree distribution in Section 3.3, most nodes have limited neighbors so they can make direct payments with very few nodes. However, two nodes might not want to open a channel when they need to make transactions. The reason is that adding edges (i.e., opening channels) in LN costs money so it's wasteful to open a new channel that will not be frequently used. Thus it's necessary to use existing channels, especially for small nodes (i.e., nodes with very small edges or capacities). Fortunately, payments can be routed by using HTLC to ensure state consistency along the path as explained before.

We evaluate the network routing performance from three perspectives: network

connectivity, channel capacity and routing efficiency, which evaluate the routing path lengths, routing transaction amount and routing success rate, respectively.

4.1.1 Effective Eccentricity

Network routing performance is affected by the connectivity of the network. To analyze the length of shortest paths between a node and others, we define effective eccentricity as the least number of hops for a node to reach certain fraction of the network. Figure 4.1 gives the effective eccentrity of G'_{LCC} with fraction $\alpha = 0.7, 0.9, 1$ on April 1, 2019 at 12:00. We can learn that effective eccentricity grows with α as it typically takes more steps to reach more nodes. The diameter of G'_{LCC} is 11, which is the maximum shortest path lengths of the graph. Around 87% of the nodes can reach all other nodes within 8 steps, around 90% nodes can reach 90% of the network in 4 steps and around 72% nodes can reach 70% of the network in 3 steps. This result suggests the network has the potential to perform efficient routing.

4.1.2 Channel Capacity Distribution

The capacity of a sequence of edges is restricted to the minimal edge capacity along the path. To perform routing, every edge on the path should have enough deposits. So we will investigate the channel capacity distribution. Before further analysis, we address the problem of channel direction and clarify a subtle difference between channel capacities and edge capacities.

As explained in Section 3.2, we neglect the channel direction by assuming all the channel balance belongs to the payment sender. Hence, we consider the *best* routing performance. Recall the construction of G in Section 3.2, the edge weight is the sum of capacities of all the channels between the same two nodes. To achieve best routing performance, the maximal routing amount should depend on the edge capacity instead of the channel capacity. That's to say, two nodes can use all their channels for routing at the same time. The reason is that multiple channels between two nodes can use several HTLC's that share the same secret key R to

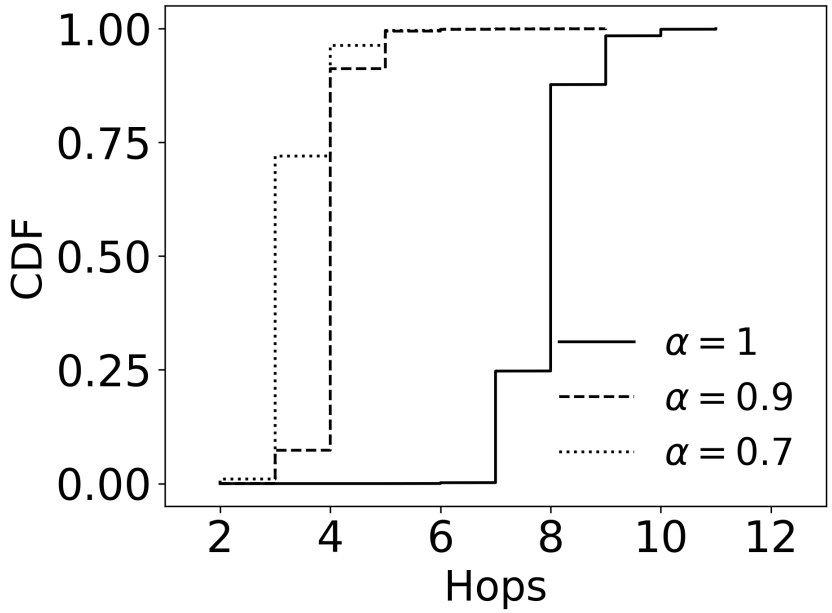


Figure 4.1: Cumulative distribution of effective eccentricity

ensure money in these channels is pulled correctly by the receiver.

Figure 4.2 demonstrates the empirical cumulative distribution (CDF) of channel capacity and edge capacity. The two CDFs are extremely close, indicating that the number of node pairs having multiple channels is really small. Interestingly, we observe that the capacity does not follow uniform distribution. Instead, there are some obvious jumps at certain capacities including $0.005, 0.05, 0.1, \ldots, 0.2$, revealing that nodes like to choose these amounts when opening channels, probably for ease and simplicity. We can see 90% of the channel capacity is below 0.10 BTC, indicating that for a payment amount bigger than 0.10 BTC, routing is unlikely to be successful as only 10% channels are available.

4.1.3 Network Routing Efficiency

Before explaining network routing efficiency, we define the qualified component (QC) as a CC of the original network after removing all edges whose capacity is less

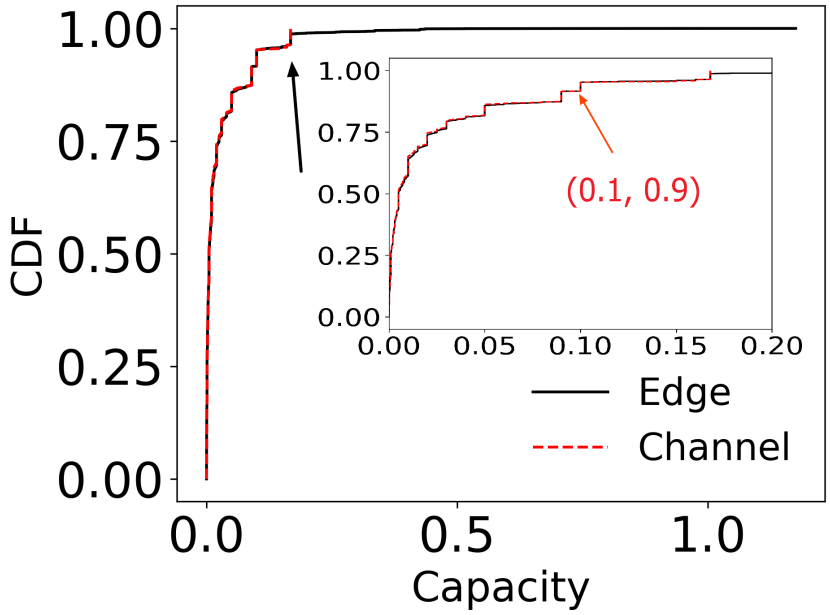


Figure 4.2: Cumulative distribution of channel capacity

than the targeted routing amount. Of all the QC's, the largest one and second largest one (in terms of node numbers) is denoted by largest qualified component (LQC) and SQC. Here the LQC and SQC of G'_{LCC} is represented by G'_{LQC} and G'_{SQC}. As a simple example, in Figure 4.3, we want to perform a routing of 0.005 BTC and remove edges whose weights are less than 0.005. Then the LQC is formed by the nodes in the dashed circle, and the SQC is formed by the green nodes. Note that the QC's depend on the routing amount.

For any two nodes in the same QC, a path can be found with all edge weights greater than or equal to the routing amount. That's to say, a routing of a target amount is always successful in the QC. To consider the best routing performance of LN, we focus on G'_{LQC}. Thus network routing efficiency is defined as the fraction of G'_{LQC} size over the network size. The routing efficiency reflects the success rate of a routing. In Figure 4.3, the network routing efficiency of 0.005 BTC is $5/11$.

Figure 4.4 gives the routing efficiency under different amounts. The horizontal axis is the routing amount percentile β of all edge weights. The black line

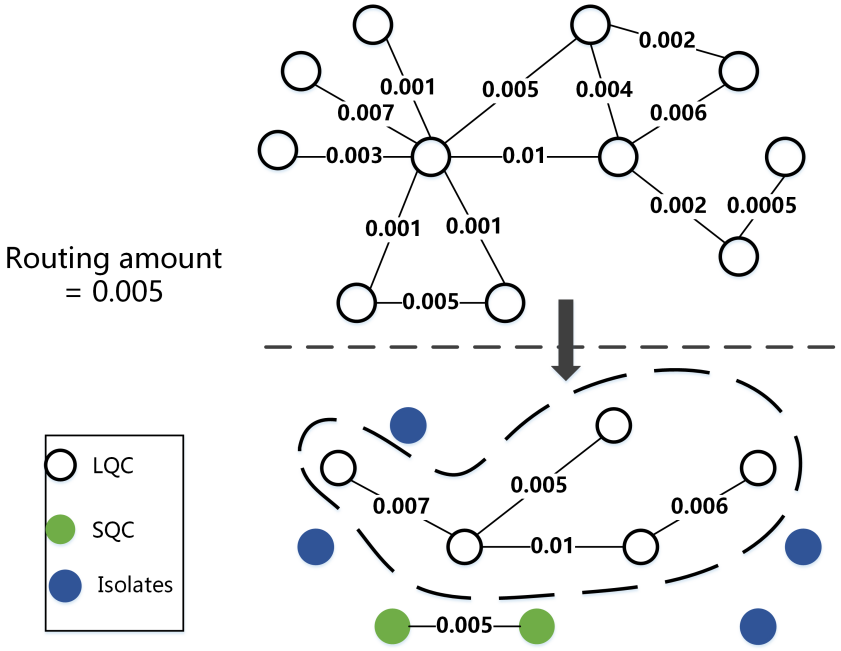


Figure 4.3: An example of routing efficiency

with circle markers represents the efficiency. We also plot the G'_{SQC} size fraction using the red line with triangle markers. The G'_{SQC} is close to 0, indicating that although nodes in other smaller QC's can also perform routing inside themselves, it's reasonable to consider only G'_{LQC} for network routing efficiency. For a routing with amount β of 50th percentile, the successful rate is 0.59. The efficiency is approximately inversely proportional to amount except for some drastic change at some points. This matches our observation in Section 4.1.2, that users prefer some specific amounts as channel deposits.

4.1.4 Routing Evolution

Our analysis above only studies the routing performance on April 1, 2019 at 12:00. To evaluate the evolution of routing performance over time, we plot the routing efficiency from April, 2018 to April, 2019 in Figure 4.5a. The blue, orange and green bars denote routing efficiency with routing amount β of 20th, 50th and 80th

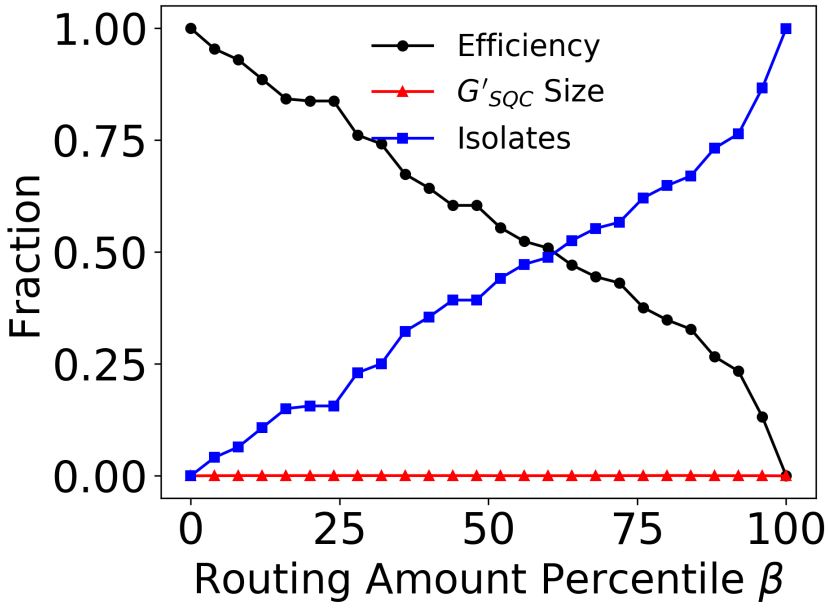


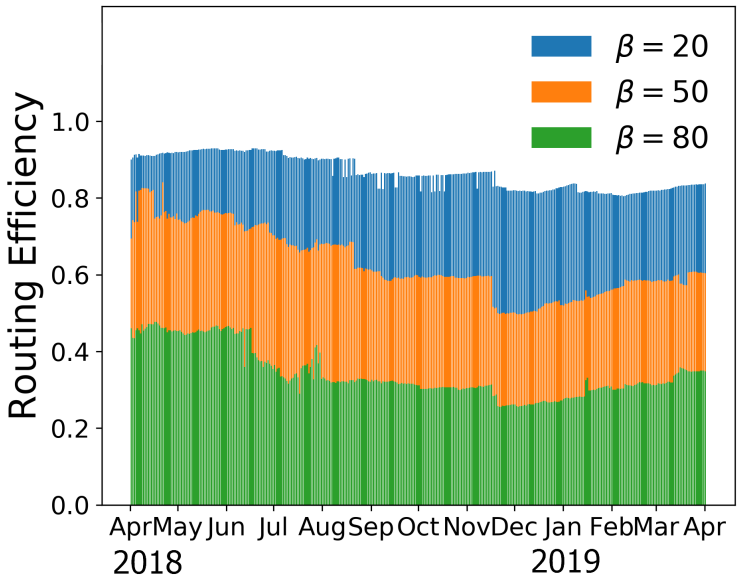
Figure 4.4: Network routing efficiency

percentile, respectively. We can see the routing efficiency declines over time in year 2018 and rises in 2019. A possible reason is that exchange markets or hubs emerge or play a more important role in LN.

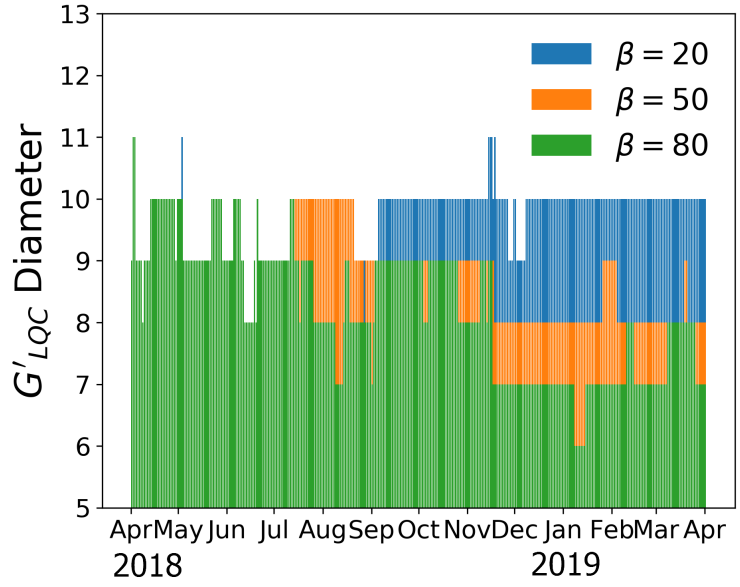
Figure 4.5b demonstrates the diameter of G'_{LQC} with three different routing amount β's, which represents the longest routing path length. The result shows that G'_{LQC} diameter fluctuates from 6 to 11, and the smaller the β, the larger the G'_{LQC} diameter. This shows that the G'_{LQC} under a large β is more densely connected compared to the G'_{LQC} with a small β (i.e., nodes with higher channel capacities are more closely connected).

4.2 Network Resilience Under Attack

Network resilience is an important factor which reflects network's anti-attack ability. There exists some "big nodes" (e.g., nodes that have many channels) in LN. These nodes can serve as exchange markets and make it easier for small nodes to



(a) Routing efficiency



(b) G'_{LQC} diameter for routing

Figure 4.5: Network routing efficiency evolution

use LN. (This is because small users cannot afford to open channels with every other user as explained in Section 4.1.) Meanwhile, such important nodes may influence network resilience. The problem is, how does the network perform when under attack (i.e., taking down important nodes)?

To answer this question, we first rank nodes using three common algorithms in graph theory, then evaluate which algorithm describes node importance the best through network efficiency [13]. Furthermore, we analyze the anti-attack performance in terms of network capacity and G'_{LCC} size.

4.2.1 Identifying Important Nodes

We select three common metrics or algorithms in graph theory that evaluate node importance: closeness centrality, betweeness centrality and PageRank algorithm (PR) [3, 24, 32]. We then compare these three metrics using network efficiency. Network efficiency is the sum of the reciprocal of all shortest path lengths and it measures the efficiency of a network when exchanging information.

Closeness centrality describes how close a node is to all other nodes. It is calculated as the reciprocal of the sum of the length of the shortest paths between the node and all other nodes in the graph. Thus, the more central a node is, the closer it is to all other nodes. Betweeness centrality represents the degree to which nodes stand between each other. A node with higher betweenness centrality would have more control over the network, because information would pass through that node. PageRank algorithm evaluates the number and quality of the edges of a node. It was originally designed as an algorithm to rank web pages and calculated based on the structure of the incoming links.

We can attack a network by randomly or intentionally removing nodes or edges [6, 8]. Note that in LN, channels are kept locally between two users so the attack to a node will not spread out to its neighbors. Hence, we simulate attacks to LN by removing nodes from the most important to the least important one by one. Figure 4.6 shows the network performance under attack with nodes evaluated using the above three metrics. The horizontal axis denotes the fraction of removed nodes γ and the vertical axis denotes the resulting network efficiency. We can see that

efficiency decreases the fastest when nodes are ranked by PageRank algorithm, revealing that PageRank algorithm describes nodes importance the best. Therefore, we use PageRank algorithm to rank nodes in the rest of this section.

Table 4.1 lists the 10 most important nodes evaluated by PageRank algorithm. For the ease of presentation, we use the first three bytes of a node's public key to denote it. The aliases of these 10 nodes are listed below:

- powernode.io
- LightningPowerUsers.com
- rompert.com
- ACINQ
- 1ML.comnodeALPHA
- ln1.satoshilabs.com
- BitMEXResearch
- SLEEPYARK-v0.7.0
- tippin.me
- BOLTENING.club

We can see the channel number is not necessarily consistent with the PageRank value. A possible reason is that a node connected to many unimportant nodes might be less influential than a node connected to a few but important nodes. The third and fourth rows list the node capacity and node channel numbers. The last row shows the category of nodes. Some play the role of exchange markets and encourage others to connect to them. Some collect LN information and serve as search and analysis engines. The tippin.me [1] is a project node that helps users to receive tips by allowing others to use their channels in routing.

[1] https://tippin.me/

Table 4.1: Top 10 Important Nodes

pubkey	02809e	0331f8	02ad6f	03864e	021789	0279c2	039503	02f672	03c2ab	02529d
pr	0.0129	0.0118	0.0117	0.0114	0.0094	0.0085	0.0075	0.0067	0.0063	0.0058
capacity/BTC	4.4573	25.2234	9.9129	32.4169	7.6270	27.0349	34.8319	1.3005	10.4720	9.6110
channel	1,249	935	1,082	929	833	819	732	211	589	568
category	exchange	exchange	search	company	search	research	research	/	company	research

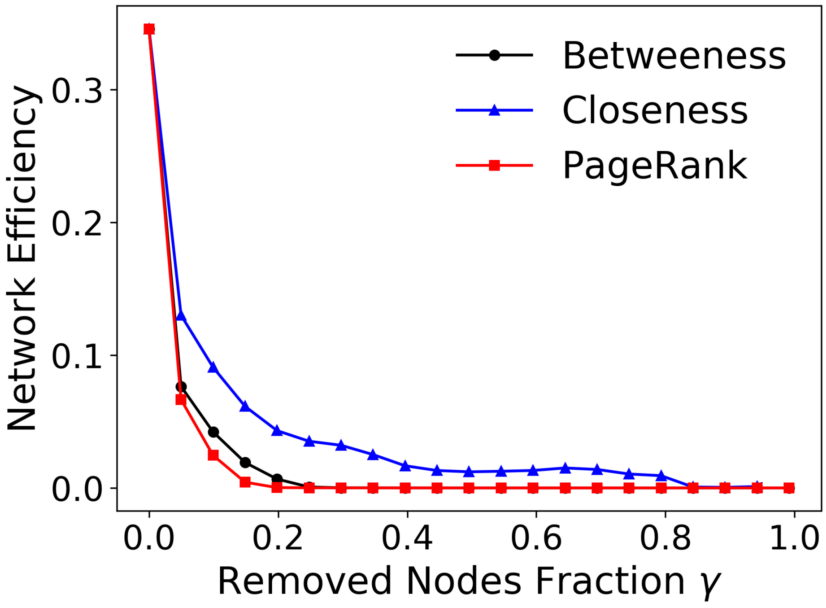


Figure 4.6: Node importance evaluation

4.2.2 Network Anti-Attack Performance

We evaluate LN resilience from network capacity and G'_{LCC} size. Network capacity represents the total amount of money the network holds and G'_{LCC} size reflects the best routing efficiency (i.e., the routing amount is below the minimum channel capacity so all edges are qualified). Before further analysis, we elaborate the detailed situation of attacks to LN.

When node A is taken down, it becomes uncooperative (i.e., unresponsive or fraudulent). Here taken down means that A does not respond because its under DDoS attack or offline, or it is a malicious node that doesn't follow the protocol. If A refuses to response, all of its neighbors will have to wait until their channels expire and then can get their money back. If A claims a fake balance state, the other party can submit a proof (i.e., the last valid HTLC state) to the blockchain and claim the correct state. That's to say, though there will be no actual loss in the end, all of A's channels become unavailable and the deposits are locked for some

time. Thus, it's reasonable for us to simulate an attack to a node by removing all its edges.

Figure 4.7 demonstrates the network capacity and G'_{LCC} size under attack, both normalized using original values. The results show that the network loses around 94% capacity and 46% G'_{LQC} nodes after removing just top 5% nodes and the network almost paralyzes if removing top 20% important nodes. This indicates that LN relies heavily on a few important nodes, which is similar to the "robust yet fragile" structure of the Internet topology [7]. Moreover, we learn that there exists a trade-off between network resilience and the ease of use. That's to say, on the one hand, hubs can make it more efficient for users to send transactions in LN, on the other hand, they reduce the degree of decentralization of the network.

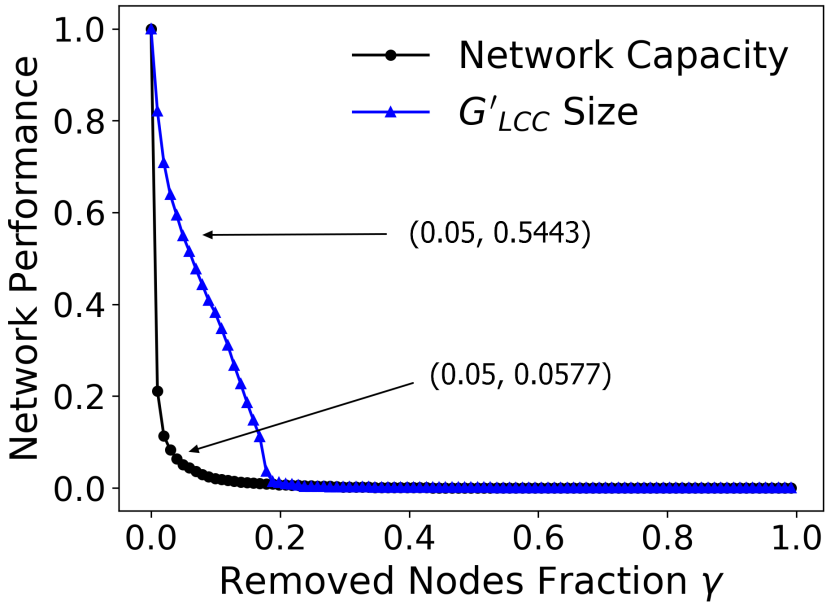


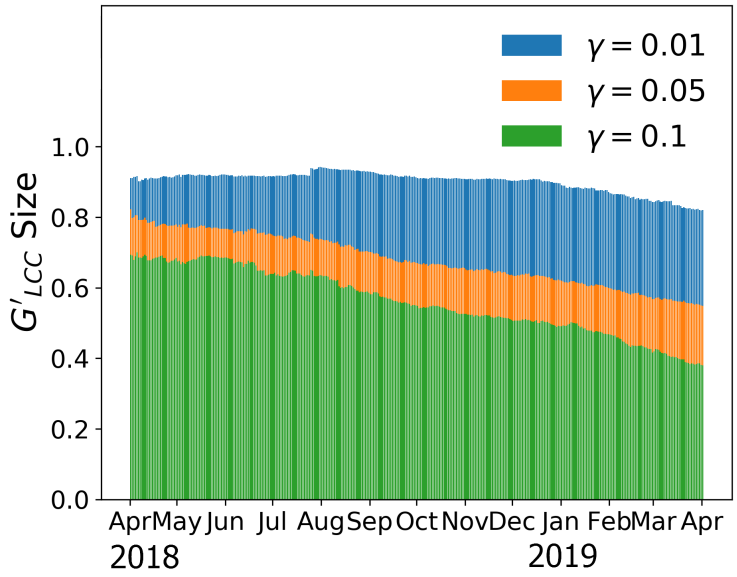
Figure 4.7: Network resilience

4.2.3 Resilience Evolution

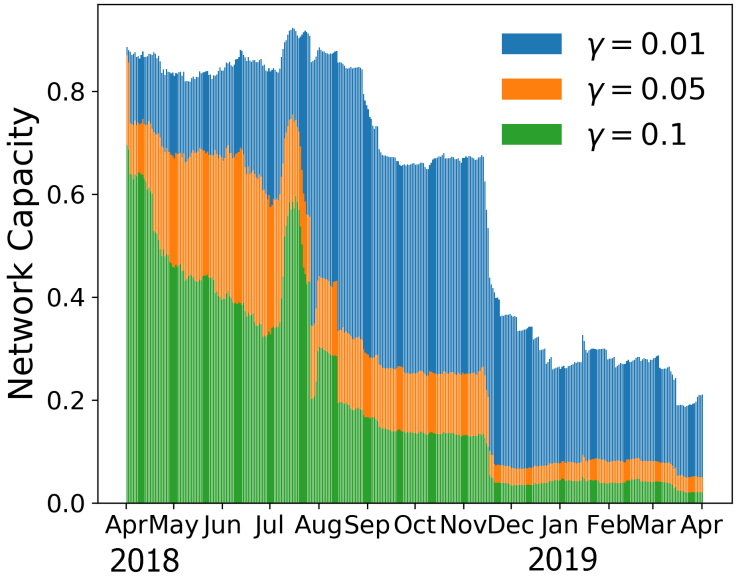
Previously, we focused on the network resilience on April 1, 2019 at 12:00. To

evaluate the evolution of resilience over time, we plot the G'_{LCC} size and network capacity under attack from April, 2018 to April, 2019 in Figure 4.8a and Figure 4.8b. In Figure 4.8a, the blue, orange and green bars denote G'_{LCC} size (normalized using the original network size) after removing top 1%, 5% and 10% nodes, respectively. We can see G'_{LCC} size decreases over time, especially in 2019. Recall in Section 4.1.4, we deduced the arise of of hubs in the same period. This suggests that hubs weaken LN's ability to resist attacks, which is in consistent with our analysis in Section 4.2.2.

Figure 4.8b shows that the network capacity under attack also decreases over time. However, it experienced a clear "up and down" period in mid-July as well as a sudden jump in mid-November. This indicates changes of network structure, especially among important nodes. Actually this reflects appearance of new influential nodes. The LNBIG nodes as mentioned in Chapter 1 started to join LN from November 13, 2018 and injected hundreds of bitcoins into the network rapidly. At first these LNBIG nodes are not connected to existing hubs, which makes them less important when evaluated by PageRank algorithm. Thus they preserve some network capacity when other hubs are removed. The drop in August and mid-November is possibly caused by the strengthen of the connections among important nodes.



(a) G'_{LCC} size under attack



(b) Network capacity under attack

Figure 4.8: Network resilience evolution

Chapter 5

Channel Characteristic Analysis

In this chapter, we move from graph topology to channel characteristics between node pairs. We first analyze some channel descriptive statistics and then study pair communication performance in terms of channel temporal distribution.

5.1 Channel Statistics

Table 5.1 lists the statistics of all collected channels, including channel life, channel capacity and channel costs (i.e., channel open fee and channel close fee). In LN, as long as the two parties of a channel agree to cooperate, the channel can stay open without a mandatory timeout period. The channel can be closed unilaterally or bilaterally. If both parties decide to close the channel (because of channel depletion for example), the channel will be closed immediately. If fraud happens or one party becomes unresponsive, the other party can close the channel unilaterally.

Channel life is the duration from channel open time to channel close time. We can see the average channel life is 29.89 days and the average channel capacity is 0.0196 BTC. Both have a large variance. This is intuitive as channel life and capacity are decided by the two owners of the channel.

Channel open or close fee is the fee of the broadcasted on-chain transaction that opens or closes the channel. The higher the transaction fee a user offers, the more likely miners will add the transaction into their blocks and so the faster the transaction will be confirmed. We can see the channel close fee is a little higher

Table 5.1: Channel Statistics

index	channel life /day	channel capacity /BTC	channel open fee /e-5 BTC	channel close fee /e-5 BTC
average	29.89	0.0196	2.34	2.63
median	12.44	0.0040	1.11	1.67
COV	1,943.65	0.0015	0.00027	0.00037

than the channel open fee, revealing that users tend to pay higher fees to close the channel. Since higher fees can accelerate the process of blockchain transactions, a possible reason is that the channel close process is more urgent than the open process. This is intuitive as users want to close their channels under 3 conditions: the channel is depleted and unavailable; both users have agreed on the final balance and want to claim their Bitcoin on the blockchain; there is a fraud and the honest party needs to declare the true balance states in time. Thus it's better to publish the closing transaction on the blockchain faster.

5.2 Pair Communication Performance

This section studies communication performance among pairs. We first analyze channel number distribution of node pairs (i.e., how many channels a pair of nodes own) and then define communication ability and communication stability. Furthermore, we investigate these two metrics among multi-channel pairs. The three observation time we use are S_1: June 20, 2018 at 12:00, S_2: December 5, 2018 at 12:00 and S_3: April 1, 2019 at 12:00, respectively.

Some node pairs in LN have more than one channels until the observation time and are denoted by multi-channel pairs. Figure 5.1 shows the CDF of all pair channel numbers until S_3. We can see that 85% pairs only have one channel and 11% pairs (i.e., 0.96-0.85) have two channels in total. The top three pairs that have most channels are also listed here. Node BOLTENING.club and 038247 opened 473 channels from February 16, 2019 to March 16, 2019, all with a deposit of 0.0002 BTC. Most of their channels remain open until S_3. Meanwhile, most of the channels among the second and the third pairs are already closed. These

pairs are possibly maintaining multiple channels for testing the network function or for future use, which we can tell from the aliases of the last pair (cln-test-01, cln-test-02).

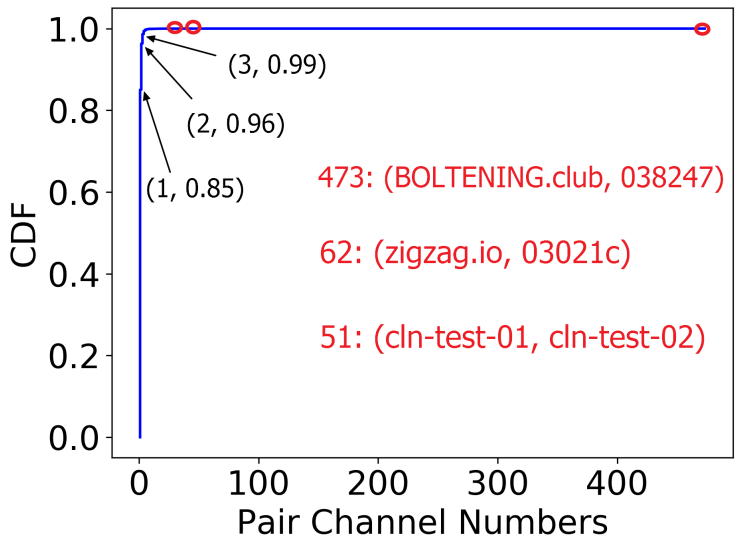


Figure 5.1: Cumulative Distribution of Pair Channel Numbers

5.2.1 Pair Communication Ability

The communication of a pair of nodes depends on all the channels they own. The time interval of these channels can be overlapping or non-overlapping. We define communication ability as the fraction of total time two nodes can communicate (via at least one active channel) over the duration from the moment the two nodes first open a channel to the observation time. Nodes communication availability evaluates the probability that two nodes can make transactions since first connected.

Definition: Pair communication ability, $\rho_a(i,j) = \frac{\sum_{k=1}^{m'_{ij}} |s'_k|}{t_s - t_{o_1}}$, where t_s is the observation time, m_{ij} is the total number of channels opened before t_s between node i and j, t_{o_k} and t_{c_k} is the open and close time of the kth channel between i and j, channel is sorted in ascending order of t_{o_k}, so t_{o_1} is the open time of the first channel

between i and j, $s_k = (t_{o_k}, \min\{t_{c_k}, t_s\}), k = 1, 2 \cdots m_{ij}$, s_k is the time interval of kth channel, $|s_k| = \min(t_{c_k}, ts) - t_{o_k}$, $(s'_1, s'_2 \cdots s'_{m'_{ij}}) = (s_1 \cup s_2 \cdots \cup s_{m_{ij}})$ is the union of all multi-channel intervals.

As a simple example, in Figure 5.2, the communication ability until t_s is $\rho_a(1,2) = \frac{t_{c2}(1,2) - t_{o1}(1,2)}{t_s - t_{o1}(1,2)}$ for node pair $(1,2)$, $\rho_a(1,4) = 1$, $\rho_a(2,3) = \frac{(t_{c1}(2,3) - t_{o1}(2,3)) + (t_{c2}(2,3) - t_{o2}(2,3))}{t_s - t_{o1}(2,3)}$.

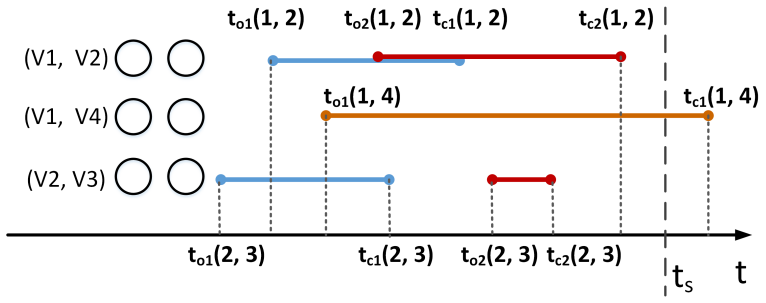


Figure 5.2: Pair Communication Performance

In Figure 5.3a, the solid lines show the CDF of communication ability ρ_a among all pairs. The red, black and blue lines correspond to S_1, S_2 and S_3. We can see the overall ρ_a decreases from S_1 to S_2, indicating that many pairs lose the ability to send payments. It then increases from S_2 to S_3, possibly because some pairs maintained a long-lasting channel. Until S_3, only 60% of the pairs can communicate for more than 30% of the time and about 41% of the pairs can keep communicating all the time (i.e., $\rho_a = 1$) since opening their first channel. This demonstrates that the communication status among pairs is not quite reliable.

5.2.2 Pair Communication Stability

Communication between two nodes is interrupted if their multi-channel intervals are not overlapping (i.e., a new channel is opened after the old channel is closed). To describe the continuous communication ability, we define pair communication stability as the fraction of the longest continuous communication time over the duration from the pair are firstly connected to the observation time. It evaluates the probability that two nodes can continuously make transactions since their first

connection.

Definition: Pair communication stability $\rho_s(i,j) = \frac{\max_k |s'_k|}{t_s - t_{o_1}}$. This is similar to $\rho_a(i,j)$ except that the denominator is the longest unioned channel interval. Hence $\rho_s(i,j) <= \rho_a(i,j)$.

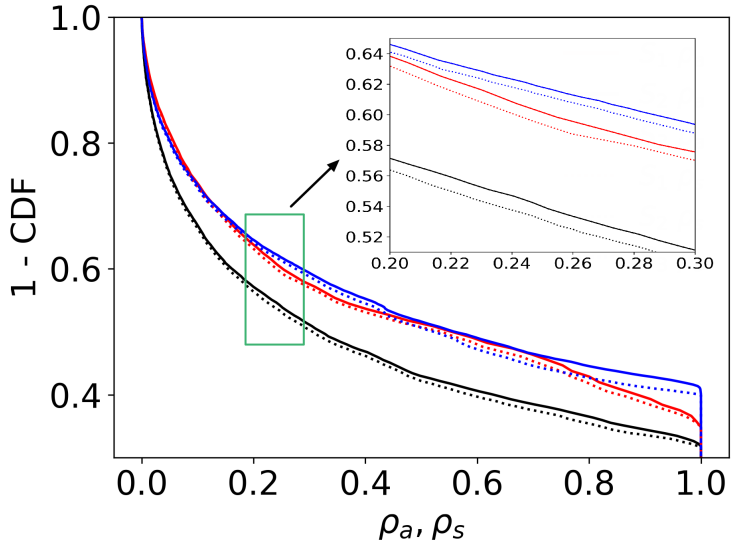
For example, in Figure 5.2, the communication stability until t_s is $\rho_s(1,2) = \frac{t_{c2}(1,2) - t_{o1}(1,2)}{t_s - t_{o1}(1,2)}$ for node pair $(1,2)$, $\rho_s(1,4) = 1$, $\rho_s(2,3) = \frac{\max\{t_{c1}(2,3) - t_{o1}(2,3),\ t_{c2}(2,3) - t_{o2}(2,3)\}}{t_s - t_{o1}(2,3)}$.

The dotted lines in Figure 5.3a demonstrate the CDF of communication stability ρ_s among all pairs until S_1, S_2 and S_3. We can see ρ_s is quite close to ρ_a for the same observation time. This is because most pairs own only one channel as shown in Figure Figure 5.1.

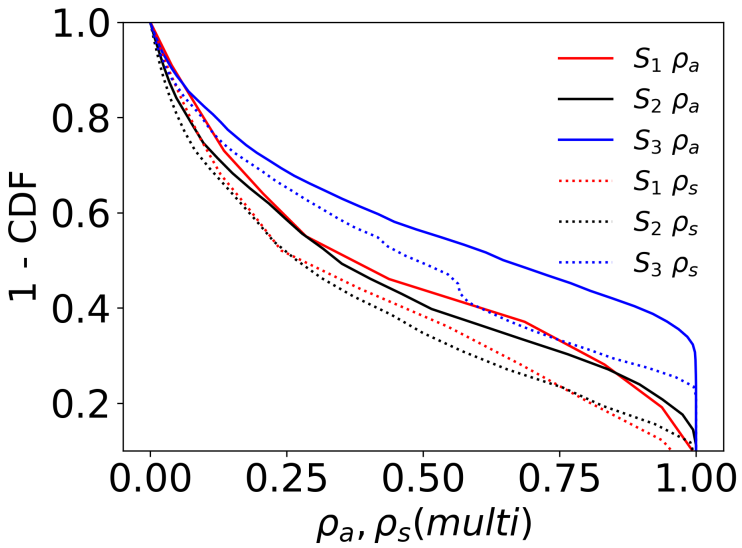
5.2.3 Multi-channel Pair Communication Performance

The characteristics of multi-channel pairs are important. This is because with the development of LN, more and more existing channels will be closed or depleted, thus new channels are needed among node pairs. To obtain further insights about current multi-channel pairs and their channel temporal distribution, we study their ρ_a and ρ_s in Figure 5.3b.

Figure 5.3b shows that ρ_a of multi-channel pairs is significantly higher than ρ_s for the same observation time, indicating their channel intervals are not overlapping. The reason is that $\rho_a = \rho_s$ if multi-channel intervals are overlapping. For example in Figure 5.2, $\rho_a(2,3) > \rho_s(2,3)$ and $\rho_a(1,2) = \rho_s(1,2)$ as the two channels for pair (2, 3) are non-overlapping while the two channels of pair (1, 2) are overlapping. This is intuitive as it's unnecessary to maintain several channels with the same peer at the same time.



(a)



(b)

Figure 5.3: Channel temporal distribution: 5.3a shows communication performance for all node pairs, solid lines for ρ_a, dotted lines for ρ_s, red, black and blue lines correspond to S_1, S_2 and S_3; 5.3b shows communication performance for multi-channel pairs.

Chapter 6

Conclusions and Future Work

LN is a "Layer 2" system that aims to increase throughput of current Bitcoin network. We conduct a measurement study on LN in this thesis. Our graph construction of LN provides some insights behind the user behavior. Through routing analysis, we demonstrate the routing success rate relies heavily on the routing amount. Through resilience analysis, We identify a "robust yet fragile" structure in LN. Besides, our investigation of channel characteristics shows more efficient ways to use LN. The above observations indicate several issues of the current LN and suggest ways of future protocol design. Below we conclude our findings in this thesis and enumerate some directions for future work.

6.1 Improving Routing Efficiency

Our routing analysis in Section 4.1.3 and Section 4.1.4 revealed that for a large amount transaction (e.g., above the median of all channel capacities), the success rate of payment routing is low and is decreasing over time. Therefore, users who want to send large amount payments through LN should open a channel directly with their receivers. Besides, our analysis of pair payment channels in Section 5.2 indicated that the communication among most pairs are not stable. Hence, we suggest that some important nodes can serve as exchange markets maintaining stable communication and other users only need to connect to these hubs. In this way, the routing success rate can be greatly improved and the length of routing path can be

reduced.

As directions for future work, we propose studying all paths between a peer. For a large transaction, we can split it to multiple small-amount transactions and send them through different paths separately. In this way, we can make full use of those channels with small deposits and increase the success rate of routing. Another factor we can take account of is the channel fee. Nodes usually need to pay some fees to use others' channels as intermediaries for routing. A more general routing algorithm should not only find a path between users, but minimize the fees along the path.

6.2 Enhancing Network Resilience

Though the underlying Bitcoin Network is ideally decentralized, we found that the current LN relies heavily on some important nodes in Section 4.2.2. Users should be aware of such potential risks even if they can recover their balances after the attack. Furthermore, our work demonstrated that the topology among those important hubs can affect network resilience. Hence, it would be an interesting topic to adjust connections of exchange markets in order to get the best performance in terms of both routing efficiency and resilience.

In this thesis we simulated attacks to the LN just by removing top important nodes. Another interesting question that can be studied is to design more powerful attacking vectors. For example, attackers may collude to isolate some important nodes and partition the network.

6.3 Node Evaluation System

Nodes have many properties such as node capacity, channel numbers, node uptime and channel stability (metrics we defined in Section 5.2). As future work, we can apply the trust computation mechanisms [4, 21, 22] in Internet of Thing area to the scenario of LN and build a more reliable and efficient system based on the attributes of nodes. For example, we can build a node evaluation system in both subjective and objective ways. In the subjective model, each node computes the score of its neighbors locally on the basis of its own experience. In the objective model, the information about each node is distributed and stored making use of a distributed

hash table structure so that any node can make use of the same information.

Such node evaluation system can be maintained through a weighted combination of node properties. By giving higher weights to attributes of interest, users can select their neighbors more efficiently. In addition, such rating system can be extended to evaluate different channels. Nodes can score channels they have used for routing or rebalancing and send that score to others. This will helps others to choose paths when multiple ones are available.

6.4 Protocol Simulation

Several existing protocols in LN are closely related to the network topology [11, 29]. Thus, the use of real-world offchain network topology can better examine the performance of those protocols. In addition, the network topology we provide can be taken as a unified test platform for comparing similar protocols.

Bibliography

[1] V. Bagaria, S. Kannan, D. Tse, G. Fanti, and P. Viswanath. Deconstructing the blockchain to approach physical limits. *arXiv preprint arXiv:1810.08092*, 2018. → page 2

[2] A. Biryukov, D. Khovratovich, and I. Pustogarov. Deanonymisation of clients in bitcoin p2p network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 15–29. ACM, 2014. → page 6

[3] U. Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001. → page 29

[4] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, and X. Wang. Trm-iot: A trust management model based on fuzzy reputation for internet of things. *Comput. Sci. Inf. Syst.*, 8(4):1207–1228, 2011. → page 43

[5] T. Chen, Y. Zhu, Z. Li, J. Chen, X. Li, X. Luo, X. Lin, and X. Zhange. Understanding ethereum via graph analysis. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 1484–1492. IEEE, 2018. → page 5

[6] G. Dong, J. Gao, R. Du, L. Tian, H. E. Stanley, and S. Havlin. Robustness of network of networks under targeted attack. *Physical Review E*, 87(5): 052804, 2013. → page 29

[7] J. C. Doyle, D. L. Alderson, L. Li, S. Low, M. Roughan, S. Shalunov, R. Tanaka, and W. Willinger. The "robust yet fragile" nature of the internet. *Proceedings of the National Academy of Sciences*, 102(41):14497–14502, 2005. → page 33

[8] J. Gao, S. V. Buldyrev, S. Havlin, and H. E. Stanley. Robustness of a network of networks. *Physical Review Letters*, 107(19):195701, 2011. → page 29

[9] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 51–68. ACM, 2017. → page 2

[10] M. Green and I. Miers. Bolt: Anonymous payment channels for decentralized currencies. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 473–489. ACM, 2017. → page 5

[11] R. Khalil and A. Gervais. Revive: Rebalancing off-blockchain payment networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 439–453. ACM, 2017. → pages 3, 5, 44

[12] P. Koshy, D. Koshy, and P. McDaniel. An analysis of anonymity in bitcoin using p2p network traffic. In *International Conference on Financial Cryptography and Data Security*, pages 469–485. Springer, 2014. → page 5

[13] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Physical review letters*, 87(19):198701, 2001. → page 29

[14] M. Lischke and B. Fabian. Analyzing the bitcoin network: The first four years. *Future Internet*, 8(1):7, 2016. → page 5

[15] A. Miller, J. Litton, A. Pachulski, N. Gupta, D. Levin, N. Spring, and B. Bhattacharjee. Discovering bitcoin's public topology and influential nodes. 2015. → pages 5, 6

[16] A. Miller, I. Bentov, S. Bakshi, R. Kumaresan, and P. McCorry. Sprites and state channels: Payment networks that go faster than lightning. In *International Conference on Financial Cryptography and Data Security*, pages 508–526. Springer, 2019. → pages 3, 5

[17] S. Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008. → page 1

[18] T. Neudecker, P. Andelfinger, and H. Hartenstein. Timing analysis for inferring the topology of the bitcoin peer-to-peer network. In *Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences*, pages 358–367. IEEE, 2016. → page 5

[19] M. E. Newman. Mixing patterns in networks. *Physical Review E*, 67(2): 026126, 2003. → page 20

[20] M. E. Newman. Power laws, pareto distributions and zipf's law. *Contemporary physics*, 46(5):323–351, 2005. → page 18

[21] M. Nitti, R. Girau, L. Atzori, A. Iera, and G. Morabito. A subjective model for trustworthiness evaluation in the social internet of things. In *2012 IEEE 23rd international symposium on personal, indoor and mobile radio communications-(PIMRC)*, pages 18–23. IEEE, 2012. → page 43

[22] M. Nitti, R. Girau, and L. Atzori. Trustworthiness management in the social internet of things. *IEEE Transactions on knowledge and data engineering*, 26(5):1253–1266, 2013. → page 43

[23] M. Ober, S. Katzenbeisser, and K. Hamacher. Structure and anonymity of the bitcoin transaction graph. *Future internet*, 5(2):237–250, 2013. → page 5

[24] K. Okamoto, W. Chen, and X.-Y. Li. Ranking of closeness centrality for large-scale social networks. In *International Workshop on Frontiers in Algorithmics*, pages 186–195. Springer, 2008. → page 29

[25] J. Poon and T. Dryja. The bitcoin lightning network: Scalable off-chain instant payments. 2016. → pages 3, 4

[26] F. Reid and M. Harrigan. An analysis of anonymity in the bitcoin system. In *Security and privacy in social networks*, pages 197–223. Springer, 2013. → page 5

[27] M. Rigo. *Advanced graph theory and combinatorics*. John Wiley & Sons, 2016. → page 20

[28] D. Ron and A. Shamir. Quantitative analysis of the full bitcoin transaction graph. In *International Conference on Financial Cryptography and Data Security*, pages 6–24. Springer, 2013. → page 5

[29] S. Roos, P. Moreno-Sanchez, A. Kate, and I. Goldberg. Settling payments fast and private: Efficient decentralized routing for path-based transactions. *arXiv preprint arXiv:1709.05748*, 2017. → pages 3, 5, 44

[30] I. A. Seres, L. Gulyás, D. A. Nagy, and P. Burcsi. Topological analysis of bitcoin's lightning network. *arXiv preprint arXiv:1901.04972*, 2019. → page 5

[31] P. Wang, H. Xu, X. Jin, and T. Wang. Flash: Efficient dynamic routing for offchain networks. *arXiv preprint arXiv:1902.05260*, 2019. → page 5

[32] W. Xing and A. Ghorbani. Weighted pagerank algorithm. In *Communication Networks and Services Research, 2004. Proceedings. Second Annual Conference on*, pages 305–314. IEEE, 2004. → page 29